UNIVERSITÉ PARIS-EST

École doctorale Mathématiques et Sciences et Technologies de l'Information et de la Communication

Thèse de doctorat

Spécialité : Mathématiques

Présentée par

Laura Joana Silva Lopes

Pour obtenir le grade de

Docteur de l'Université Paris-Est

Numerical methods for simulating rare events in Molecular Dynamics

Abstract

In stochastic dynamical systems, such as those encountered in molecular dynamics, rare events naturally appear as events due to some low probability stochastic fluctuations. Examples of rare events in our everyday life includes earthquakes and major floods. In chemistry, protein folding, ligand unbinding from a protein cavity and opening or closing of channels in cell membranes are examples of rare events. Simulation of rare events has been an important field of research in biophysics over the past thirty years.

The events of interest in molecular dynamics generally involve transitions between metastable states, which are regions of the phase space where the system tends to stay trapped. These transitions are rare, making the use of a naive, direct Monte Carlo method computationally impracticable. To deal with this difficulty, sampling methods have been developed to efficiently simulate rare events. Among them are splitting methods, that consists in dividing the rare event of interest into successive nested more likely events.

Adaptive Multilevel Splitting (AMS) is a splitting method in which the positions of the intermediate interfaces, used to split reactive trajectories, are adapted on the fly. The surfaces are defined such that the probability of transition between them is constant, which minimizes the variance of the rare event probability estimator. AMS is a robust method that requires a small quantity of user defined parameters, and is therefore easy to use.

This thesis focuses on the application of the adaptive multilevel splitting method to molecular dynamics. Two kinds of systems are studied. The first one contains simple models that allowed us to improve the way AMS is used. The second one contains more realistic and challenging systems, where AMS is used to get better understanding of the molecular mechanisms. Hence, the contributions of this thesis include both methodological and numerical results.

We first validate the AMS method by applying it to the paradigmatic alanine dipeptide conformational change. We then propose a new technique combining AMS and importance sampling to efficiently sample the initial conditions ensemble when using AMS to obtain the transition time. This is validated on a simple one dimensional problem, and our results show its potential for applications in complex multidimensional systems. A new way to identify reaction mechanisms is also proposed in this thesis. It consists in performing clustering techniques over the reactive trajectories ensemble generated by the AMS method.

The implementation of the AMS method for NAMD has been improved during this thesis work. In particular, this manuscript includes a tutorial on how to use AMS on NAMD. The use of the AMS method allowed us to study two complex molecular systems. The first consists in the analysis of the influence of the water model (TIP3P and TIP4P/2005) on the β-cyclodextrin and ligand unbinding process. In the second, we apply the AMS method to sample unbinding trajectories of a ligand from the N-terminal domain of the Hsp90 protein.

Key words: rare events, molecular dynamics, adaptive multilevel splitting, cyclodextrin, alanine dipeptide, Hsp90

Résumé

Dans les systèmes dynamiques aléatoires, tels ceux rencontrés en dynamique moléculaire, les évènements rares apparaissent naturellement, comme étant liés à des fluctuations de probabilité faible. En dynamique moléculaire, le repliement des protéines, la dissociation protéine-ligand, et la fermeture ou l'ouverture des canaux ioniques dans les membranes, sont des exemples d'événements rares. La simulation d'événements rares est un domaine de recherche important en biophysique depuis presque trois décennies.

En dynamique moléculaire, on est particulièrement intéressé par la simulation de la transition entre les états métastables, qui sont des régions de l'espace des phases dans lesquelles le système reste piégé sur des longues périodes de temps. Ces transitions sont rares, leurs simulations sont donc assez coûteuses et parfois même impossibles. Pour contourner ces difficultés, des méthodes d'échantillonnage ont été développées pour simuler efficacement ces événement rares. Parmi celles-ci les méthodes de splitting consistent à diviser l'événement rare en sous-événements successifs plus probables. Par exemple, la trajectoire réactive est divisée en morceaux qui progressent graduellement de l'état initial vers l'état final.

Le Adaptive Multilevel Splitting (AMS) est une méthode de splitting où les positions des interfaces intermédiaires sont obtenues de façon naturelle au cours de l'algorithme. Les surfaces sont définies de telle sorte que les probabilités de transition entre elles soient constantes et ceci minimise la variance de l'estimateur de la probabilité de l'événement rare. AMS est une méthode avec peu de paramètres numériques à choisir par l'utilisateur, tout en garantissant une grande robustesse par rapport au choix de ces paramètres.

Cette thèse porte sur l'application de la méthode adaptive multilevel splitting en dynamique moléculaire. Deux types de systèmes ont été étudiés. La première famille est constituée de modèles simples, qui nous ont permis d'améliorer la méthode. La seconde famille est faite de systèmes plus réalistes qui réprésentent des vrai défis, où AMS est utilisé pour avancer nos connaissances sur les mécanismes moléculaires. Cette thèse contient donc à la fois des contributions de nature méthodologique et numérique.

Dans un premier temps, une étude conduite sur le changement conformationnel d'une biomolécule simple a permis de valider l'algorithme. Nous avons ensuite proposé une nouvelle technique utilisant une combinaison d'AMS avec une méthode d'échantillonnage préférentiel de l'ensemble des conditions initiales pour estimer plus efficacement le temps de transition. Celle-ci a été validée sur un problème simple et nos résultats ouvrent des perspectives prometteuses pour des applications à des systèmes plus complexes. Une nouvelle approche pour extraire les mécanismes réactionnels liés aux transitions est aussi proposée dans cette thèse. Elle consiste à appliquer des méthodes de clustering sur les trajectoires réactives générées par AMS.

Pendant ce travail de thèse, l'implémentation de la méthode AMS pour NAMD a été améliorée. En particulier, ce manuscript présente un tutoriel lié à cette implémentation. Nous avons aussi mené des études sur deux systèmes moléculaires complexes avec la méthode AMS. Le premier analyse l'influence du modèle d'eau (TIP3P et TIP4P/2005) sur le processus de dissociation ligand–β-cyclodextrine. Pour le second, la méthode AMS a été utilisée pour échantillonner des trajectoires de dissociation d'un ligand du domaine N-terminal de la protéine Hsp90.

Mots-clés: événements rares, dynamique moléculaire, adaptive multilevel splitting, cyclodextrine, alanine dipeptide, Hsp90

Contents

Chapter 1

Introduction

The objective of this work is to study the application of the adaptive multilevel splitting (AMS) method to the sampling of reactive trajectories and the estimation of transition times in molecular dynamics. A range of systems were used, which can be separated into two groups. The first one contains simple models that allowed us to propose improvements to the AMS method in general. The second one contains more realistic and challenging systems, where AMS is used to advance our understanding on the molecular mechanisms.

This chapter presents the framework of the thesis. The reader will find a description of molecular dynamics, force fields and a review of different methods to simulate rare events in molecular dynamics. Next, a summary of the main contributions, including a brief description of the encountered problems and the obtained results is presented.

1.1 Molecular Dynamics

Molecular Dynamics is the name given to the numerical method used to simulate molecules in vacuum or in solvent, assuming that the nuclei envolve following classical Newtonian dynamics plus possibly some terms to model the chosen thermodynamical ensemble. Introduced by Alder and Wainwright in the 50's, the goal was originally to describe and understand intrinsically multibody effects, like phase transitions[2], describing molecules as rigid spheres. The method became quickly popular among theoretical chemists and physicists and the first studies of liquids at a molecular level appeared in the literature in the 70's[3]. The raising interest in describing the behavior of large scale systems, for which the quantum approaches are still impossible, pushed the development of the model. In the last five decades, a range of molecular dynamics programs and classical potentials, called force fields, have been developed.

1.1.1 Langevin Dynamics

Langevin dynamics is typically used to describe the movement of atoms at a fixed temperature. This dynamics modifies deterministic Hamiltonian dynamics, which preserves energy, with stochastic terms, which model the fluctuations of the system due to temperature. Let us call (q_t, p_t) the positions and momenta at time t of the particles in $\mathbb{R}^{6N}$, where N is the number of atoms. Langevin dynamics models the evolution of (q_t, p_t) as follows:

$$\begin{cases} dq_t &= M^{-1}p_t dt, \\ dp_t &= -\nabla V(q_t)dt - \gamma M^{-1}p_t dt + \sqrt{2\gamma\beta^{-1}}dW_t. \end{cases} \tag{1.1}$$

In the equation above, V denotes the potential function, also called force field, M is the mass tensor and γ is the friction parameter. The process W_t is a Brownian motion in dimension $3N$. The multiplicative term in front of W_t depends on the temperature via the parameter $\beta^{-1} = k_B T$.

It is important to mention that, for a molecular system, the timestep of any numerical solution of Langevin dynamics is bounded from above. This is due to the natural oscillations caused by the covalent bonds, whose typical periods sets a maximum value for the timestep, such that the numerical solution is capable of simulating them accurately. The higher frequency oscillations are those of covalent bonds with hydrogen atoms. A C-H bond stretch in alkanes has a typical period around 10 femtosecond. This gives an upper bound on the timestep of the order of 1 fs. A typical strategy to raise the timestep is to fix all the lengths of the covalent bonds involving hydrogen atoms in the system, enabling a timestep of 2 fs[4].

1.1.2 Force Fields

Force field is the given name for the classical empirical potential V of the molecular system. This function includes two types of terms: those that try to carry a physical sense to the interactions; and those added when the latter terms alone are not able to predict the correct behavior of the molecule, and which have no clear physical interpretation. The first terms include the bonded terms, which describe the interactions between two to four bonded atoms, and depend on bond lengths, angles and dihedral angles; and the non-bonded terms, which describe the interactions between atoms which are not covalently bonded, in the same molecule or not, through a Coulomb and a Lennard-Jones potential. When the latter terms are not sufficient to reproduce the correct behavior of the molecule, other terms are added. For example, the improper term is a harmonic potential over a dihedral between non-bonded atoms. Functions that depends on two internal variables, called cross terms, can be used to model interactions between these internal degrees of freedom.

The chosen force field fixes the functional forms of the terms used to model the interactions. Once the functional form is chosen, the functions' parameters are empirically determined, through a fit over experimental or *ab initio* data. For example, for the CHARMM force field[5], the most common terms in the potential are given by:

$$\begin{aligned} V_{\text{CHARMM}} &= \sum_{\text{bonds}} K^b_{ij}\left(b_{ij} - b^0_{ij}\right)^2 + \sum_{\text{angles}} K^\theta_{ijk}\left(\theta_{ijk} - \theta^0_{ijk}\right)^2 + \sum_{\text{dihedrals}} K^\varphi_{ijkl}\left[1 + cos\left(n\varphi_{ijkl} - \delta\right)\right] \\ &\quad + \sum_{\substack{\text{nonbonded}\\\text{pairs}}} \frac{q_i q_j}{\epsilon r_{ij}} + \varepsilon_{ij}\left[\left(\frac{r^0_{ij}}{r_{ij}}\right)^{12} - 2\left(\frac{r^0_{ij}}{r_{ij}}\right)^6\right] + \sum_{\text{impropers}} K^\omega_{ij}\left(\omega_{ij} - \omega^0_{ij}\right)^2. \end{aligned} \tag{1.2}$$

For some types of molecules, which have common atom types and environments, generic parameters can be found. This is typically the case for proteins, where force fields like AMBER and CHARMM are well validated to describe their behavior[5, 6]. But this is only possible because proteins are composed of a small variety of repetitive units. For a few molecules studied in this thesis a specific, and thus non-transferable, force field was parameterized. It is important to mention that, because of the large number of parameters to be determined, the optimization problem is not easy to solve. There is a well established protocol to parameterize force fields by determining the most well-determined and important parameters first.

In this thesis, the CHARMM force field was used and the parameterizations were done with the help of the force field toolkit (FFTK) from VMD[7, 8], whose aim is to determine optimal parameters with respect to *ab initio* computations. When fitting a force field to *ab initio* data, the first step is to obtain the lowest energy positions of the nuclei, namely the optimized geometry. This will give the equilibrium values for the internal variables. The first goal of a force field is indeed to correctly describe the equilibrium conformation. The bonds and angles are described by a harmonic potential, which requires to compute the Hessian matrix for the system in order to obtain the force constants. The last parameters are the charges and the parameters related to the dihedrals angles, that are less direct and harder to determine.

Because charges are introduced to model the intramolecular interactions, they are primarily fitted to correctly describe the strongest bond between two molecules, which is the hydrogen bond with a water molecule. Thus, next to every donor and acceptor atom, a water molecule is placed and its distance and orientation are optimized with a quantum calculation. Using this data, and starting from a first guess that is given by the user, the charges are optimized, generally using a simulated annealing algorithm. This optimization is not easy and sometimes needs additional *ab initio* data to converge.

The torsion parameters, associated with the dihedral angles, are determined by fitting the result of an *ab initio* relaxed energy scan. This means that, for a range of values around the equilibrium, the dihedral angle is fixed and the geometry is optimized. The result is the energy as a function of the dihedral angles, which is then fitted using a periodic cosine function. Despite the apparent simplicity of this step, it is in this last stage that the necessity to use improper or cross terms is discovered. For example, in the parameterization presented in Chapter 7, a cross term was needed to describe the energy variation caused by two dihedral angles that had three atoms in common, and thus were correlated and could not be computed as a sum of two terms. For that case we used a CMAP correction, which is a grid based energy correction function of two dihedral angles, firstly introduced in 2004 to better describe protein backbones in CHARMM[9].

Another particularity of the force field parameterized in Chapter 7 was the use of two fragments of the molecule to calculate the bond, angle and charge parameters. This was necessary because the molecule was too large to compute the full Hessian. These fragments were again used when the charge optimization revealed a spurious dipole in the molecule, which was corrected by fitting the charges to both fragments separately.

The parameterization of the force field is an essential step to reach reliable results. It is important to mention that classical force fields are currently the cheapest way to calculate the energy of the system, and thus enable the calculation of the classical dynamics of large systems using the current computational resources. Notice that, during the dynamics, the forces of the system need to be computed at every timestep, which in molecular dynamics is limited by a maximum of 2 fs. For different problems one can use more elaborated force fields, including polarization effects or even the possibility of a chemical reaction, which imply of course a larger computational cost.

1.1.3 The NAMD molecular dynamics software

All the molecular simulations in this thesis were performed using the NAMD program[10]. NAMD is a molecular dynamics program that has a good scalability and is thus appropriate to simulate large scale systems on different architectures. For this reason, it is widely used by the biophysics community. The implementation of the AMS method in NAMD was initiated by C. Mayne and I. Teo in [11], and was pursued in the framework of this PhD thesis (see Chapter 5). The AMS method was implemented in Tcl, the language used by NAMD to parse the configuration file. It is therefore easy to implement a new method for this program. It is also important to mention that the AMS method requires the definition of a progress function, called a reaction coordinate, that depends on internal variables of the system. The Colvars plugin for NAMD/VMD[12], which has Jérôme Hénin as one of its developers, was used to easily obtain the collective variables to compute the reaction coordinate.

1.1.4 Metastable states and their transitions

Metastable states are defined as regions where the system stays trapped for a considerable amount of time. Hence, transitions between two metastable states, or the escape from one, are rare events. Examples of rare events in our everyday life includes earthquakes or major floods. In chemistry, protein folding, ligand unbinding from a protein cavity and opening or closing of channels in cell membranes are examples of rare events.

Rare events are, by definition, very difficult to simulate by brute force Monte Carlo methods, since observing them requires many independent trials. Over the past thirty years, a range of methods were specially developed to simulate transitions between metastable states. They can be divided into two families: biased and non-biased methods. The first family consists in methods where the dynamics is biased in order to push the system out of the metastable state faster, typically to compute thermodynamic quantities. This includes adiabatic bias molecular dynamics (ABMD)[13], and free energy methods, such as adaptive biasing force[14]. The second family aims at obtaining kinetic information about the transition. The dynamics is not biased and other strategies are used in order to shorten the computational cost. AMS, which is the method studied in this work, belongs to this second family.

1.2 Methods for the simulation of reactive paths

In this thesis, we will focus on unbiased methods to obtain kinetic information about reactive paths. The goal is in particular to obtain the transition time, or its inverse, the transition rate. Let us call A a metastable region from which we want to simulate escapes, and B a target state. Let us consider a system with N atoms, so the dynamics is over $\mathbb{R}^{6N}$ (position and momentum for all particles). This means that A and B are subsets of $\mathbb{R}^{6N}$. In chemistry, those states are defined using a small set of internal variables, that in practice actually only depends on the positions of the particles.

The reader will find a brief discussion about transition path theory, including the definition we will use for transition time and reaction rate, in section 1.2.1. Then we discuss different equations used to calculate the transition time and the transition rate in section 1.2.2. Next we present a summary of methods commonly used in molecular dynamics to simulate transition paths. Section 1.2.3 focus on transition path sampling and its derivatives, and section 1.2.4 on splitting methods.

1.2.1 Transition Path Theory

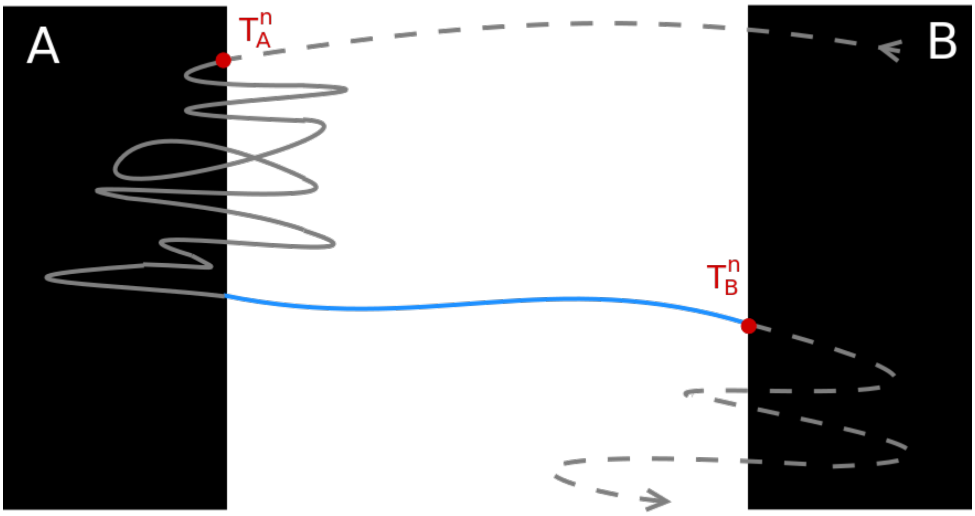


Figure 1.1 – Fragment of an equilibrium trajectory. The blue segment corresponds to a reactive trajectory between states A and B. The transition time is the average duration of trajectories like the one represented by the solid line.

Let us consider a trajectory at equilibrium. By ergodicity, the two states A and B are visited infinitely many times. Let us consider the successive first entrances in one of those states, after having visited the other one (see the red dots on figure 1.1). We call $(T_A^n)_{n\geq 0}$ the times for which the entrance is in A, and $(T_B^n)_{n\geq 0}$ in B. The segments between T_A^n and T_B^n are called the transition paths from A to B. Notice that those trajectories contain a path that links state A to state B (in blue on figure 1.1), called the reactive trajectory. The average duration of the transition paths at equilibrium is called the transition time[15, 16]. The transition time is then defined as:

$$T_{AB} = \lim_{N\to+\infty} \frac{1}{N} \sum_{n=1}^{N} T_B^n - T_A^n.$$

It is also common to define the transition rate, which is the inverse of the transition time:

$$k_{AB} = \frac{1}{T_{AB}}.$$

The transition path theory gives formulas for these quantities using the committor function, which for each point in space measures the probability to reach B before A starting from that point, see for example Proposition 1.8 in [15].

1.2.2 Computing the transition time

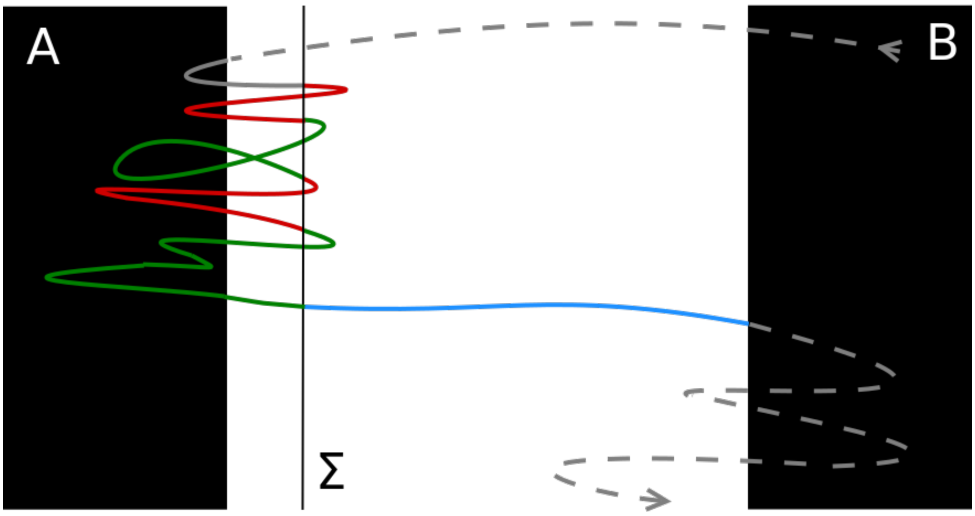


Figure 1.2 – Decomposition of the transition path using an intermediate region Σ near A, in order to calculate the transition time via the probability of transition starting from Σ at equilibrium.

One can calculate the transition time using the probability of transition starting from an intermediate region Σ in the neighborhood of A. For that, the transition path is splitted into pieces at every time it crosses Σ, if A was visited just before (see figure 1.2). Whenever the particle crosses Σ there are two possible events: going back to A or reaching B. This is a Bernoulli law, and if we call p the probability at equilibrium to reach B starting from Σ, the system will go back to A a number $1/p-1$ of times before reaching B. Let us call $\mathbb{E}\left(T_{\text{loop}}\right)$ the average time of those returns to A at equilibrium, passing through Σ. The total time spent doing loops, from A to Σ and back to A, before a transition can be computed as $(1/p-1)\mathbb{E}\left(T_{\text{loop}}\right)$. Calling $\mathbb{E}\left(T_{\text{reac}}\right)$ the average reactive trajectory duration at equilibrium, the transition time can thus be computed as:

$$\mathbb{E}\left(T_{AB}\right) = \left(\frac{1}{p} - 1\right)\mathbb{E}\left(T_{\text{loop}}\right) + \mathbb{E}\left(T_{\text{reac}}\right). \tag{1.3}$$

Notice that Σ can be chosen as the border of A, which does not change the equation above.

Equation (1.3) is used to compute the transition time in the AMS method (see [17] and Chapter 3). Other rare event methods[18] use the following equation to obtain the transition rate:

$$k_{AB} = \frac{p}{\mathbb{E}\left(T_{\text{loop}}\right)}. \tag{1.4}$$

Notice that, if the probability p is small, the second term of equation (1.3) is negligible compared to the first. Hence, equation (1.4) is equivalent to equation (1.3) in this regime.

Another common way to compute the transition rate is via the correlation function $C(t)$, which measures the probability to be inside B at time t, if the particle was inside A at the initial time. Let us introduce the function h_A (resp. h_B) that is unity inside A (resp. B) and zero elsewhere. The correlation function $C(t)$ is defined as:

$$C(t) = \frac{\langle h_A(x_0) h_B(x_t) \rangle}{\langle h_A(x_0) \rangle}, \tag{1.5}$$

where x_t is the position of all the particles in the system at time t. This function is linear in the long time limit, and the linear constant is the reaction rate, i.e. $C(t) \approx k_{AB}t$[18].

1.2.3 Transition Path Sampling

Transition path sampling (TPS) is a method introduced in the 90's [19], were the reactive trajectories are sampled directly in the path space using a Metropolis Monte Carlo algorithm. A first reactive trajectory is generated and a new one is obtained from the first in a trial move. The new trajectory have a probability of acceptance that is nonzero only if it is also reactive. There are various versions of the algorithm, which differ in the kernel to propose new trajectories, the calculation of the acceptance probability, and the calculation of the transition rate.

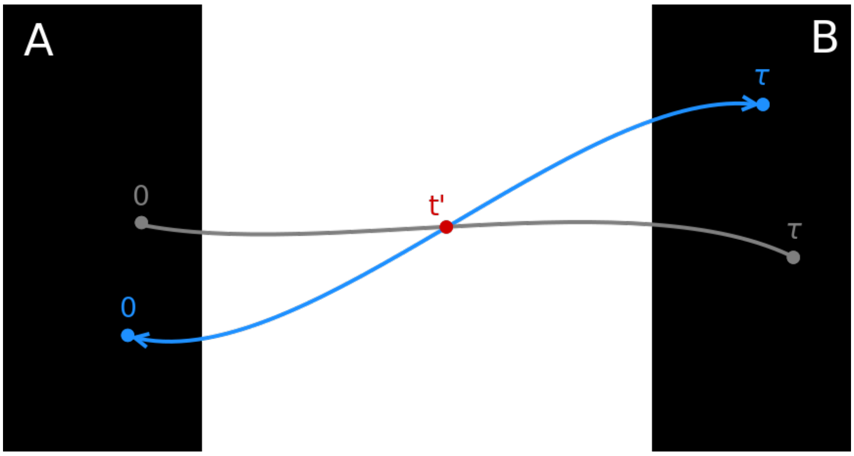


Figure 1.3 – The shooting move used to generate new trajectories in TPS and its derivatives. The gray trajectory is the old one. From the shooting point (red dot) the momentum of the atoms is changed and the dynamics is run forward and backward for the same number of timesteps. In other versions the new segments are run until it reaches A or B, in order to allow the trajectory duration to vary.

In the original version of TPS, new trajectories are generated through the so-called shooting move, starting from a randomly chosen point of an existing trajectory of fixed size τ. Let us call $(x_t^o)_{t\in[0,\tau]}$ the first trajectory, and $x_{t'}^o$ the chosen point, called the shooting point. A perturbation in the momentum of the atoms is done, generating a new point $x_{t'}^n$, from where the dynamics is run both forward and backward, in order to complete the new path of equal size τ, $(x_t^n)_{t\in[0,\tau]}$. The probability to accept this trial move is only nonzero if the new trajectory is reactive, and depends on the density of the first points of both trajectories, as:

$$\mathbb{P}_{\text{acc}}(o \to n) = h_A\left(x_0^n\right) h_B\left(x_\tau^n\right) \min\left[1, \frac{\rho\left(x_0^n\right)}{\rho\left(x_0^o\right)}\right].$$

The final transition rate is calculated using the correlation function from equation (1.5).

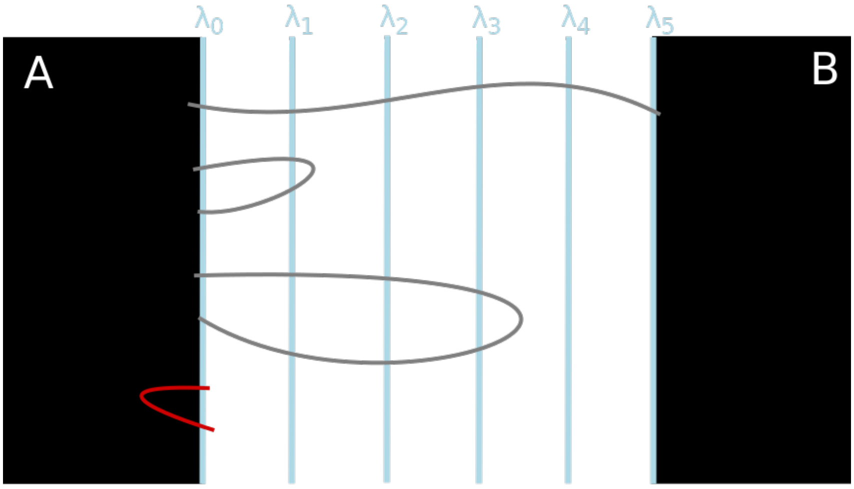


Figure 1.4 – Space between A and B split using five interfaces, and a few trajectories. The first trajectory is reactive. The second crosses λ_1, so it belongs to the path ensemble $[1^+]$. The third goes further, and hence belongs to $[3^+]$. The red trajectory is only considered in RETIS, and belongs to the path ensemble $[0^-]$.

A variation of the TPS method, called transition interface sampling (TIS), was later developed, in which the rate constant is calculated using the transition probability via equation (1.4)[1]. Using isolevel surfaces of an order parameter $\lambda : \mathbb{R}^{6N} \to \mathbb{R}$, the space between states A and B is split. The state A is defined as $\{x, \lambda < \lambda_0\}$, and a last interface n defines state B as $\{x, \lambda > \lambda_n\}$. The probability p in equation (1.3) is obtained as the product of the conditional probabilities to reach one interface starting from the previous one, as:

$$p = \prod_{i=0}^{n-1} \mathbb{P}\left(\lambda_i \to \lambda_{i+1}\right).$$

The interfaces also defines the path ensembles $[i^+]$, that contains all the trajectories that crosses the interface λ_i.

The trajectories in TIS are generated by performing either a time reversal move or a shooting move. In the first move, a new trajectory is generated by changing the time direction of the original path. The second move is similar to the one performed in TPS, but the duration of the trajectories vary. Starting from the shooting point, the segments are run forward and backward in time until A or B is reached. Differently from TPS, the acceptance probability is nonzero if the new trajectory starts in A and crosses either the same or a further interface than the original trajectory. This means that if the original trajectory belongs to $[i^+]$, the new trajectory must at least be in the same path ensemble. This makes TIS a more efficient algorithm than the original TPS, as more trajectories are accepted.

The replica exchange transition interface sampling (RETIS) is a variation of TIS where the trajectories are generated in the same way, but additional swapping moves are performed between the path ensembles[1]. Those decrease the correlation between the trajectories in the same ensemble, leading to a more efficient algorithm. An additional ensemble $[0^-]$ is also considered, which contains the trajectories that explore state A, i.e. cross interface λ_0 but in the direction of state A.

1.2.4 Splitting methods

In splitting methods, one introduces intermediate interfaces between A and B, as in TIS, but the way the trajectories are sampled is different. The strategy is to simulate paths that link two successive interfaces. The final transition probability is again calculated as a product of the probability to pass from each interface to the next one.

Forward flux sampling (FFS) is a commonly used splitting method[18]. All the points that crosses the interfaces are kept, and are then used to generate new attempts to observe trajectories that reaches the next interface. These attempts are also used to compute the probability between the interfaces. In FFS, the interfaces that split the space are fixed as level sets of a chosen scalar valued order parameter (a.k.a. reaction coordinate). Let us mention that there exists an adaptive version of the algorithm were their positions are set after a few FFS runs, in order to minimize the variance of the probability estimator[20].

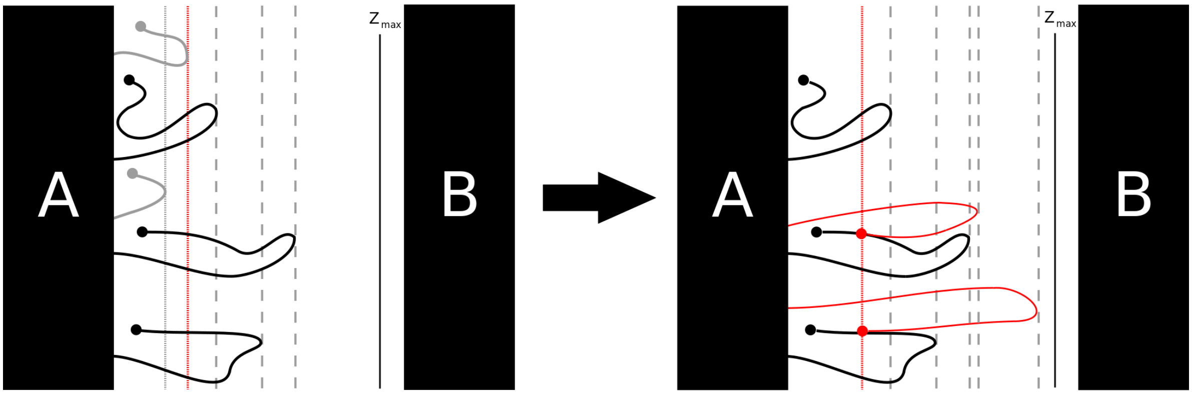


Figure 1.5 – First AMS iteration with $N = 5$ and $k = 2$. Both lower level replicas (in gray) are killed. Two of the remaining replicas are randomly selected to be copied up to level z^0_{kill} (dotted red line) and then continued until they reach A (typically more likely) or B.

In the adaptive multilevel splitting (AMS) method, the interfaces are set on the fly[21]. Each trajectory is associated with a level, which is the maximum reached value of the reaction coordinate along the trajectory. At each iteration, the k^{th} trajectory level, called the killing level, defines a new interface. All the trajectories with equal or lower level are killed, and replaced by randomly chosen and replicated trajectories among the living ones. The replication consists in copying the trajectory until the first point that went further than the killing level, and running the dynamics independently from that point until A or B is reached (see figure 1.5). A more detailed description of the algorithm is given in Chapter 2.

This way of positioning the interfaces optimizes the variance of the estimator. This also makes AMS a method with very few user defined parameters, which is thus more robust and easy to use. A mathematical proof of the unbiasedness of the probability estimator whatever the algorithm parameters, which are the total number of trajectories, k and the order parameter, can be found in [22].

1.3 Outline of this manuscript

In this thesis, two kinds of studies were done, and for this reason the chapters are organized in two parts. Part I contains the methodological results, which rely on simulations on toy models and includes chapters 2, 3 and 4. Chapter 2 contains the first study, done using a simple biomolecule. Two questions raised during that project led to the next chapters of Part I. The first one concerns the sampling of the initial conditions in AMS to compute the transition time. We propose a new technique combining AMS and importance sampling, presented in Chapter 3. The second one is about using the reactive trajectories generated with AMS to elucidate reaction mechanisms, for which we propose clustering techniques, presented in Chapter 4. Part II contains numerical results on more complicated molecular systems studied thanks to the AMS method, and includes chapters 5, 6 and 7. Chapter 5 contains the tutorial of the AMS implementation on NAMD. Chapter 6 presents the study of the influence of the water model in the β-cyclodextrin and ligand unbinding process. Chapter 7 contains the application of the AMS method to sample unbinding trajectories of a ligand from a protein cavity. In the following sections, we present a brief summary of each of these projects.

It is important to mention that the chapters of this manuscript are written in a self contained form, so that the reader can find all the necessary information to understand the project in the same chapter. This implies some repetitions from one chapter to another, in particular, the description of the AMS algorithm and the formula used to compute the transition time. The most precise and complete description of those are found in chapters 2 and 3, respectively.

Chapter 2: Alanine di-peptide

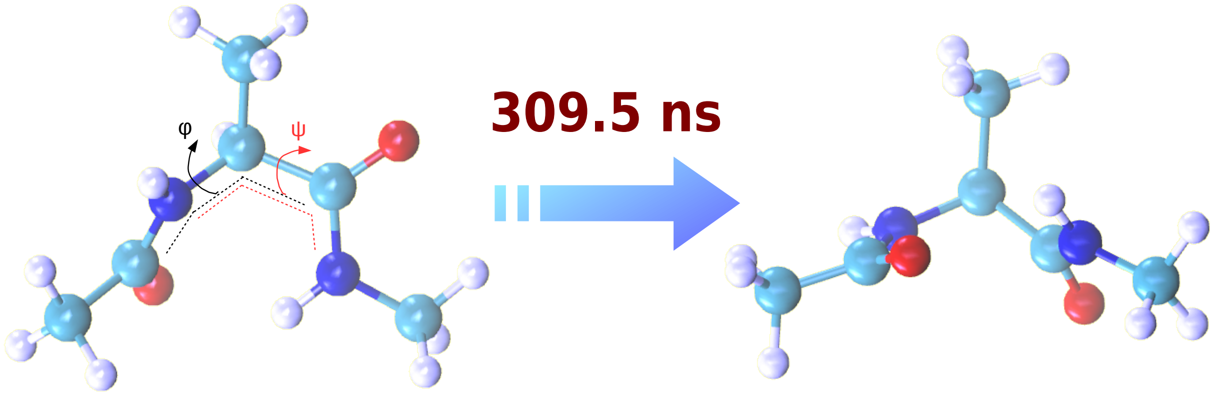


Figure 1.6 – The two stable conformations of the alanine di-peptide molecule and the dihedral angles ϕ and ψ used to distinguish them.

The first studied system is a molecular toy model, the alanine di-peptide. This molecule is small and exhibits two stable conformations in vacuum. Because of its similarity with peptides, which has an important role in protein folding, one of the most difficult process to simulate, this model became a commonly used system to test new methods for the analysis of biomolecular systems. Figure 1.6 shows the two conformations of the alanine di-peptide, easily described by two dihedral angles.

The study of the conformational changes of this molecule allows us to first validate AMS against brute force results, and demonstrates the robustness of the AMS results, in particular with respect to the choice of the reaction coordinate. Moreover, we propose a new protocol to correctly sample the initial condition when using AMS to obtain the transition time. We also explain how to estimate two interesting quantities using the trajectories generated by AMS in order to explore the reaction pathways. The first one is the flux of reactive trajectories, and the second one is the committor function. Results of this study were published in the Journal of Computational Chemistry[23].

Chapter 3: 1D potential

In Chapter 3, we study transitions on the one dimensional potential $V(x) = x^4 - 2x^2$, which exhibits two metastable states, around $x = -1$ and $x = +1$. This was also studied by T. van Erp in [1], where the FFS and RETIS methods were applied. In [1], the results obtained by FFS do not coincide with reference results.

First we perform the same numerical experiment as in [1] using AMS, and show that, despite the apparent simplicity of the problem, the sampling of initial conditions when using AMS, and thus also FFS, is crucial to obtain consistent results. We thus explain and propose a solution to the numerical observations of [1]. We then propose a new technique, combining AMS and importance sampling, to more efficiently sample the initial conditions, which we validate on this one dimensional case. We also discuss how to apply this technique to multidimensional cases.

Chapter 4: Clustering of reactive trajectories

In this work we propose a new way to extract reaction mechanisms, by performing a clustering on the ensemble of reactive trajectories obtained with AMS. Obtaining the transition mechanism is an old problem. The literature on the subject does not provide a clear definition of a transition mechanism[24–27]. Moreover, many of the previous works assume that there is only one possible mechanism, which is not always the case in complex systems. The transition tubes introduced by Vanden-Eijnden in [16] was the first method to consider more than one mechanism, but they are not uniquely defined. By performing a clustering of the reactive trajectories, the representative paths from each cluster can be considered as a possible reaction mechanisms. In addition, the clustering technique not only enables the existence of more than one mechanism, but also gives a weighting probability to each one.

Chapter 4 presents preliminary results obtained using two systems. The first one is a bi-channel potential in dimension two, where the temperature influences the preferable path, which is seen by the difference in the cluster weights. The second is the alanine di-peptide starting from different initial conditions, where the number of mechanisms vary. This study was made in collaboration with Jacques Printems, from Université Paris-Est Créteil.

Chapter 5: AMS tutorial for NAMD

In this thesis the implementation of the AMS method for NAMD in Tcl was improved, and a set of bash scripts was written in order to provide a more easy way to use the method. One can set AMS simulations providing one simple configuration file and a few scripts to set the parameters of the algorithm, including the reaction coordinate. In order to diffuse the method among the NAMD community, a tutorial based on the conformational change of the alanine dipeptide molecule was written. This tutorial is published in the NAMD tutorials webpage, that also provides all the files necessary to complete the tutorial. Chapter 5 presents the published tutorial.

Chapter 6: β-cyclodextrin with ligand

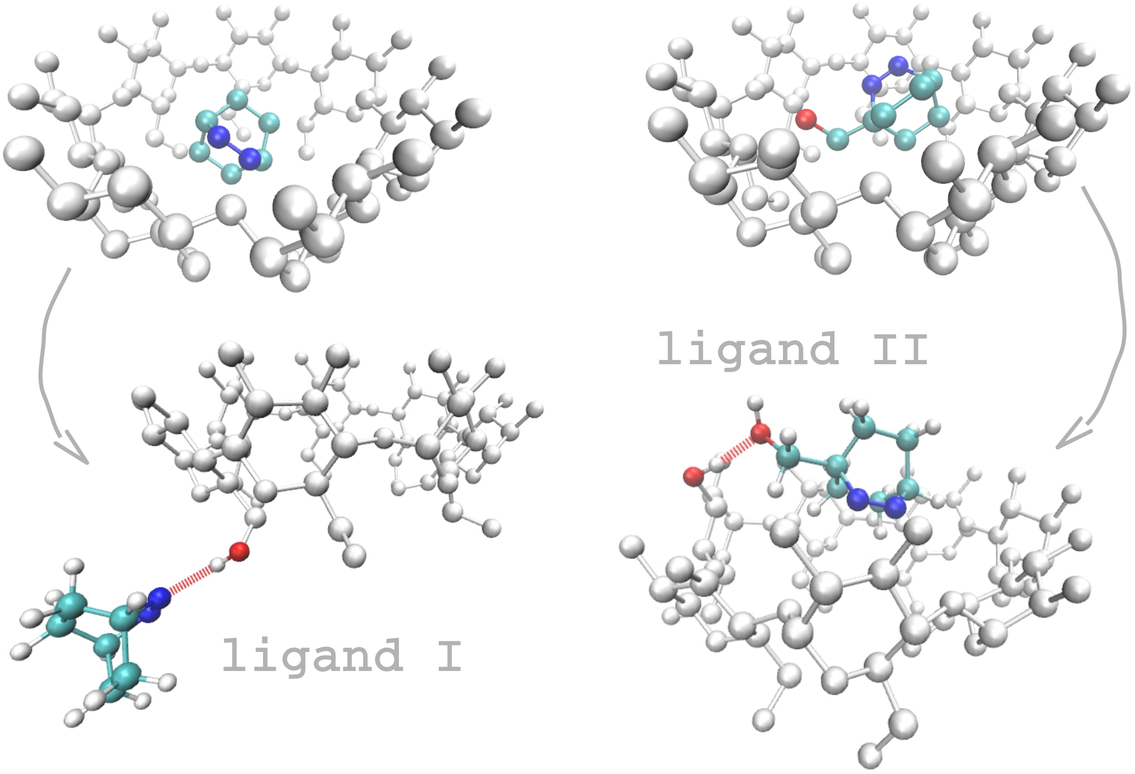


Figure 1.7 – β-cyclodextrin with ligands I and II, and two frames of an unbinding trajectory generated with AMS. Ligand I exits from the bottom, still maintaining contact with the β-cyclodextrin. Ligand II exits from the top, and its contact is done through its hydroxyl group. Our results showed that the contacts between the ligand and the trap have a significant role in the unbinding mechanism.

Cyclodextrins are a family of molecules formed by repeated glucopyranose units, generating a ring structure with a hydrophobic interior and a hydrophilic exterior. Hence, cyclodextrins have the ability of increasing the solubility of hydrophobic molecules in water, and are thus useful for many industrial applications[28]. In this thesis, we simulate the unbinding of two different ligands from the β-cyclodextrin interior to an aqueous environment. The goal of this project is thus to apply AMS to a more complex case and compare our findings with published experimental results for the ligand's unbinding rate[29]. We show that the AMS calculations give reliable result, with a computational cost divided by 400 when comparison with direct numerical simulations is possible.

We observed some discrepancies between the results from the molecular dynamics model and the experimental results. Willing to gain more knowledge about this system, we then change the water model and explore its influence on the unbinding process.

Since the ligands stay trapped in the interior of the β-cyclodextrin because of its hydrophobic interior, its is clear that the water plays an important role in the ligand's escape. This is a property also seen in other unbinding processes from cage molecules, and thus the analysis of the influence of the water model is of general interest. We thus compare two commonly used water models in biomolecular systems: TIP3P and TIP4P/2005[30, 31].

Our results show no significant difference in the unbinding mechanism, meaning that the change from TIP3P to TIP4P/2005, a more computationally costly model, does not interfere in the described behavior of the molecules. However, a significant difference is observed for the unbinding time. This is caused by variations of the diffusion coefficients and, to a less extent, by the varying lifetime of the H-bonds between the ligand and the β-cyclodextrin. It is also important to mention that we still need to explore other effects to entirely explain the difference seen in the unbinding times, such as the solvation free energyof the ligands in water.

Chapter 7: Heat Shock Protein 90

Heat shock protein 90 is a human chaperone protein that is overexpressed in some types of cancer, making those cancerous cells more sensitive to drugs that blocks this protein activity. A typical target in this protein is its N-terminal domain, that binds ATP to power the protein's functional cycle. Because the drug's efficiency depends on its residence time in the binding site, it is important to obtain an estimation for the unbinding time when searching for a new drug. In this chapter we present a project made in collaboration with researchers from the pharmaceutical company Sanofi, where we apply the AMS method to simulate the unbinding of a drug candidate from the N-terminal domain of Hsp90.

The crystallographic structure of the ligand inside the binding site was first provided by Sanofi, and then later published as Protein Data Bank id 5LR1 (see figure 1.8). The uncommon structure of the ligand requires a force field parameterization based on a CMAP cross term, where the molecule was divided into two fragments to enable the calculation of the bond and angle force constants, as well as the charges.

The determination of the bound metastable state was made using a first simulation following the free dynamics, starting from the crystallographic structure. In addition, adiabatic bias molecular dynamics (ABMD) simulations[13] were used to explore possible additional metastable states, bound or intermediate. This approach showed to be incomplete, and it was with AMS that two other bound states were actuallyfound.

Using these AMS results, and together with free energy calculations, we were able to determine the nature of the newly found states, and to propose a new reaction coordinate and a new definition for the bound state. The simulations using this new setting are currently running, and four unbinding

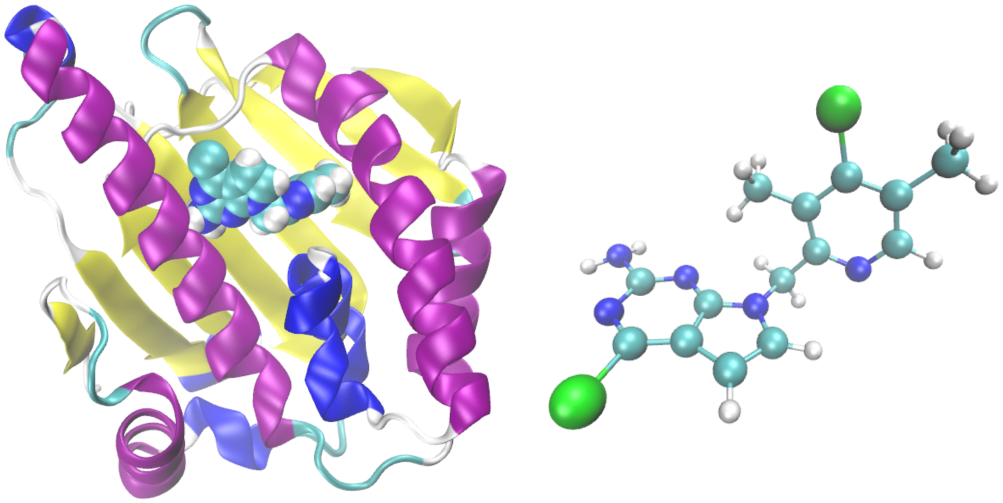


Figure 1.8 – N-terminal part of Hsp90, with ligand inside its cavity (structure PDB 5LR1), and the ligand molecule (A003498614A).

trajectories were generated until this moment. All the simulations were performed using the HPC resources from GENCI [Occigen].

Part I

Methodology

Chapter 2

Characterizing AMS using a simple biomolecule

Results of this chapter are published at the Journal of Computational Chemistry (2019).

Analysis of the Adaptive Multilevel Splitting method on the isomerization of alanine dipeptide

Laura J. S. Lopes, Tony Lelièvre
CERMICS, École des Ponts ParisTech, 6-8 avenue Blaise Pascal, 77455 Marne-la-Vallée, France

We apply the Adaptive Multilevel Splitting method to the $C_{eq} \to C_{ax}$ transition of alanine dipeptide in vacuum. Some properties of the algorithm are numerically illustrated, such as the unbiasedness of the probability estimator and the robustness of the method with respect to the reaction coordinate. We also calculate the transition time obtained via the probability estimator, using an appropriate ensemble of initial conditions. Finally, we show how the Adaptive Multilevel Splitting method can be used to compute an approximation of the committor function.

2.1 Introduction

Simulation of rare events has been an important field of research in biophysics for nearly two and a half decades now. The goal is to obtain kinetic information for processes like protein (un)folding or ligand-protein (un)binding. A typical quantity of interest is the transition rate, or equivalently its inverse, the transition time. This quantity is, for example, directly related to drug-target affinity, making its calculation an important step in drug design[32]. The committor function, which gives the probability to reach a targeted conformation before going back to the initial one, is also interesting for computational and modeling purposes[16].

The events of interest in molecular dynamics generally involve transition between metastable states, which are regions of the phase space where the system tends to stay trapped. These transitions are rare, making the simulation too long and sometimes even computationally impracticable. To deal with this difficulty, sampling methods have been developed to efficiently simulate rare events. Among them are splitting methods, that consists in dividing the rare event of interest into successive nested more likely events. For example, a reactive trajectory is divided into pieces which gradually progress from the initial state to the target one. Examples of splitting methods include Milestoning[33], Weighted Ensemble[34], Forward Flux Sampling[35] and Transition Interface Sampling[36]. In these methods, the intermediate milestones or dividing surfaces, used to split the rare event of interest, are fixed, so they are parameters that should be defined in advance. Let us however mention that there exists an adaptive version of the Forward Flux Sampling method[35], in which a few preliminary runs enable to optimize the position of the dividing surfaces.

The Adaptive Multilevel Splitting (AMS) method[21] is a splitting method in which the positions of the intermediate interfaces, used to split reactive trajectories, are adapted on the fly, so they are not parameters of the algorithm. The surfaces are defined such that the probability of transition between them is constant, which are known to be the best surfaces in terms of the variance of the rare event probability estimator[37]. Moreover, as illustrated in this paper, the method gives reliable results for a large class of sensible reaction coordinates, making it particularly straightforward to use for practitioners. This method has been used with success to estimate rare events probabilities in many contexts. In particular, the AMS method was already efficiently applied to a large scale system to calculate unbinding time[11]. Let us emphasize that the AMS algorithm can be used not only to estimate the probability of a rare event, but also to simulate the associated rare events (typically, the ensemble of reactive trajectories in the context of molecular dynamics). This allows us to study the possible transition mechanisms, that are often more than one, and to estimate the committor function, for example.

Compared to previous publications on AMS[11, 38], we provide in this paper a full description of the correct way to implement the algorithm in a discrete in time setting. The reader will find this description in Section 2.2, as well as a brief discussion of some important properties of the method and the way to obtain the transition time using AMS. We apply the method to a toy problem, namely the isomerization of alanine dipeptide in vacuum ($C_{eq} \to C_{ax}$ transition). In this small example, we are able to numerically illustrate the consistency and the unbiasedness of the AMS method, as well as to explore in details its properties, by comparing the results to brute force direct numerical simulation. These numerical results are reported in Section 2.3. They illustrate the interest of the method and lead us to draw useful practical recommendations to get reliable results with AMS.

2.2 Methods

Assume that the simulations are done using Langevin dynamics. Let us denote by $\mathbf{X}_t = (\mathbf{q}_t, \mathbf{p}_t) \in \mathbb{R}^{2d}$ the positions and momenta of all the particles in the system at discrete time t, d being three times the number of atoms. The vector $\mathbf{X}_t$ evolves according to a time discretization of the Langevin dynamics such as:

$$\left\{\begin{array}{lcl} \mathbf{p}_{t+\frac{1}{2}} & = & \mathbf{p}_t - \frac{\Delta t}{2}\nabla V(\mathbf{q}_t) - \frac{\Delta t}{2}\gamma M^{-1}\mathbf{p}_t \\ & & +\sqrt{\Delta t \gamma \beta^{-1}}\mathbf{G}^t \\ \mathbf{q}_{t+1} & = & \mathbf{q}_t + \Delta t M^{-1}\mathbf{p}_{t+\frac{1}{2}} \\ \mathbf{p}_{t+1} & = & \mathbf{p}_{t+\frac{1}{2}} - \frac{\Delta t}{2}\nabla V(\mathbf{q}_{t+1}) \\ & & -\frac{\Delta t}{2}\gamma M^{-1}\mathbf{p}_{t+1} + \sqrt{\Delta t \gamma \beta^{-1}}\mathbf{G}^{t+\frac{1}{2}}. \end{array}\right. \tag{2.1}$$

Here, V denotes the potential function, M is the mass tensor, γ is the friction parameter, $\beta^{-1} = k_B T$ is proportional to the temperature, and $(\mathbf{G}^t, \mathbf{G}^{t+\frac{1}{2}})_{t\geq 0}$ is a sequence of independent centered Gaussian vectors with covariance identity. Let us emphasize that, although we use this dynamic as an example to present the algorithm, it applies to any Markovian stochastic dynamics (like overdamped Langevin, Andersen thermostat, kinetic Monte Carlo, etc...).

Let us call A and B the source and target regions of interest. The goal is to sample reaction trajectories, linking A and B, and to estimate associated quantities. Both A and B are subsets of $\mathbb{R}^{2d}$, although in practice, they are typically defined only in terms of positions. In addition, assume that A is a metastable region for the dynamics, which means that, starting from a point in the neighborhood of A, the trajectory is most likely to enter A before visiting B. The progress from A to B is measured by a reaction coordinate ξ, i.e. a real-valued function defined over $\mathbb{R}^{2d}$, whose values will be called levels. Again, in practice, ξ typically only depends on the positions of the atoms. The function ξ is assumed to satisfy the following condition:

$$\exists\, z_{max} \in \mathbb{R} \text{ such that } B \subset \xi^{-1}(]z_{max}, +\infty[), \tag{2.2}$$

that makes necessary to exceed a level z_{max} of ξ to enter B when starting from A. Let us emphasize that this is the only condition we assume on ξ in the following: the algorithm can thus be applied with many different reaction coordinates.

Note that the definitions of the zones A and B are independent of the reaction coordinate. Since ξ does not need to be continuous, the former condition can be enforced by just forcing ξ to be infinity on B. More precisely, if a function $\tilde{\xi}$ is a good candidate for the reaction coordinate but does not satisfy the previous condition (2.2), it is possible to obtain ξ from $\tilde{\xi}$ by setting:

$$\xi(\mathbf{X}) = \begin{cases} \tilde{\xi}(\mathbf{X}) & \mathbf{X} \in \mathbb{R}^{2d} \setminus B \\ \infty & \mathbf{X} \in B. \end{cases} \tag{2.3}$$

The condition (2.2) is then satisfied with z_{max} equal to the maximum value of $\tilde{\xi}$ outside B.

We will focus on the estimation of the probability to observe a reaction trajectory, that is, coming from a set of initial conditions in $\mathbb{R}^{2d} \setminus (A \cup B)$, the probability to enter B before returning to A. Let us call τ_A and τ_B the first hitting times of A and B, respectively (see equations (2.4) and (2.5) below). What we aim to calculate is then the probability $\mathbb{P}(\tau_B < \tau_A)$. As will be further explained, this probability can be used to compute transition times. As mentioned earlier, AMS also yields a consistent ensemble of reactive trajectories (this will be illustrated in Section 2.3).

A detailed description of the AMS algorithm is given in Section 2.2.1. Some interesting features of the method are presented in Section 2.2.2. In Section 2.2.2 we present a brief discussion of some interesting features of the method. In Section 2.2.3 we present the computation of the transition time, from the probability obtained with AMS using an appropriate set of initial conditions.

2.2.1 The AMS algorithm

The three numerical parameters of the algorithm are: the reaction coordinate ξ, the total number of replicas N, and the minimum number k of replicas killed at each iteration. Let us denote by $\mathbf{X}_t^{n,q}$ the vector of positions and momenta at time t of the n^{th} replica ($1 \leq n \leq N$) at iteration q of the AMS algorithm. Let us now consider a set of initial conditions $(\mathbf{X}_0^{n,0})_{1 \leq n \leq N}$, which are i.i.d. random variables distributed according to a distribution μ_0 over $\mathbb{R}^{2d}$, supported outside but in a neighborhood of A. For all $n \in \{1, ..., N\}$ the path from $\mathbf{X}_0^{n,0}$ to either A or B is computed, creating the first set of replicas $(\mathbf{X}^{n,0}_{t \in [0, \tau^{n,0}_{AB}]})_{1 \leq n \leq N}$, where $\tau^{n,0}_{AB} = \min(\tau^{n,0}_A, \tau^{n,0}_B)$ with:

$$\tau_A^{n,0} = \inf\left\{ t \geq 0 : \mathbf{X}_t^{n,0} \in A \right\} \tag{2.4}$$

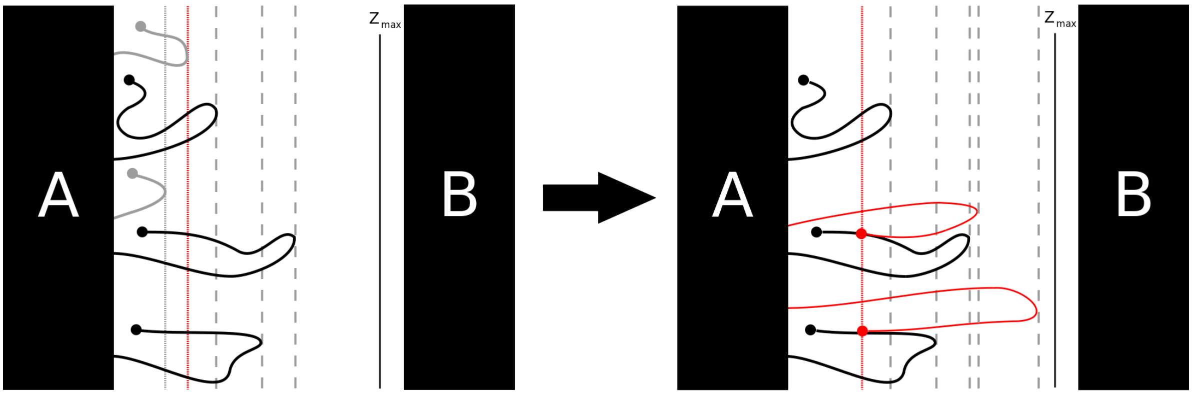


Figure 2.1 – First AMS iteration with $N = 5$ and $k = 2$. Both lower level replicas (in gray) are killed. Two of the remaining replicas are randomly selected to be copied up to level z^0_{kill} (dotted red line) and then continued until they reach A (typically more likely) or B.

and

$$\tau^{n,0}_B = \inf\left\{t \geq 0 : \mathbf{X}^{n,0}_t \in B\right\}. \tag{2.5}$$

So $\tau^{n,0}_{AB}$ is the first time that the n^{th} replica at iteration $q = 0$ enters A or B. In this initialization step, since the trajectories start in a neighborhood of A, they enter A before B with a probability very close to one. Notice that the replica $\mathbf{X}^{n,0}_{t\in[0,\tau^{n,0}_{AB}]}$ reaches B if and only if $\tau^{n,0}_B < \tau^{n,0}_A$. Let us denote by $(w_{n,0})_{1\leq n\leq N}$ the weight of each replica, that is initialized as $1/N$:

$$\forall \quad 1 \leq n \leq N, \quad w_{n,0} = \frac{1}{N}. \tag{2.6}$$

The algorithm then consists of iterating over $q \geq 0$ the three following steps:

1. Computation of the killing level.
   At the beginning of iteration q the set of replicas is $(\mathbf{X}^{n,q}_{t\in[0,\tau^{n,q}_{AB}]})_{1\leq n\leq N}$. Let us note by z^q_n the highest achieved value of the reaction coordinate by the n^{th} replica:

   $$z^q_n = \sup\{\xi(\mathbf{X}^{n,q}_t) : 0 \leq t \leq \tau^{n,q}_{AB}\}. \tag{2.7}$$

   This is called the level of the replica. To compute the killing level, the replicas are ordered according to their level. Hence, let us introduce the permutation $\alpha^q : [1,N] \to [1,N]$ of the trajectories' labels such that:

   $$z^q_{\alpha^q(1)} \leq z^q_{\alpha^q(2)} \leq \ldots \leq z^q_{\alpha^q(N)}. \tag{2.8}$$

   The killing level is defined as the kth order level, i.e. $z^q_{kill} = z^q_{\alpha^q(k)}$. If all the replicas have a level lower or equal to the killing level one sets $z^q_{kill} = +\infty$.

2. Stopping criterion.
   The algorithm stops at iteration q if $z^q_{kill} > z_{max}$. This happens if all the replicas reached the last level z_{max} or if $z^q_{kill} = +\infty$, a situation called extinction in the following. When the stopping criterion is satisfied, the algorithm is stopped and the current iteration index q is stored in a

variable called Q_{iter}. Notice that Q_{iter} may be null, since q starts from zero. The integer Q_{iter} is exactly the number of replication steps (see step 3 below) that have been performed when the algorithm stops.

3. Replication.
All the k^{q+1} replicas for which $z_n^q \leq z_{kill}^q$ are killed. Notice that $k^{q+1} \in \{k, k+1, ..., N-1\}$. Among the $N - k^{q+1}$ remaining replicas, k^{q+1} are uniformly chosen at random to be replicated. Replication consists in copying the replica up to the first time it goes beyond the level z_{kill}^q, so the last copied point has a level strictly larger than z_{kill}^q. From that point, the dynamics is run until A or B is reached. This will generate k^{q+1} new trajectories with level larger than z_{kill}^q. Once all the killed replicas have been replaced, the new set of replicas $(\mathbf{X}^{n,q+1}_{t\in[0,\tau_{AB}^{n,q+1}]})_{1\leq n\leq N}$ is defined. To complete iteration q one has to update the new weights by:
$$\forall \quad 1 \leq n \leq N, \quad w_{n,q+1} = \frac{N - k^{q+1}}{N} w_{n,q}. \tag{2.9}$$
From this, q is incremented by one and one comes back to the first step to start a new iteration.

Let us consider the set of all M replicas $\mathbf{X}^m_{t\in[0,\tau^m_{AB}]}$ generated during the algorithm run, including the killed ones, and call w_m their weight. The estimator of $\mathbb{E}(F(\mathbf{X}_{t\in[0,\tau_{AB}]}))$, for any path functional F is[22]

$$\sum_{m=1}^{M} w_m F(\mathbf{X}^m_{t\in[0,\tau^m_{AB}]}). \tag{2.10}$$

This will be used in Section 2.3.3 to compute the committor function over the phase space.

Note from the description of the algorithm that, at a giving iteration, all the living replicas have the same weight. The weight of a killed replica stops being updated after it is killed. Therefore, the replica weight depends on up to which iteration it has survived.

As previously mentioned, we will be particularly interested in the estimation of the probability $\mathbb{P}(\tau_B < \tau_A)$, which corresponds to the choice of the path functional $\mathbb{1}_{\tau_B<\tau_A}(\mathbf{X}_{t\in[0,\tau_{AB}]})$ in (2.10). This means that only the trajectories that survived until the end of the algorithm run will be taken into account. Therefore, using condition (2.2) and Equation (2.10):

$$p_{AMS} = \sum_{n=1}^{N} w_{n,Q_{iter}} \mathbb{1}_{\tau_B^{n,Q_{iter}} < \tau_A^{n,Q_{iter}}} \tag{2.11}$$

is an estimator of $\mathbb{P}(\tau_B < \tau_A)$. Here the weights are all equal. Using Equations (2.6) and (2.9), and denoting by r the number of replicas that reached B at the last iteration of the algorithm, p_{AMS} can be rewritten as

$$p_{AMS} = \frac{r}{N} \prod_{q=0}^{Q_{iter}-1} \left(\frac{N - k^{q+1}}{N} \right), \tag{2.12}$$

where by convention $\prod_{q=0}^{-1} = 1$. To gain intuition in this formula, notice that the term $\frac{N-k^{q+1}}{N}$ in Equation (2.12) is an estimation of the probability of reaching level z_{kill}^q, conditioned to the fact that level z_{kill}^{q-1}

has been reached, (where by convention $z_{kill}^{-1} = -\infty$). Also, as an example, if all the replicas in the initial set $(\mathbf{X}^{n,0}_{t\in[0,\tau^{n,0}_{AB}]})_{1\leq n\leq N}$ reached B, $r = N$ and thus $p_{AMS} = 1$. In case of extinction $r = 0$, because no replica reached B, and thus $p_{AMS} = 0$.

Note that the number k^{q+1} of killed replicas at iteration q may exceed k. The situation were $k^{q+1} > k$ happens if there is more than one replica with level equal to z_{kill}^{q}. There are typically two situations for which this occurs. First, this may happen if there exists a region where the reaction coordinate is constant. Second, it may be a consequence of the replication step at a previous iteration if the following occurs: (1) The point up to which the replica is copied has a ξ-value which is the maximum of the ξ-values along the trajectory (namely the level of the replica); (2) The replicated replica has the same level as the copied replica. Notice that this happens because the AMS method is applied to a discrete in time Markov process.

This algorithm is implemented in NAMD [10] as a Tcl script, easily used via the configuration file. The script is compatible with NAMD version 2.10 or higher[39]. In order to decrease the computational cost, the reaction coordinate of a point in the trajectory is only calculated every $K_{\text{AMS}} = \Delta t_{\text{AMS}}/\Delta t$ timesteps. This means that, in practice, the algorithm is actually applied to the subsampled Markov chain $(\mathbf{X}_{sK_{\text{AMS}}})_{s\in\mathbb{N}}$. It is indeed useless to consider the positions of the trajectory at each simulation time step, as no significant change occurs in a 1 or 2 fs time scale. Also notice that, along a trajectory, only the points that can possibly be used in future replication steps must be recorded, reducing memory use. This corresponds to points for which the reaction coordinate strictly increases.

2.2.2 Properties of the AMS method

Let us recall some important properties of the AMS method obtained in previous works. One of them is the unbiasedness of the algorithm. It can be proven[22] that the expected value of the probability estimator is equal to the probability to be calculated:

$$\mathbb{E}(p_{AMS}) = \mathbb{P}(\tau_B < \tau_A). \tag{2.13}$$

This is more generally true for the estimator (2.10):

$$\mathbb{E}\left(\sum_{m=1}^{M} w_m F(\mathbf{X}^{m}_{t\in[0,\tau^{m}_{AB}]})\right) = \mathbb{E}(F(\mathbf{X}_{t\in[0,\tau_{AB}]})). \tag{2.14}$$

Hence, in practice, the algorithm is run more than once and the result is obtained as an empirical average of the estimators for each run. This also provides naturally asymptotic confidence interval on the results, using the central limit theorem. Notice that unbiasedness holds whatever the choice of the reaction coordinate ξ, the number of replicas N and the minimum number of killed replicas k at each iteration. Therefore, one can compare the results obtained with different sets of parameters (in particular different reaction coordinates) to gain confidence in the result. These parameters however affect the variance of the estimator and, consequently, its efficiency.

Another paper[40] considers the ideal case, namely the situation where the reaction coordinate is the committor function. It can be proven that this is the best reaction coordinate in terms of the variance of p_{AMS}. Moreover, this case is interesting since explicit computations give some insights on the efficiency of the algorithm, that are observed to be useful beyond the ideal case. In the ideal case, variance and the efficiency of the method are then proportional to $1/N$. Let us recall that the efficiency of a Monte Carlo method can be defined as the inverse of the product of the computational cost and the variance[41]. Again in the ideal case, the number of iterations Q_{iter} is a random variable that follows a Poisson distribution with mean value $-N\log(\mathbb{P}(\tau_B < \tau_A))$. This indicates that the method is well suited to estimate small probabilities, hence appropriate to the simulation of rare events.

We concentrated here on the estimation of the probability $\mathbb{P}(\tau_B < \tau_A)$, but as explained above, see (2.10), other estimations can be made with this method[22]. It is possible, for example, to calculate unbiased estimators of $\mathbb{E}(F((\mathbf{X}_{t\in[0,\tau_{AB}]}))\mathbb{1}_{\tau_B<\tau_A})$ for any path functional F by simply making averages over the trajectories obtained at the end of the algorithm that reached B before A. Consequently, it is also possible to obtain estimators of conditional expectations $\mathbb{E}(F((\mathbf{X}_{t\in[0,\tau_{AB}]}))|\tau_B < \tau_A)$. Such estimators have a bias of order $1/N$ in the large N limit. This will be used in particular in Section 2.3 to compute the flux of reactive trajectories from A to B.

2.2.3 The transition time equation

Another quantity that we aim to obtain is the transition time from A to B, using the probability estimated by AMS. The transition time is the average time of the trajectories, coming from B, from its first entrance in A until the first entrance in B afterwards[16, 42]. As A is metastable, the dynamics makes in and out of A loops before visiting B. To correctly define those loops let us fix an intermediate value z_{min} of the reaction coordinate, defining an isolevel surface $\Sigma_{z_{min}}$:

$$\Sigma_{z_{min}} = \{\mathbf{X} \in \mathbb{R}^{2d} : \xi(\mathbf{X}) = z_{min}\}. \tag{2.15}$$

If A is metastable and $\Sigma_{z_{min}}$ is close to A the number of loops made between A and $\Sigma_{z_{min}}$ before visiting B is large. After some of them, the system reaches an equilibrium. When this equilibrium is reached the first hits of $\Sigma_{z_{min}}$ follow a so-called quasi-stationary distribution μ_{QSD}. Here, we call the first hitting points of $\Sigma_{z_{min}}$ the first points that, coming from A, have a ξ-value larger than z_{min}. If one then uses as a set of initial conditions the random variables $(\mathbf{X}_0^{n,0})_{1\leq n\leq N}$ distributed according to μ_{QSD}, it is possible to evaluate the probability p to reach B before A starting from $\Sigma_{z_{min}}$ at equilibrium by using AMS. As A is metastable, the number of loops needed to reach the equilibrium is small compared to the total number of loops made before going to B, so it can be neglected.

Let us now use these considerations to estimate the transition time from A to B. Consider an equilibrium trajectory coming from B that enters A and returns to B. The goal is to calculate the average time of this trajectory[16]. A good strategy is to split this path in two: the loops between A and $\Sigma_{z_{min}}$, and the reaction trajectory, i.e. the path from A to B that does not comes back to A after reaching $\Sigma_{z_{min}}$. This is outlined in Figure 2.2. We will call T_{AB} the time of one trajectory between the first hit of $\Sigma_{z_{min}}$ after reaching A and the first subsequent entry in B, neglecting the first time taken to go out of A, which is in practice very short. One can define as T_{loop}^k the time of the k^{th} loop between two subsequent hits of $\Sigma_{z_{min}}$, conditioned to have visited A between them, and as T_{reac} the time of the reaction trajectory. If

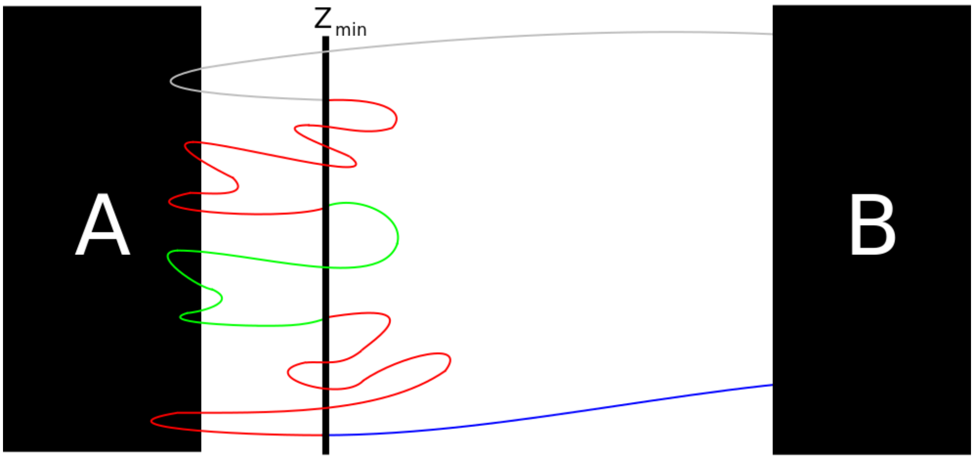


Figure 2.2 – The loops between A and $\Sigma_{z_{min}}$ (red and green), that corresponds to times T^1_{loop}, T^2_{loop} and T^3_{loop} (see (2.16), with $n = 3$); and the reaction trajectory (blue), that corresponds to T_{reac}. The time of the colorful trajectory is then T_{AB}.

the number of loops made before visiting B is n, the time T_{AB} can be obtained as:

$$T_{AB} = \sum_{k=1}^{n} T^k_{loop} + T_{reac}. \tag{2.16}$$

At each passage over $\Sigma_{z_{min}}$ there are two possible events, first enter A or first enter B. As mentioned in the previous paragraph, it is possible to obtain with AMS the probability p at equilibrium to visit B before A starting from the probability distribution μ_{QSD} on $\Sigma_{z_{min}}$. Therefore, the system enters B after $1/p$ passages over $\Sigma_{z_{min}}$, so the mean number of loops made before that is $1/p-1$. This leads us to the final equation for the expected value of T_{AB}:

$$\mathbb{E}(T_{AB}) \simeq \left(\frac{1}{p}-1\right)\mathbb{E}(T_{loop}) + \mathbb{E}(T_{reac}). \tag{2.17}$$

The mathematical formalization of this reasoning is a work in progress. The consistency of (2.17) has already been tested on various systems in previous works[11, 38]. In this paper, we numerically investigate the quality of formula (2.17) using the estimate of p obtained with AMS starting from μ_{QSD} (see Section 2.3.2). Note that the sampling of μ_{QSD} as well as $\mathbb{E}(T_{loop})$ can be obtained with short direct simulations while AMS is used to get both p and $\mathbb{E}(T_{reac})$. The first term in Equation (2.17) is much larger than the last one in the case of a rare event, making crucial the achievement of good probability estimations to obtain acceptable estimations for the transition time. Typically, the term $\mathbb{E}(T_{reac})$ is small compared to $\mathbb{E}(T_{AB})$ and can be ignored. In fact, forward flux sampling[18] approximates the reaction rate $k_{AB} = \mathbb{E}(T_{AB})^{-1}$ by $p/\mathbb{E}(T_{loop})$, which is consistent with our formula (2.17).

Choosing the parameter z_{min} may be delicate. The closer $\Sigma_{z_{min}}$ to A, the smaller the probability p to estimate. On the other hand, if $\Sigma_{z_{min}}$ is too far from A, there will be fewer loops, so the time to reach the quasi-stationary distribution will not be negligible. Moreover, the simulation time needed to obtain a good estimation of $\mathbb{E}(T_{loop})$ will be larger. This will again be discussed in the numerical example in the next section.

2.3 Results

We apply the AMS method to the $C_{eq} \to C_{ax}$ transition of the N-acetyl-N'-methylalanylamide, also known as alanine dipeptide or dialanine. The transition between its two stable conformations in gas phase occurs in a time scale of the order of a hundred nanoseconds, allowing us to obtain direct numerical simulation (DNS) estimations to compare to results obtained with AMS.

Both conformations can be characterized by two dihedral angles, φ and ψ (Figure 2.3). Regions A and B (C_{eq} and C_{ax}, respectively), are defined as ellipses that covers the two most significant wells on the free energy landscape (Figure 2.4).

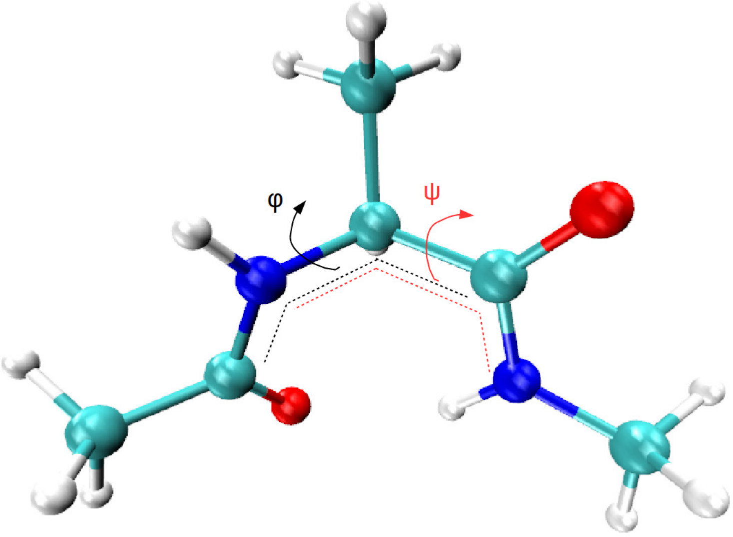


Figure 2.3 – The dihedral angles φ and ψ used to distinguish between the C_{eq} and C_{ax} conformations.

Two reaction coordinates are investigated. The first one (see (2.18)) is a continuous piecewise affine function of φ and the second one (see (2.19)) is a measure of the distance to the two regions A and B. Here are the precise definitions of ξ_1 and ξ_2 (see Figure 2.5 for a contour plot of ξ_2):

$$\xi_1(\varphi) = \begin{cases} -5.25 & \text{if } \varphi < -52.5 \\ 0.1\varphi & \text{if } -52.5 \le \varphi \le 45 \\ 4.5 & \text{if } 45 < \varphi < 92.5 \\ -0.122\varphi + 15.773 & \text{if } 92.5 \le \varphi \le 172.5 \\ -5.25 & \text{if } \varphi > 172.5 \end{cases} \tag{2.18}$$

$$\xi_2(\varphi,\psi) = \min(d_A, 6.4) - \min(d_B, 3.8) \tag{2.19}$$

In Equation (2.19), d_A (resp. d_B) is the sum of the Euclidean distances to the foci of the ellipse A (resp. B).

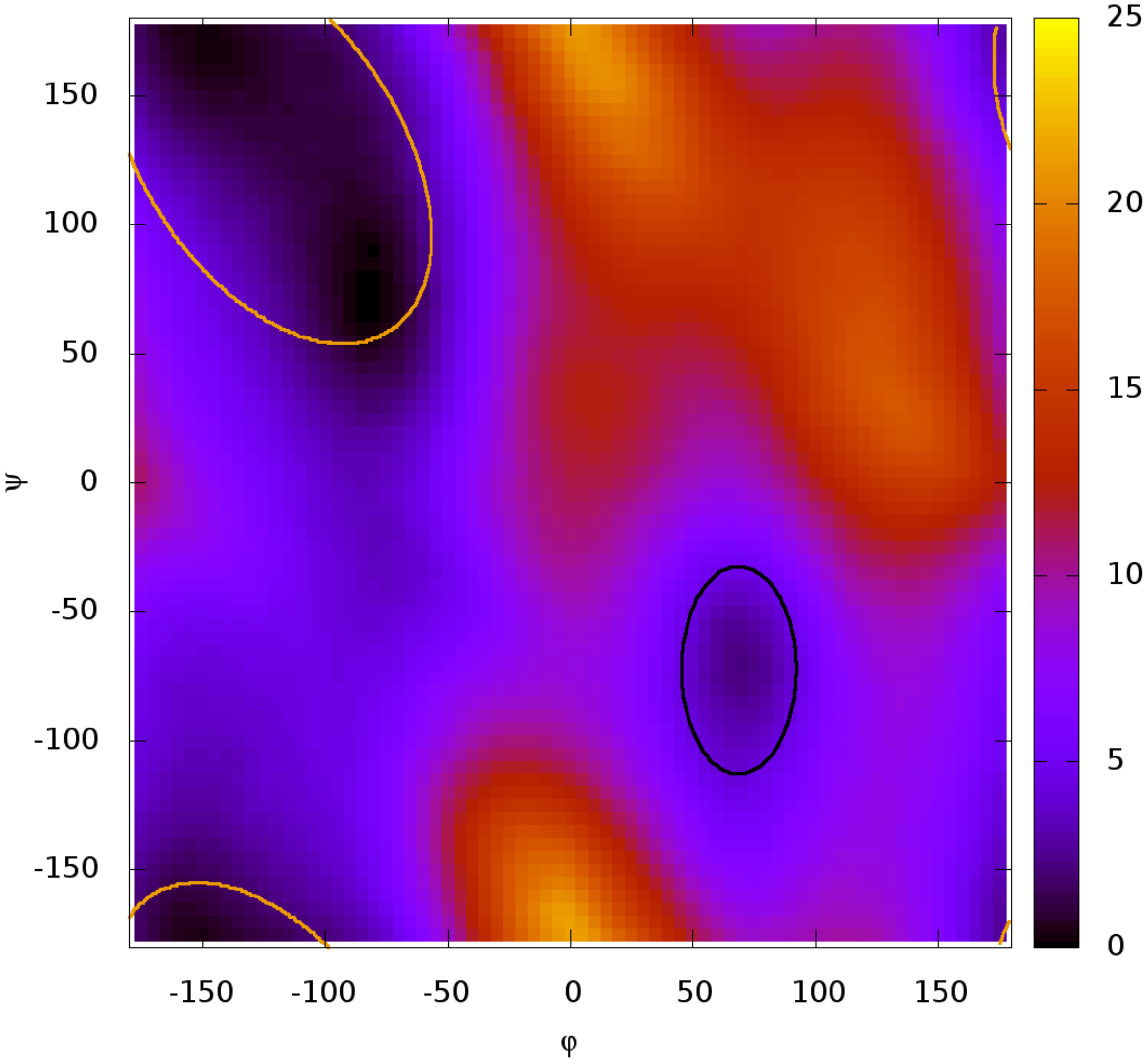


Figure 2.4 – The free energy landscape [14], used to define zones A (yellow) and B (black).

The values of z_{max} used for the simulations are 4.49 for ξ_1 and 4.9 for ξ_2. All the simulations are performed using NAMD[10] version 2.11 with the CHARMM27 force field.

To numerically illustrate some properties of the algorithm, we first calculate the transition probability starting from one fixed (deterministic) initial condition. These results are presented in Section 2.3.1, as well as the flux of reaction trajectories. The estimations of transition times are reported in Section 2.3.2, where a proper way to sample μ_{QSD} is proposed. Finally, we present in Section 2.3.3 a way to use AMS in order to compute an approximation of the committor function.

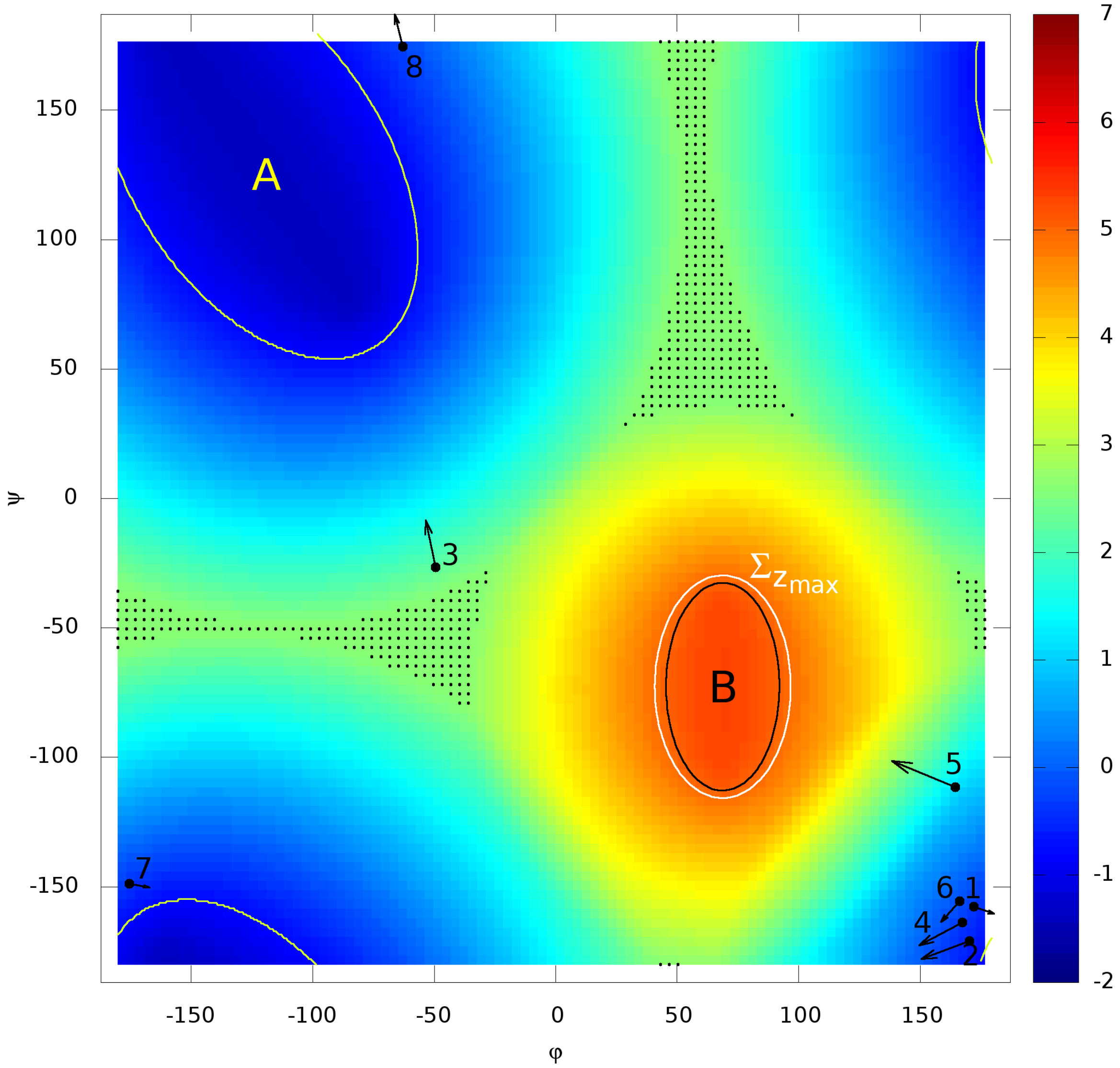


Figure 2.5 – Contour plot of the second reaction coordinate ξ_2. Regions A and B are marked in yellow and black, respectively. The region $\Sigma_{z_{max}}$ used for the AMS runs ($z_{max} = 4.9$) is marked in white. The zone covered with black dots corresponds to regions where ξ_2 is constant and equal to 2.6. The numbered vectors corresponds to the initial coordinates used for some of the simulations, whose results are presented below.

2.3.1 Calculating the Probability with AMS

To evaluate the efficiency of the algorithm to estimate the probability to visit B before A, we first initiate all the replicas from the same point $\mathbf{x}$ (fixed positions and velocities for all atoms), i.e. $\forall n \in [1, N]$, $\mathbf{X}_0^{n,0} = \mathbf{x}$. This enables us to compare estimates of the probability to enter B before A obtained with AMS with accurate values obtained using DNS. In DNS, simulations start from $\mathbf{x}$ and stop when A or B is reached. The ratio of the number of times B is reached over the total number of simulations is the DNS estimation for the probability $\mathbb{P}(\tau_B < \tau_A)$. Results (both for DNS and AMS) are reported in

Figure 2.6 for four different choices of **x** (points 1 to 4 in Figure 2.5).

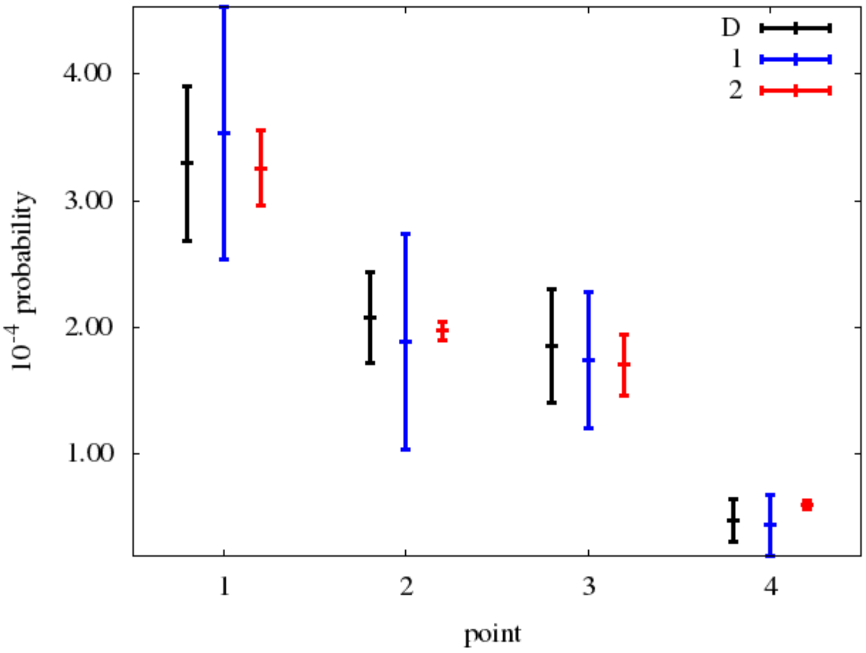


Figure 2.6 – Probability estimations using four different points as a initial conditions (see Figure 2.5): D is for DNS, 1 is for AMS using ξ_1 and 2 is for AMS using ξ_2. For each point we made about 200 AMS runs and a 15 ns DNS.

First note from Figure 2.6 the robustness of the AMS algorithm with respect to the choice of the reaction coordinate. The two reaction coordinates indeed give probability estimates in accordance with the direct simulation values. The second interesting feature is the change in the confidence interval, that tends to be smaller for ξ_2. This illustrates the fact that the average of the estimator is the same whatever the choice of ξ (see (2.13)), but the variance depends on ξ.

Notice from results in Figure 2.7 that different values of k and N yield consistent estimates of the probability. This is again a numerical illustration of (2.13). Notice that the variance scales as $1/N$, as already discussed in Section 2.2.2.

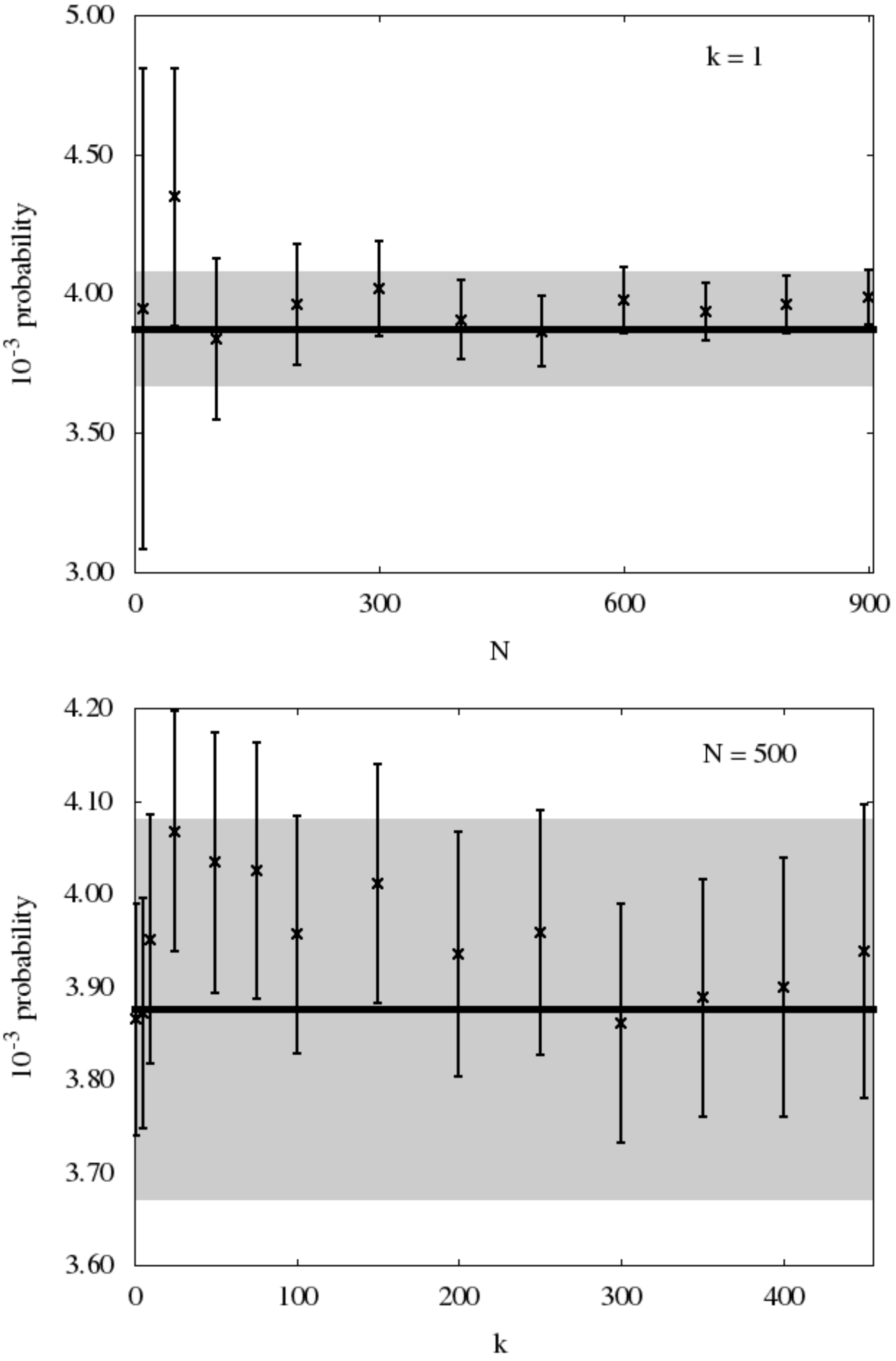


Figure 2.7 – AMS estimations for the probability with different values of k and N. Results were obtained using a fixed initial condition (point 1 in Figure 2.5) with ξ_2 and 1000 AMS runs for each value of N and k.

Concerning the reaction coordinate ξ_2, an interesting fact can be illustrated by looking at the number of killed replicas at each killing level (z^q_{kill}) over the AMS runs (Figure 2.8). The number of replicas

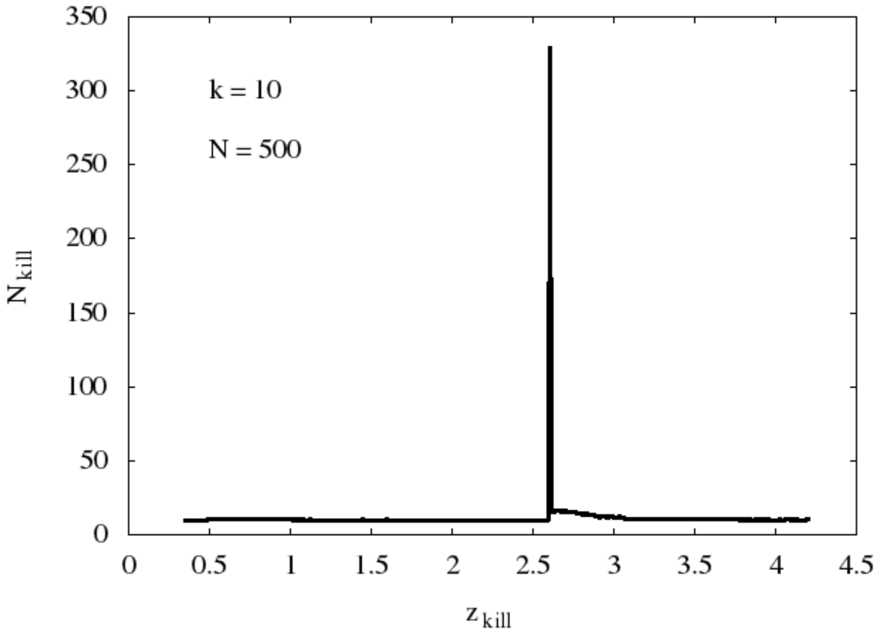


Figure 2.8 – Variation of the number of replicas killed as a function of the killing level. This graph was obtained with a mean over 1000 AMS runs.

remains close to k except for $\xi_2 = 2.6$, which is the value of the reaction coordinate in regions where it is constant (see Figure 2.5). This implies that a large number of replicas are at the same level when exploring these regions. So, at the stage where $z_{kill} = 2.6$, all replicas in this level are killed, which explains this result. This phenomenon increases the possibility of getting zero as an estimator of the probability, thus increases the variance. It is important to note that, even with such a locally constant reaction coordinate, ξ_2 exhibits good results with low variances, showing again that the AMS algorithm is robust in terms of the choice of the reaction coordinate.

To obtain information on the reaction paths and thus on the reaction mechanism, the flux of the reaction trajectories is evaluated by a numerical approximation of (inspired by[16] and Remark 1.13[42]):

$$J(\mathbf{x}) = \mathbb{E}^{\nu}\left(\mathbb{1}_{\tau_B<\tau_A}\frac{1}{\tau_B}\int_0^{\tau_B}\dot{\mathbf{q}}(t)\delta(\mathbf{x}-\mathbf{q}(t))dt\right), \tag{2.20}$$

where $\mathbf{q}(t)$ is the vector of positions at time t and ν is the distribution of initial points $\mathbf{X}(0)$, supported in a neighborhood of A. For the system at equilibrium, the distribution ν can be approximated by the distribution μ_{QSD}, introduced in Section 2.2.3. For other purposes, one can also consider a Dirac, *i.e.* $\mathbf{X}(0) = \mathbf{X}_0$.

To approximate this equation the (φ, ψ) space is split into L cells and the flux $J(C_l)$ is defined over each cell $(C_l)_{1\leq l\leq L}$. Using a set $\{(\mathbf{X}_t^1)_{t\in[0,\tau_B^1]}, ..., (\mathbf{X}_t^n)_{t\in[0,\tau_B^n]}\}$ of reaction trajectories obtained with the AMS method, each trajectory i has a weight of w_i and can be associated with a vector $(\boldsymbol{\theta}_t^i)_{t\in[0,\tau_B^i]}$, where $(\boldsymbol{\theta}_t^i)$=$(\varphi(\mathbf{X}_t^i), \psi(\mathbf{X}_t^i))$ are the two dihedral angles (see Figure 2.3). Equation (2.20) can then be

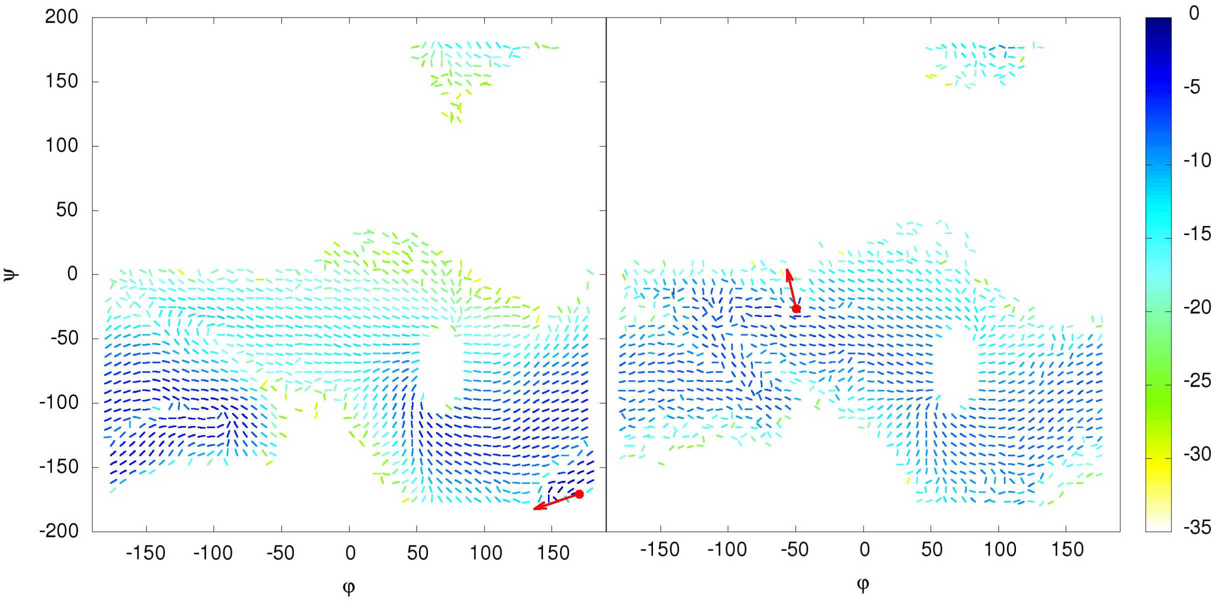


Figure 2.9 – The fluxes of the reaction trajectories starting from points 2 (left) and 3 (right) (see Figure 2.5). Initial conditions are represented by the red vectors. The colors represent minus the log of the norm of the flux (in fs^{-1}). The fluxes are averages over 500 000 trajectories, obtained by 1000 independent AMS runs with 500 replicas each.

approximated by:

$$J(C_l) = \frac{\sum_{i=1}^{n} w_i \sum_{t=0}^{\tau_B^i - 1} \left(\boldsymbol{\theta}_{t+1}^i - \boldsymbol{\theta}_t^i\right) \mathbb{1}_{\boldsymbol{\theta}_t^i \in C_l}}{\sum_{i=1}^{n} w_i \tau_B^i}. \tag{2.21}$$

There is a qualitative interest in calculating the flux for different distributions ν, *i.e.* different sets of initial conditions. Such a result is useful to visualize the transition paths from A to B. These paths highly depend on the initial condition as can be seen by comparing the two results in Figure 2.9, where ν is a Dirac over two different points.

We also look at the efficiency of the method by applying it to eight initial conditions. As mentioned in Section 2.2.2, the efficiency of a Monte Carlo method is defined as the inverse of the product of the computational cost and the variance[41]. In Figure 2.10 the variation of the ratio of the AMS efficiency over the DNS efficiency as a function of the probability $\mathbb{P}(\tau_B < \tau_A)$ is showed. When this ratio is larger than 1, the AMS algorithm is more efficient than DNS. Notice that all the points show that AMS is more efficient than DNS but also that this efficiency tends to be larger when the probability decreases. This illustrates that the method is particularly well suited to calculate small probabilities. As an example, for the point with probability 10^{-7} the wall clock time for DNS is over a week, but the estimation with 1000 AMS run in parallel with 32 cores takes less than two days.

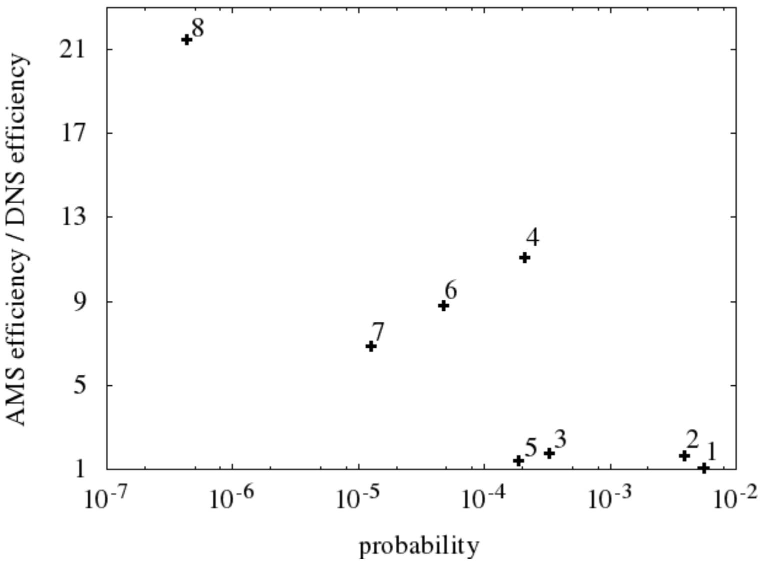


Figure 2.10 – Efficiency ratio between AMS and DNS estimations for points 1 to 8 (see Figure 2.5). The confidence intervals are too small to be seen on the graph.

2.3.2 Calculating the transition time

To evaluate the transition time using Equation (2.17) one needs estimations of p, $\mathbb{E}(T_{reac})$ and $\mathbb{E}(T_{loop})$. The last is easily obtained by a short simulation starting from A. The other two terms can be estimated using AMS, as long as the initial condition's points follow the distribution μ_{QSD}, as mentioned in Section 2.2.3. To obtain a reference value for the transition time, which is (309.5 ± 23.8) ns, a set of 97 direct simulations of 2μs each is made.

At first, we make a 2μs simulation, sufficiently long to observe transitions from A to B and thus to obtain DNS estimates for p and $\mathbb{E}(T_{reac})$. For the probability p we count the number of $\Sigma_{z_{min}} \to A$ and $\Sigma_{z_{min}} \to B$ trajectories, respectively n_A and n_B, yielding the estimate $p_{DNS} = n_B/(n_A + n_B)$. To investigate the consistency of Equation (2.17), we also calculate the transition time with these DNS values.

Using the same 2μs simulation, and for a fixed value of z_{min}, all the first hitting points of $\Sigma_{z_{min}}$ in the successive loops between A and $\Sigma_{z_{min}}$ are stored and 500 among them are randomly chosen to form the initial conditions' set to run the AMS simulations. This gives the samples distributed according to μ_{QSD}. In this process, estimates of $\mathbb{E}(T_{loop})$ are also obtained. To fix z_{min} we choose to use levels of ξ_2 and in total seven different values were adopted. The obtained results are reported in Figure 2.11.

Notice from Figure 2.11 (bottom) that the transition times obtained with the DNS estimates are consistent with the reference value. In fact, they only differ by 2 ps one from each other. This validates the use of Equation (2.17).

For the results obtained with AMS, first observe from Figure 2.11 (top) the consistency of the probability estimates obtained with the two different reaction coordinates. For some values of z_{min}, these estimations are not consistent with the DNS ones. Accordingly, for those values of z_{min}, the obtained transition times are also not compatible with the reference value, see 2.11 (bottom).

In order to understand the non consistency between the AMS and the DNS results, we look at the

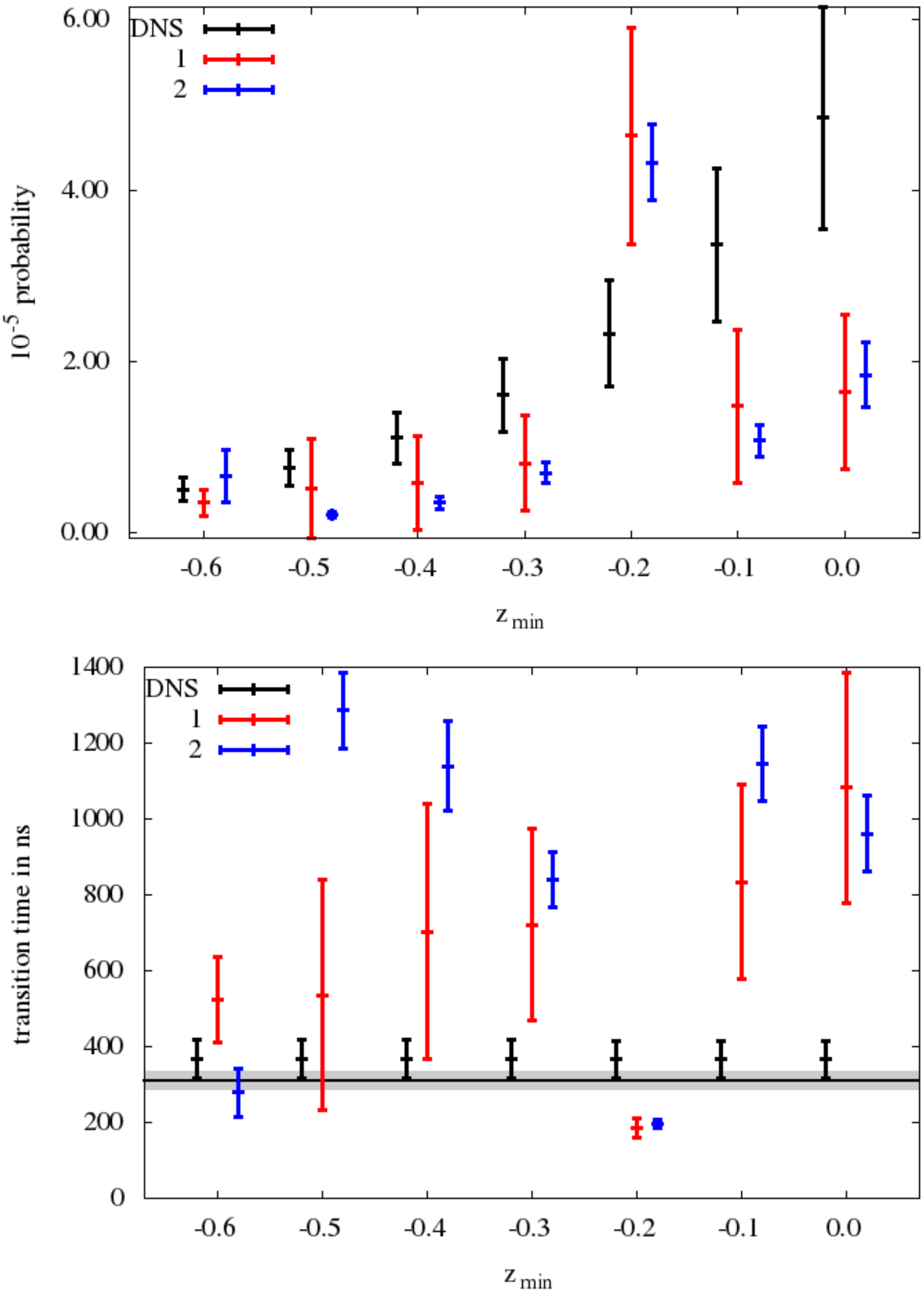


Figure 2.11 – Probability and transition time obtained for the seven sets of initial conditions with DNS and AMS with both ξ_1 (1) and ξ_2 (2). The DNS estimations were made using a 2μs simulation and the AMS with 1000 independent runs. In the bottom figure the reference value is represented as the gray interval.

sampling of the initial conditions. Recall that for AMS, an ensemble of 500 samples is chosen and fixed for all the AMS runs, while for DNS, these are actually sampled along the long trajectory. Moreover, we observe that the probability to reach B before A highly depends on the initial condition in the sample distributed according to μ_{QSD}. This yields a result which is not robust with respect to the choice of the 500 initial conditions and raises question about how to efficiently sample μ_{QSD}. The strategy we propose is, instead of fixing 500 initial conditions once for all, redraw new ones for each AMS run. This is made with a small initial simulation previously to each run, where, starting from A, the first 500 $\Sigma_{z_{min}} \to A$ trajectories are used as the first set of replicas (see Figure 2.12). This fixes the 500 initial conditions for each run. Notice that these simulations can also be used to obtain $\mathbb{E}(T_{loop})$, excluding the need to make the initial 2μs simulation previously mentioned.

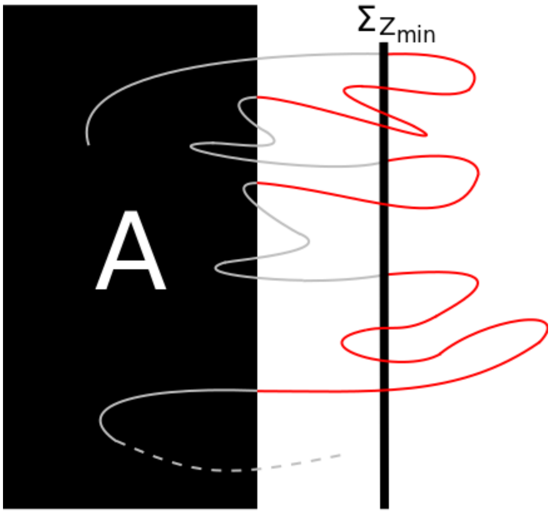


Figure 2.12 – The sample of the first 3 initial replicas (in red). The simulation is made until all the 500 replicas are obtained and this process is repeated before each AMS run.

The results using this new strategy are reported in Figure 2.13. The estimations for the probability, in Figure 2.13 (top), are in agreement with DNS. Nevertheless, observe that the larger z_{min}, i.e. the far from A, the more distant the estimator is from the reference value, and also the larger the variance. This is because the more far from A the more difficult it is to sample the distribution μ_{QSD}. Notice that the calculation of the transition time has a term in $1/p$ (see Equation (2.17)). Consequently, small errors in the probability causes large errors in the transition time. This can be observed in Figure 2.13 (bottom), where the best estimator is for the smaller value of z_{min}. Also notice that the results obtained for the transition time are in better agreement with the reference value than the previous one. We therefore conclude from this numerical experiment that it is worth redrawing new initial conditions for each AMS simulation in order to better sample the distribution μ_{QSD}.

Another important feature to be considered when fixing z_{min} is the time required to initiate the replicas and to run the AMS simulations. This is shown in Figure 2.14. The time for the initiation phase tends to grow exponentially as z_{min} is larger. However, because the AMS method is appropriate to simulate rare events, the AMS simulation time is approximately constant. Thus, we conclude it is better to have $\Sigma_{z_{min}}$ closer to A.

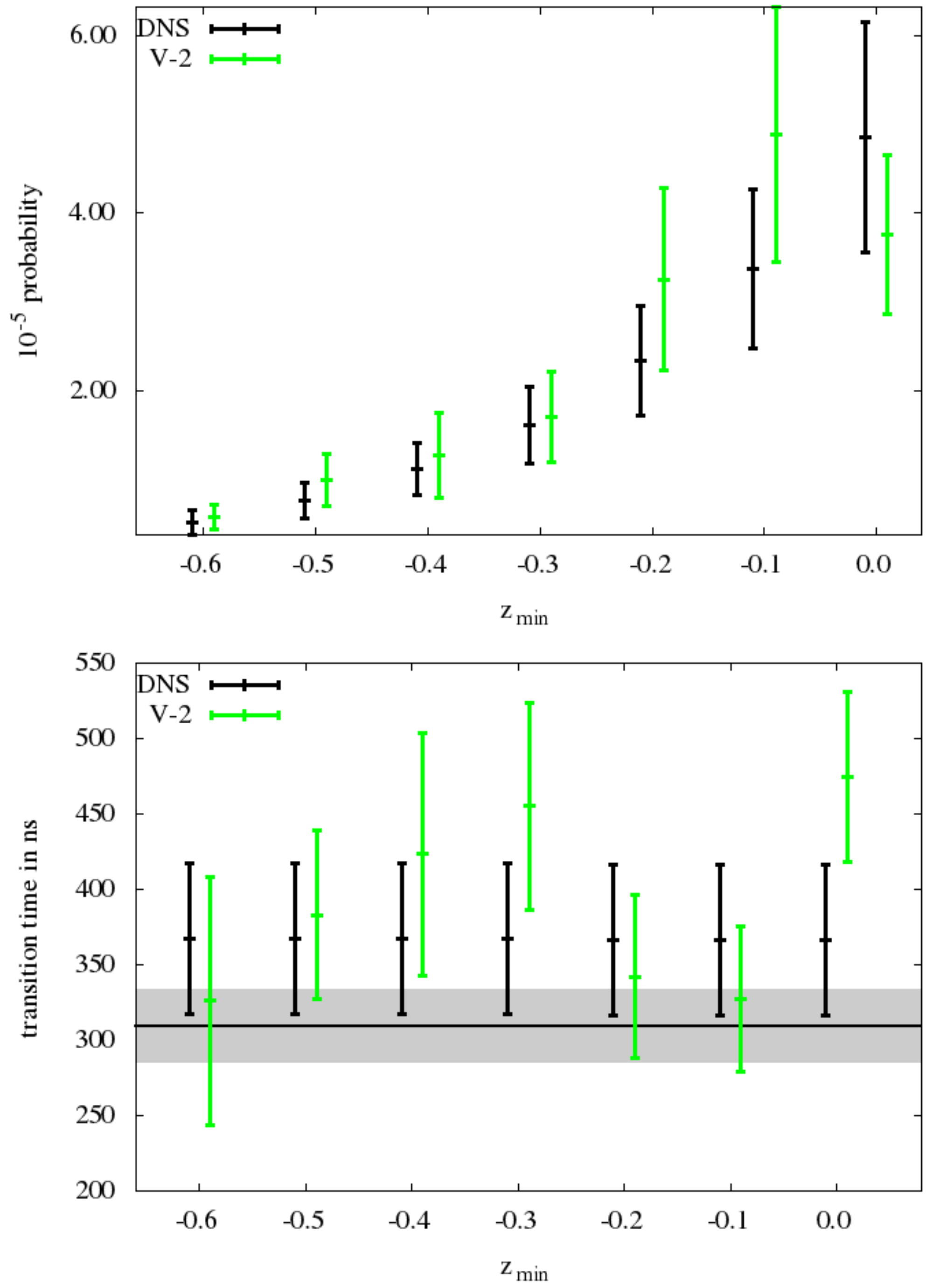


Figure 2.13 – Probability obtained varying the set of initial conditions before each AMS run with ξ_2 and the transition time calculated with them. For each value of z_{min} 1000 AMS runs were made with 500 replicas each.

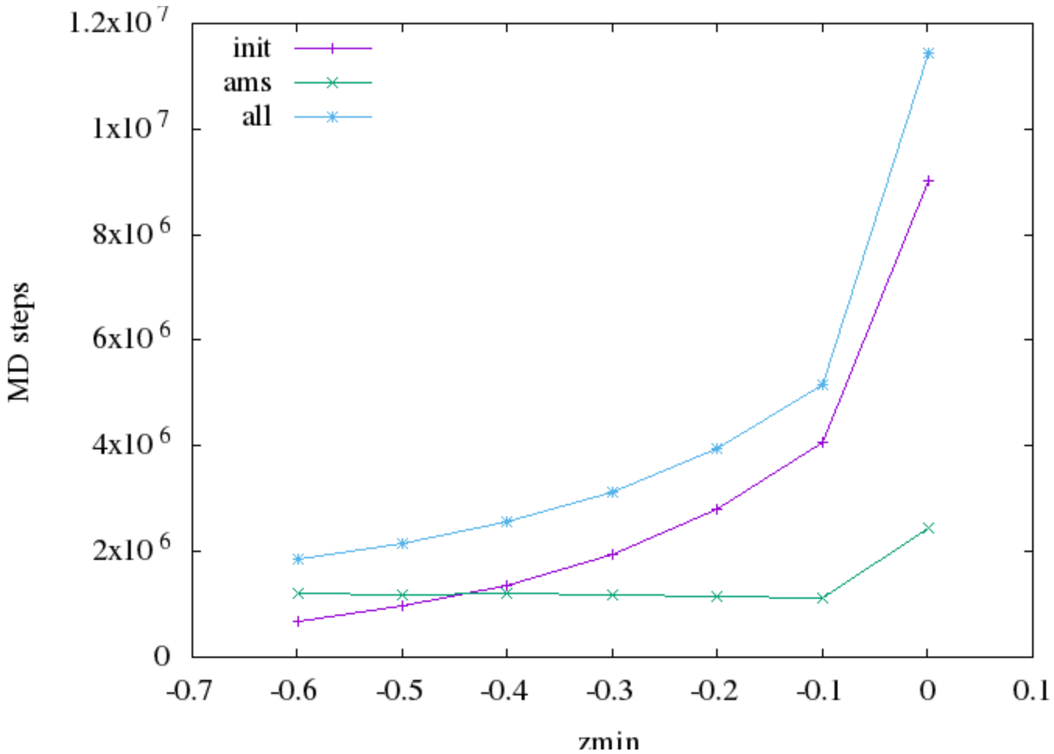


Figure 2.14 – Simulation steps used to initiate the 500 replicas and for each AMS run.

2.3.3 Calculating the committor function

Another quantity of interest is the committor function:

$$p(x) = \mathbb{P}(\tau_B < \tau_A | X_0 = x), \tag{2.22}$$

i.e. the probability of entering A before B when starting from x. Note that, from the definition of a conditional probability, it is possible to rewrite $p(x)$ as:

$$p(x) = \frac{p_{B,X_0}(x)}{p_{X_0}(x)} = \frac{\mathbb{P}(\tau_B < \tau_A \cap X_0 = x)}{\mathbb{P}(X_0 = x)}. \tag{2.23}$$

To approximate the committor function let us consider a large set of N trajectories $(\mathbf{X}_{t\in[0,\tau^n_{AB}]})_{1\leq n\leq N}$ at equilibrium that starts outside A and B. Using the same strategy as for the flux, the space is split into L cells $(C_l)_{1\leq l\leq L}$. Let us now introduce an approximation of the numerator $p_{B,X_0}(x)$ and the denominator $p_{X_0}(x)$ in Equation (2.23), for each cell C_l:

$$p_{B,X_0}(C_l) = \frac{\sum\limits_{n=1}^{N} \mathbb{1}_{\tau^n_B<\tau^n_A} \sum\limits_{t=0}^{\tau^n_{AB}} \mathbb{1}_{X^n_t\in C_l}}{\sum\limits_{n=1}^{N} (\tau^n_{AB}+1)}, \tag{2.24}$$

$$p_{X_0}(C_l) = \frac{\sum\limits_{n=1}^{N} \sum\limits_{t=0}^{\tau^n_{AB}} \mathbb{1}_{X^n_t\in C_l}}{\sum\limits_{n=1}^{N} (\tau^n_{AB}+1)}. \tag{2.25}$$

Note that this consists in counting each time a trajectory passes through C_l for $p_{X_0}(C_l)$ and considering it in $p_{B,X_0}(C_l)$ only if the trajectory enters B before A. Since we consider trajectories at equilibrium, $p_{B,X_0}(C_l)$ (resp. $p_{X_0}(C_l)$) actually approximates the probability to reach B before A and to be in C_l (resp. the probability to be in C_l) for a trajectory starting at equilibrium in C_l.

Let us now consider M AMS runs, where a total of N_m replicas $\mathbf{X}^{n,m}_{t\in[0,\tau^{n,m}_{AB}]}$ where obtained for each run m, and call $w_{n,m}$ the weight of n^{th} replica from the m^{th} run. From Equation (2.10), the following approximations for Equations (2.24) and (2.25) are obtained:

$$\tilde{p}_{B,X_0}(C_l) = \frac{\sum\limits_{m=1}^{M} \sum\limits_{n=1}^{N_m} w_{n,m} \mathbb{1}_{\tau^{n,m}_B<\tau^{n,m}_A} \sum\limits_{t=0}^{\tau^{n,m}_{AB}} \mathbb{1}_{X^{n,m}_t\in C_l}}{\sum\limits_{m=1}^{M} \sum\limits_{n=1}^{N_m} w_{n,m}(\tau^{n,m}_{AB}+1)} \tag{2.26}$$

$$\tilde{p}_{X_0}(C_l) = \frac{\sum\limits_{m=1}^{M}\sum\limits_{n=1}^{N_m} w_{n,m} \sum\limits_{t=0}^{\tau_{AB}^{n,m}} \mathbb{1}_{X_t^{n,m}\in C_l}}{\sum\limits_{m=1}^{M}\sum\limits_{n=1}^{N_m} w_{n,m}(\tau_{AB}^{n,m}+1)} \tag{2.27}$$

The division of (2.26) by (2.27) gives us an estimation $\tilde{p}(C_l)$ of the committor function in cell C_l:

$$\tilde{p}(C_l) = \frac{\sum\limits_{m=1}^{M}\sum\limits_{n=1}^{N_m} w_{n,m} \mathbb{1}_{\tau_B^{n,m}<\tau_A^{n,m}} \sum\limits_{t=0}^{\tau_{AB}^{n,m}} \mathbb{1}_{X_t^{n,m}\in C_l}}{\sum\limits_{m=1}^{M}\sum\limits_{n=1}^{N_m} w_{n,m} \sum\limits_{t=0}^{\tau_{AB}^{n,m}} \mathbb{1}_{X_t^{n,m}\in C_l}}. \tag{2.28}$$

The result obtained using Equation (2.28) is given in Figure 2.15.

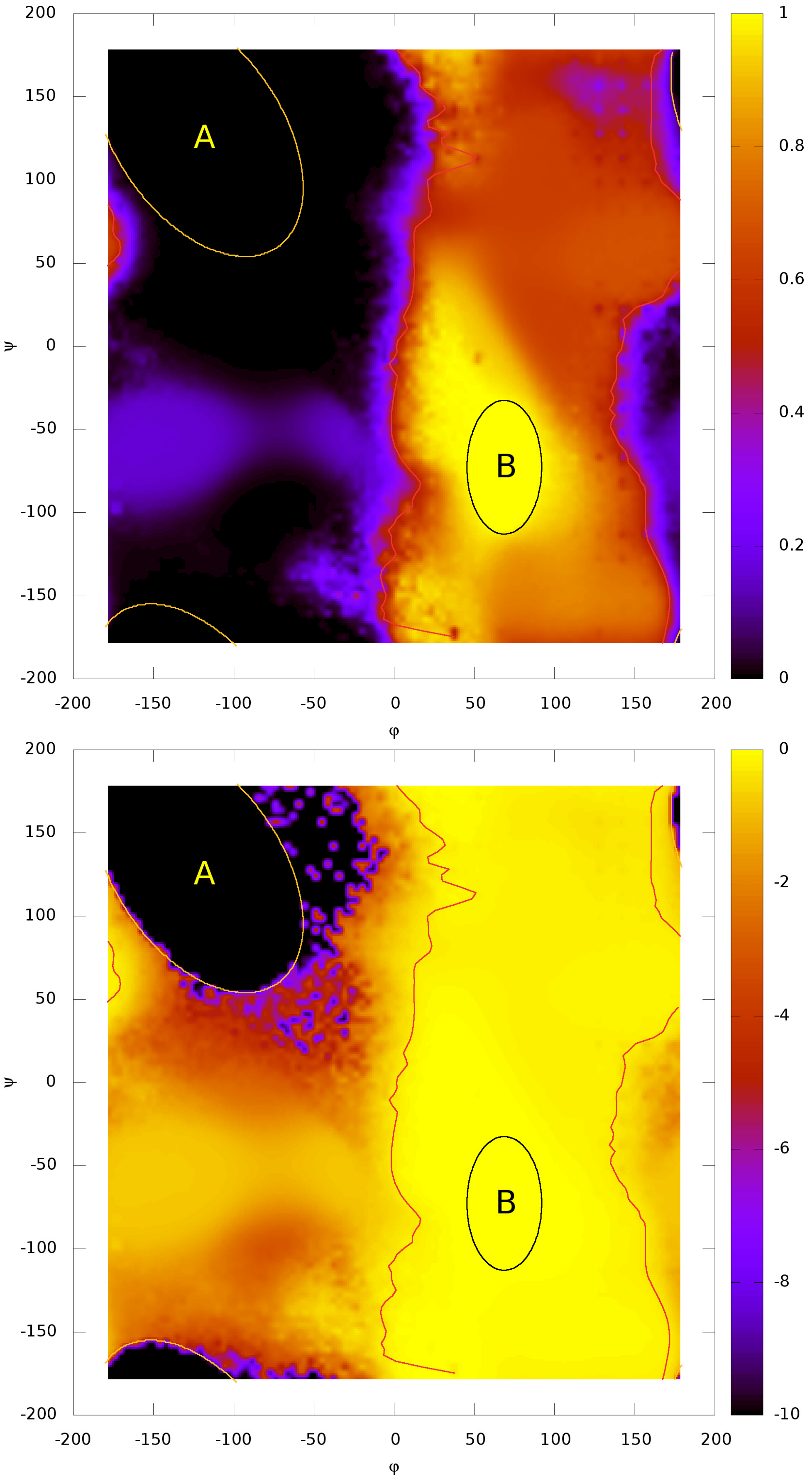


Figure 2.15 – The committor function obtained with 5000 AMS runs with 100 replicas each. In the second figure the same result is presented in log-scale, with a cut at 10^{-10}. We used initial conditions at equilibrium, starting from equally distributed (φ, ψ) positions over the Ramachandran plot. The red lines mark the isolevel 0.5, where the probability to enter A before B is the same as to enter B before A, namely the transition state.

ACKNOWLEDGMENTS

The authors would like to thank Najah-Imane Bentabet who worked on a preliminary version of the AMS algorithm for the NAMD code, and Jérôme Hénin for fruitful discussions. Part of this work was completed while the authors were visiting IPAM during the program "Complex High-Dimensional Energy Landscapes". The authors would like to thank IPAM for its hospitality. This work is supported by the European Research Council under the European Union's Seventh Framework Programme (FP/2007-2013)/ERC Grant Agreement number 614492.

Chapter 3

Combining AMS and importance sampling for simulating equilibrium transition events

3.1 Introduction

Rare events are present in several fields, and one of the most important quantities of interest is the typical time for such events to occur. When considering transitions between metastable states, this quantity is called the transition time, and its inverse, the transition rate. For example, drug-target dissociation rates at equilibrium can be directly related to drug efficiency[32], making its calculation an essential step in drug design.

Rare events are hard to simulate as a result of their low probability of occurrence. The naive Monte Carlo approach is typically inefficient because of its prohibitive computational cost. To surpass this issue, a range of methods have been developed over the last decades, using different strategies to accelerate the sampling.

The adaptive multilevel splitting (AMS)[21] is a recent rare event method, developed less than 15 years ago. Its strategy is to split the event of interest into a sequence of conditional events, easier to simulate. This is done on the fly through a reaction coordinate, given by the user. Compared to other methods, AMS have a low quantity of user defined parameters, and is thus more robust and easy to use.

Like any other splitting method, such as forward flux sampling, e.g., AMS gives an estimator for the probability of occurrence, when starting from a set of initial conditions. This probability can then be used to compute an estimation of the transition time at equilibrium, but only if the initial condition's set represent the equilibrium, in a sense to be made precise. Two problems are encountered in the sampling of the initial points. The first is the choice of the distribution of initial conditions to get the transition time. The second is the sampling of this distribution: it appears that the samples which contribute the most to the rare event probability estimator are typically not the most likely ones. This implies a large variance of the estimator.

The objective of this chapter is to propose a new importance sampling strategy for the sampling of the initial conditions. The method is validated on a one dimensional toy case, already studied in [1], where the interest is to obtain the transition time between two metastable states. Despite the apparent simplicity of the problem, the sampling of initial conditions for AMS is computationally expensive. The goal is then to explain and propose remedies to the issues raised in [1] to compute the transition time using splitting methods. We propose an adaptive importance sampling technique to sample those points both correctly and efficiently, and discuss its application to multidimensional cases.

3.2 Algorithms

In this section we present the studied system, as well as the algorithms used to obtain the transition time. Section 3.2.1 gives the one dimensional potential, as well as the dynamics used, and section 3.2.2 provides the precise definition of the transition we aim to sample. Then, the equation used to calculate the transition time via the probability of transition is derived in section 3.2.3. At last, the AMS algorithm for the 1D case is presented in section 3.2.4. The material presented in this section relies on first [1], where a similar one-dimensional example was studied, and second [17], where the mathematical foundations of the formula presented below to calculate the transition time are given.

3.2.1 Langevin dynamics over the 1D potential

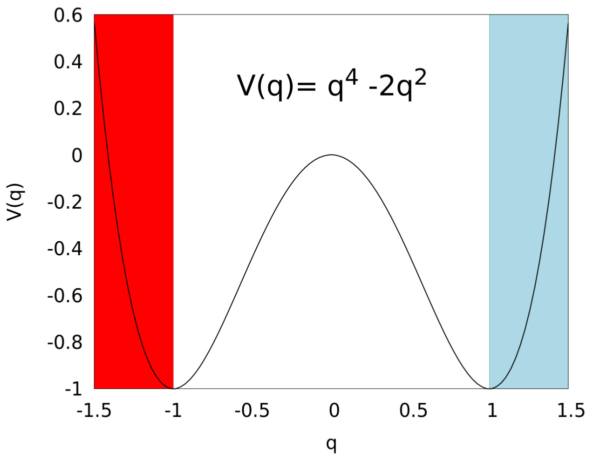


Figure 3.1 – Definitions of regions A (in red) and B (in blue) for the potential $V(q)$. The goal is to simulate transitions from the red to the blue region, and compute its mean duration.

The one dimensional potential is $V(q) = q^4 - 2q^2$, which has two minima, at $q = \pm 1$ (see figure 3.1). Let us denote by A and B the states defined by the intervals $]-\infty, -1[$ and $]1, +\infty[$, respectively. The interest here is to simulate the transition between A and B for the Langevin dynamics at equilibrium, and more specifically to obtain its average time at equilibrium. This transition is rare for this dynamics, as both wells represents regions where the system stays trapped for a long time. Therefore, a brute force Monte Carlo method is not efficient.

The Langevin dynamics is numerically solved using the midpoint Euler-Verlet-midpoint Euler scheme[43]. Let us call (q_n, p_n) the position and momentum at time $t_n = n\Delta t$. The scheme then reads:

$$\left\{\begin{array}{rcl} p_{n+1/4} & = & p_n - \dfrac{\Delta t}{4}\dfrac{\gamma}{m}\left(p_n + p_{n+1/4}\right) + G^n\sqrt{\gamma k_B T \Delta t} \\ p_{n+1/2} & = & p_{n+1/4} - \dfrac{\Delta t}{2}\nabla V\left(q_n\right) \\ q_{n+1} & = & q_n + \dfrac{\Delta t}{m} p_{n+1/2} \\ p_{n+3/4} & = & p_{n+1/2} - \dfrac{\Delta t}{2}\nabla V\left(q_{n+1}\right) \\ p_{n+1} & = & p_{n+3/4} - \dfrac{\Delta t}{4}\dfrac{\gamma}{m}\left(p_{n+1} + p_{n+3/4}\right) + G^{n+1/2}\sqrt{\gamma k_B T \Delta t} \end{array}\right. \tag{3.1}$$

Here $(G^n, G^{n+1/2})_{n\geq 0}$ are i.i.d. centered and normalized Gaussian random variables. The friction parameter γ is 0.3, the temperature T is 0.07 and the timestep Δt is 0.002. The mass m and the Boltzmann constant k_B are set to unity.

3.2.2 Definition of the transition time

The objective of this section is to precisely define the quantity of interest, namely the transition time. This requires the introduction of a few additional notations.

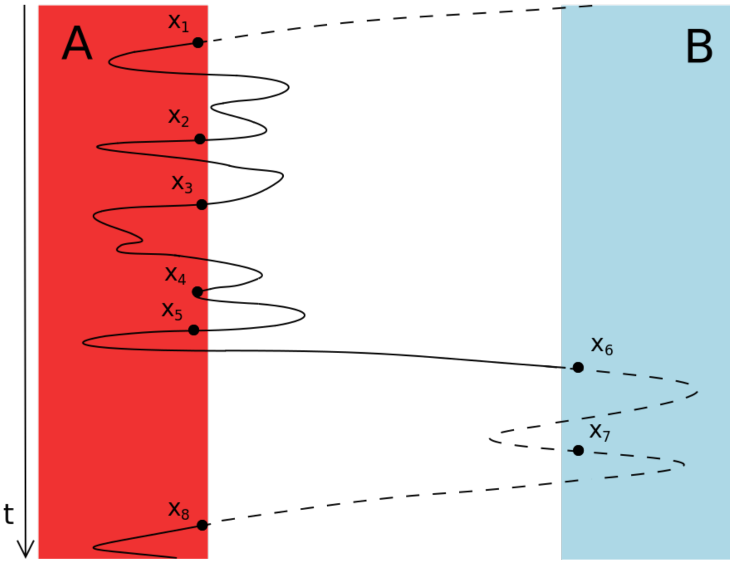


Figure 3.2 – Definition of the process x_n, used to define the distributions. The trajectory represents an interpolation of the discrete solution to the Langevin dynamics (3.1). Notice that, the points x_n, defined in (3.2), are in $A \cup B$.

Let us call $x_n = (q_{\tau_n}, p_{\tau_n})$ the position and momentum of the particle at time τ_n, where (τ_n) represents the successive entrance times in A or B, defined as follows:

$$x_n = (q_{\tau_n}, p_{\tau_n}), \text{ where } \quad \tau_n = \min\left\{m > \tau_{n-1} \,\middle|\, q_m \in A \cup B, q_{m-1} \notin A \cup B\right\}. \tag{3.2}$$

This means that x_n are the successive entrance points in A or B (see figure 3.2). In the following we will abuse notation and denote A for $A \times \mathbb{R}$ and B for $B \times \mathbb{R}$, so that: $x_n \in A \Leftrightarrow q_{\tau_n} \in A$ and $x_n \in B \Leftrightarrow q_{\tau_n} \in B$. Let us now introduce T_k^A (resp. T_k^B), the first time the particle enters A (resp. B) coming from B (resp. A): $\forall k \geq 1$

$$T_k^A = \min\left\{ n > T_{k-1}^B \middle| \, x_n \in A \right\}$$
$$T_k^B = \min\left\{ n > T_k^A \middle| \, x_n \in B \right\},$$

with the convention $T_0^B = -\infty$. The first entrance equilibrium distribution in A is defined as:

$$\nu_E = \lim_{M \to \infty} \frac{1}{M} \sum_{k=1}^{M} \delta_{x_{T_k^A}}.$$

The goal is to calculate the average transition time for the metastable transition from A to B, at equilibrium. Equivalently, we want to obtain the expected value of $T_{AB} = T_1^B - T_1^A$ over ν_E[16, 42]:

$$\mathbb{E}^{\nu_E}(T_{AB}) = \lim_{M \to \infty} \frac{1}{M} \sum_{k=1}^{M} \left(T_k^B - T_k^A \right).$$

This requires two ingredients: an appropriate rewriting of the average transition time, presented in Section 3.2.3, and a rare event sampling algorithm, presented in Section 3.2.4.

3.2.3 Computing the transition time

From the process (x_n), one can also define a distribution for the successive entrance points in A at equilibrium, as:

$$\mu_A = \lim_{N \to \infty} \frac{\sum_{n=1}^{N} \delta_{x_n} \mathbb{1}_{x_n \in A}}{\sum_{n=1}^{N} \mathbb{1}_{x_n \in A}}.$$

Let us now consider the first entrances in A and B, defined as follows:

$$\begin{array}{rcl} \tau_A & = & \min\left\{ n > 0 | q_n \in A, q_{n-1} \notin A \right\} \\ \tau_B & = & \min\left\{ n > 0 | q_n \in B, q_{n-1} \notin B \right\}. \end{array} \tag{3.3}$$

The probability for a particle to first enter B rather than A after exiting $A \cup B$ starting from μ_A, is then $\mathbb{P}^{\mu_A}(\tau_B < \tau_A)$. Let us denote by Δ the time between two subsequents entrances in $A \bigcup B$ starting from μ_A. One can then show the following so-called Hill relation[17, 44, 45]:

$$\mathbb{E}^{\nu_E}(T_{AB}) = \frac{\mathbb{E}^{\mu_A}(\Delta)}{\mathbb{P}^{\mu_A}(\tau_B < \tau_A)}. \tag{3.4}$$

Notice that:

$$\begin{array}{rcl} \mathbb{E}^{\nu_E}(T_{AB}) & = & \dfrac{\mathbb{E}^{\mu_A}(\Delta | \tau_B > \tau_A)\mathbb{P}^{\mu_A}(\tau_B > \tau_A) + \mathbb{E}^{\mu_A}(\Delta | \tau_B < \tau_A)\mathbb{P}^{\mu_A}(\tau_B < \tau_A)}{\mathbb{P}^{\mu_A}(\tau_B < \tau_A)} \\ & = & \left(\dfrac{1}{\mathbb{P}^{\mu_A}(\tau_B < \tau_A)} - 1 \right) \mathbb{E}^{\mu_A}(\Delta | \tau_B > \tau_A) + \mathbb{E}^{\mu_A}(\Delta | \tau_B < \tau_A) \end{array} \tag{3.5}$$

The big advantage [17] of this rewriting is that the right-hand side of (3.5) only contains quantities which can be computed either by sampling the reactive trajectories, such as $\mathbb{P}^{\mu_A}(\tau_B < \tau_A)$ or $\mathbb{E}^{\mu_A}(\Delta|\tau_B < \tau_A)$, or by brute force Monte Carlo, such as $\mathbb{E}^{\mu_A}(\Delta|\tau_B > \tau_A)$. As will be explained below, the sampling of reactive trajectories can be done efficiently using sampling methods or transition path sampling, for example. However, the difficulty is that μ_A is in general not analytically known, and that its sampling requires that equilibrium is reached. This is hard to simulate, because the transition to B is metastable. But, let us recall that, since A is in a metastable region, before visiting B, the system stays a considerable amount of time doing loops between A and its neighborhood. It is then possible to assume that the information about the entrance point from B is lost, and a quasi-stationary distribution is reached before B is visited[17]. We call ν_Q this distribution of entrance points in A, that substitutes $\mathbb{E}^{\nu_E}(T_{AB}) \approx \mathbb{E}^{\nu_Q}(T_{AB})$, and which is formally defined as:

$$\forall x_0 \in A, \quad \forall S \subset A, \quad \nu_Q(S) = \lim_{n\to\infty} \mathbb{P}(x_n \in S|\tau_B > n).$$

Under some assumptions quantifying the metastability of the neighborhood of $x = -1$, it is possible to show (see [17]) that $\mathbb{E}^{\nu_E}(T_{AB})$ is close to $\mathbb{E}^{\nu_Q}(T_{AB})$. Moreover, using again the Hill relation, one can show that the equivalent of (3.4) starting from ν_Q rather than ν_E is:

$$\mathbb{E}^{\nu_Q}(T_{AB}) = \frac{\mathbb{E}^{\nu_Q}(\Delta)}{\mathbb{P}^{\nu_Q}(\tau_B < \tau_A)}.$$

Notice that ν_Q is much easier to sample than μ_A, using a free dynamics over A, since no transition to B is required.

As above, the previous equation can be rewritten as:

$$\begin{aligned} \mathbb{E}^{\nu_Q}(T_{AB}) &= \frac{\mathbb{E}^{\nu_Q}(\Delta|\tau_B > \tau_A)\mathbb{P}^{\nu_Q}(\tau_B > \tau_A) + \mathbb{E}^{\nu_Q}(\Delta|\tau_B < \tau_A)\mathbb{P}^{\nu_Q}(\tau_B < \tau_A)}{\mathbb{P}^{\nu_Q}(\tau_B < \tau_A)} \\ &= \left(\frac{1}{\mathbb{P}^{\nu_Q}(\tau_B < \tau_A)} - 1\right)\mathbb{E}^{\nu_Q}(\Delta|\tau_B > \tau_A) + \mathbb{E}^{\nu_Q}(\Delta|\tau_B < \tau_A). \end{aligned} \tag{3.6}$$

In the first term, $\mathbb{E}^{\nu_Q}(\Delta|\tau_B > \tau_A)$ corresponds to the average time between two subsequent entrances of A without visiting B, and $(1/\mathbb{P}^{\nu_Q}(\tau_B < \tau_A) - 1)$ is the average number of loops from A back to A before a transition to B. The second, $\mathbb{E}^{\nu_Q}(\Delta|\tau_B < \tau_A)$, corresponds to the average time of the reactive trajectory, between the last entrance in A and the following entrance in B.

In summary, the algorithm to estimate $\mathbb{E}^{\nu_E}(T_{AB}) \approx \mathbb{E}^{\nu_Q}(T_{AB})$ consists in:

- Sampling ν_Q and estimating $\mathbb{E}^{\nu_Q}(\Delta|\tau_B > \tau_A)$. This can typically be done by running the dynamics (3.1) as it leaves and enters back in A, without observing transitions to B.

- Using rare event simulation techniques to sample the reactive trajectories starting from ν_Q, to get estimates of $\mathbb{P}^{\nu_Q}(\tau_B < \tau_A)$ and $\mathbb{E}^{\nu_Q}(\Delta|\tau_B < \tau_A)$.

As previously mentioned, AMS gives estimators for the probability to first enter B rather than A, as well as the reactive trajectories duration, when starting from a given set of initial conditions. Thus, to obtain $\mathbb{P}^{\nu_Q}(\tau_B < \tau_A)$ and $\mathbb{E}^{\nu_Q}(\Delta|\tau_B < \tau_A)$ using AMS, the points in the initial condition's set need to be sampled according to ν_Q. However, it appears that the variance of the AMS estimator with initial conditions in ν_Q is large because only a few samples from the initial set contributes a lot to the final estimator. This is shown for the studied system in section 3.3.

3.2.4 The Adaptive Multilevel Splitting in 1D

In this section we present the adaptive multilevel splitting algorithm in our specific context, for a given set of initial conditions of N points. In this method, the interval $]-1,1[$ on x, between A and B, is splitted on the fly. This separates the transition from A to B into several more probable events, easier to simulate. Doing so, an estimation of the probability that, from an ensemble of N initial points, the system reaches B before going back to A, is obtained.

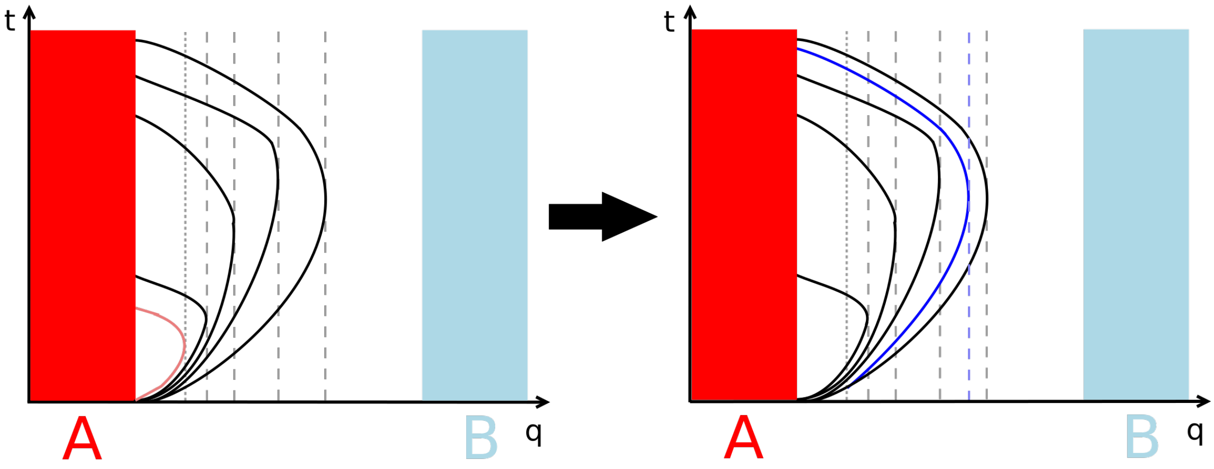


Figure 3.3 – First iteration of the AMS algorithm with $N = 5$. z^1_{kill} (fine dashed line) is the level of the replica that made less progress (in pink). This replica is then killed, and a new one (in blue) is generated by the replication of a survivor, randomly chosen.

Consider a set of N initial conditions (position and velocity). From each of those points, a trajectory is run following (3.1) until the particle enters A or B. This leads to the first set of replicas, denoted by $\left(X^0_n\right)_{n\in[1,N]}$. Each X^0_n is thus a trajectory. An initial weight of $w^0_n = 1/N$ is assigned to each replica.

To compute the progress of the replicas towards B, a reaction coordinate is needed. For the one dimensional system under study it is natural to use the position x for this purpose. The algorithm stops when all the replicas reach a fixed maximal level of the reaction coordinate. Here this level will be $x = 1$, which means that the particle is indeed inside B.

The algorithm then consists in repeating the following steps are repeated until the stopping criteria is satisfied:

- **Calculating the killing level**
  Iteration $q \geq 0$ start with the set of replicas $\left(X^q_n\right)_{n\in[1,N]}$ and the set of weights $\left(w^q_n\right)_{n\in[1,N]}$. Let us denote by z^q_n the maximum value of the position in x reached by replica n, called its level. The killing level z^{q+1}_{kill} will be $\min\left\{z^q_n \middle| n \in [1,N]\right\}$. The number of replicas with level lower or equal to z_{kill} is denoted by k^{q+1}. Notice that $k^{q+1} \geq 1$.

- **Stopping criteria**
  The algorithm stops if $z_{\text{kill}}^{q+1} > 1$ or if $k^{q+1} = N$. If one of those is true, the total number of iterations is set to $Q_{\text{iter}} = q$. Otherwise, the algorithm proceeds to the next step.

- **Killing and replicating**
  The k^{q+1} replicas with level less or equal to z_{kill}^{q+1} are killed. Among the $N - k^{q+1}$ remaining replicas, k^{q+1} are randomly chosen to be replicated. Replication consists in copying the replica up to the first point after z_{kill}^{q+1}, and then running the trajectory independently until it reaches A or B. The weight of the new replica is the same as the replicated one. At the end of this procedure there will be a new set of replicas $\left(X_n^{q+1}\right)_{n\in[1,N]}$. All the weights are then updated as:

  $$\forall n \in [1, N],\, w_n^{q+1} = w_n^q \left(\frac{N - k^{q+1}}{N}\right).$$

  The iteration counter q is then incremented by one and the algorithm returns to the first step, "Calculating the killing level".

At the end of the algorithm, the estimation for the probability $\mathbb{P}(\tau_B < \tau_A)$, starting from the set of initial points, is the sum of weights of all particles that eventually entered B:

$$p_{AMS} = \sum_{n=1}^{N} w_n^{Q_{\text{iter}}} \mathbb{1}_{\tau_B^{n,Q_{\text{iter}}} < \tau_A^{n,Q_{\text{iter}}}},$$

where $\tau_A^{n,Q_{\text{iter}}}$ (resp. $\tau_B^{n,Q_{\text{iter}}}$) is the first time that replica n enters A (resp. B) at the last iteration Q_{iter}. In our specific case, since $B = \{q > 1\}$, all the replicas actually enters B at iteration Q^{iter} if $z_{\text{kill}}^{Q^{\text{iter}}+1} > 1$, and thus $\forall n \in [1, N], \mathbb{1}_{\tau_B^{n,Q_{\text{iter}}} < \tau_A^{n,Q_{\text{iter}}}} = 1$. However, if $k^{Q^{\text{iter}}+1} = N$, none of the N replicas reach B at iteration Q_{iter}, and then $p_{AMS} = 0$. Notice that if all the replicas in the initial set reach B, then $p_{AMS} = 1$.

It is shown in [22] that the expected value of p_{AMS} is $\mathbb{P}(\tau_B < \tau_A)$, regardless of the choice of the algorithm parameters. Therefore, the results of this chapter will be mean values of the estimation obtained among independent AMS runs.

Let us mention that AMS can also be used to get estimations of any path functional. This property of the method will be used below to obtain the mean velocity along the reactive trajectories, following the procedure we will now describe. The space between A and B is split, producing the intervals $(I_l)_{1\le l\le L}$. We gather all the reactive trajectories built over M different AMS runs, as well as their weights at the last iteration. This yields an ensemble of trajectories $\left(q_{n,i}^j, v_{n,i}^j\right)_{j\in[0,T_{n,i}]}$ and associated weights $\left(w_{n,i}\right)$, where $i \in [1, M]$ and corresponds to the different AMS runs, and $n \in [1, N]$. The mean velocity along the reactive trajectories in interval I_l is estimated by:

$$\overline{v}(I_l) = \frac{\sum_{i=1}^{M}\sum_{n=1}^{N} w_{n,i} \sum_{j=0}^{T_{n,i}} v_{n,i}^j \mathbb{1}_{q_{n,i}^j \in I_l}}{\sum_{i=1}^{M}\sum_{n=1}^{N} w_{n,i} \sum_{n=0}^{T_{n,i}} \mathbb{1}_{q_{n,i}^j \in I_l}}, \tag{3.7}$$

where $T_{n,i}$ the duration of the n^{th} replica from the i^{th} AMS run. A discrete representation of the mean velocities along the reactive trajectories is then $(x_l, \overline{v}(I_l))_{1\leq l\leq L}$, where x_l is the center of interval I_l.

3.3 Numerical results ans a new importance sampling procedure for the initial conditions

The 1D potential from figure 3.1 was already studied in [1]. This paper used the RETIS and FFS methods (see Chapter 1), and exhibits inconsistency in the FFS results. In this section, we report on numerical results obtained using AMS, where one expects to obtain similar results as for FFS in [1], since AMS is also a splitting method, which can be seen as an adaptive version of FFS. Results of these numerical experiments are presented in section 3.3.1.

In section 3.3.2, we calculate an analytical expression for the distribution of initial points, and the results obtained using this function, but this time following another protocol for the sampling, based on this analytical expression. We show that the inconsistent results obtained in section 3.3.1 originate from an insufficient sampling of the initial condition. Next, in section 3.3.3, we explain how to use an importance sampling technique, together with AMS, to sample the initial points more efficiently. The section also contains the results obtained using the optimal importance function, in order to explore the maximum computational gain one can hope. We then propose, in section 3.3.4, an adaptive importance sampling technique to efficiently sample the distribution at equilibrium without a previous knowledge of the optimal importance function.

Besides, the small dimension of the problem allowed us to obtain brute force results to be compared with the AMS estimations. An independent free dynamics over a total of 3.3×10^{12} timesteps were carried out to generate direct numerical simulation results (DNS), which are refer to below as the reference results.

As a preliminary remark, we would like to emphasize a change of notation from section 3.2 to section 3.3. In order to stick with the notation used in other publications [16, 17, 42, 45], we so far defined the transition time as the average time from the first entrance in A to the first entrance in B, with loops defined between successive entrances in A. In this section, and in accordance with other works [38, 46], we will actually remove from the transition time the first part between the first entrance in A and the first exit from A, and we will work with loops defined between successive exits from A. In practice, for metastable systems, this does not change the value of the transition time since this is only a small irrelevant part of the transition path. With this new definition of the transition time, equations (3.5) and (3.6) are still valid, changing the distributions ν_E, μ_A and ν_Q to their counterparts ν_E^{ex}, μ_A^{ex} and ν_Q^{ex}, which are just obtained by considering the map which to a distribution μ in A associates the distribution of the first exit point from A following the dynamics (3.1) starting from μ.

3.3.1 Reproducing the numerical experiment from [1]

In order to sample initial points according to the quasi-stationary distribution and estimate the mean duration time for the loops, $\mathbb{E}^{\nu_Q^{ex}}(\Delta|\tau_A < \tau_B)$, a dynamics of 10^7 steps was run. During this trajectory, the position and velocity of all the exits of A where kept. This procedure generated an ensemble of initial conditions with 8867 points. A total of a thousand AMS runs, with $N = 8867$, were made using this fixed ensemble of initial conditions.

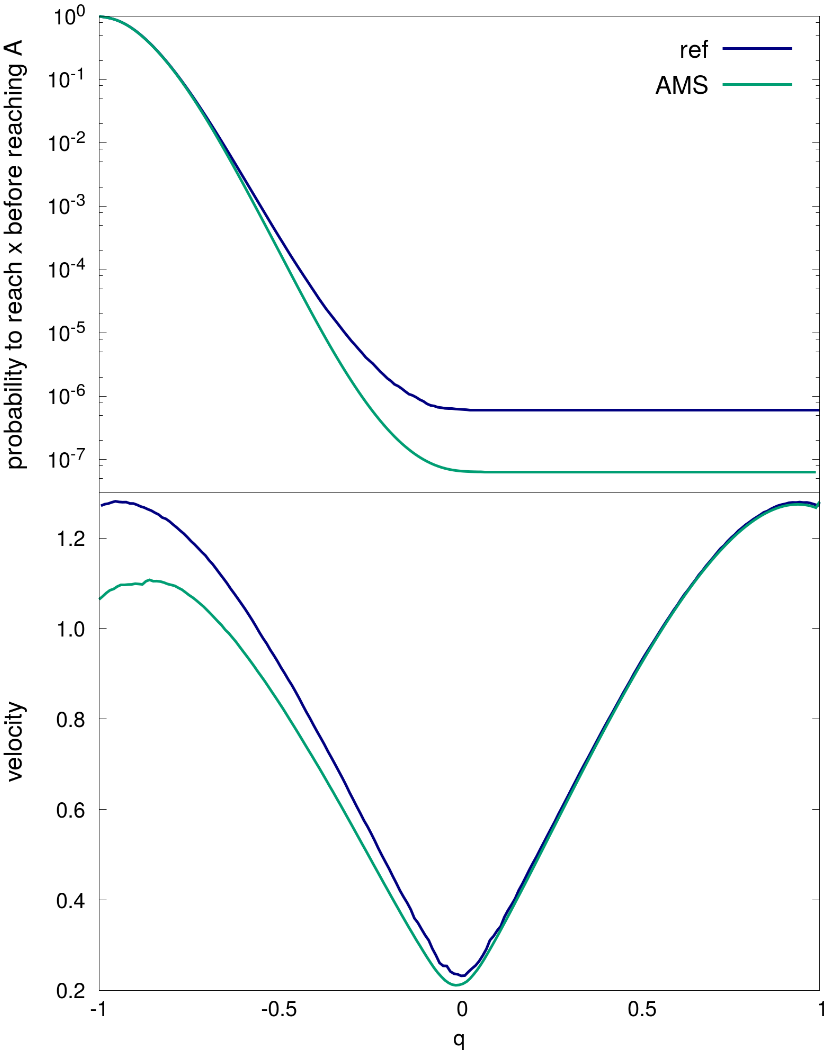


Figure 3.4 – Estimators of the probability $\mathbb{P}^{\nu_Q^{ex}}(\tau_x < \tau_A)$ to reach x before coming back to A, and the mean velocity along the reactive trajectories. The AMS result was generated using a fixed set of 8867 points generated with a 10^7 timesteps free dynamics.

The obtained results are consistent with those published in [1], and hence present the same issues that will now be discussed. Notice from figure 3.4, that the probability $\mathbb{P}^{\nu_Q^{ex}}(\tau_q < \tau_A)$, to reach position q before entering A, is underestimated by AMS, compared to the reference result. Accordingly, the estimator for the probability $\mathbb{P}^{\nu_Q^{ex}}(\tau_B < \tau_A)$ is smaller than the reference value, and consequently, the transition time, estimated using (3.6), is overestimated. The value obtained with AMS is $(1.63 \pm 0.20) \times 10^7$, and the reference value is $(3.63 \pm 0.08) \times 10^6$.

The second graph in figure 3.4 show the mean velocity along the reactive paths simulated by AMS, obtained using equation (3.7). To generate this trajectory, the space between A and B is split into intervals of size 0.01. The total ensemble of 8,867,000 reactive trajectories is used. The brute force DNS result is obtained using the 1800 reactive paths observed along the long free dynamics of 3.3×10^{12} timesteps. The potential $V(q)$ and the set $A \cup B$ are invariant under the transformation $q \to -q$, and thus the mean velocity along the reactive trajectories at equilibrium should be also invariant under the same transformation, using the fact that the equilibrium trajectories are time reversible up to momentum reversal. Although this was confirmed by the DNS result, it was not reproduced by the trajectories generated with AMS. For the latter, the initial velocity is smaller than the reference estimation. This suggests a lack of initial points with higher velocity in the ensemble of initial conditions.

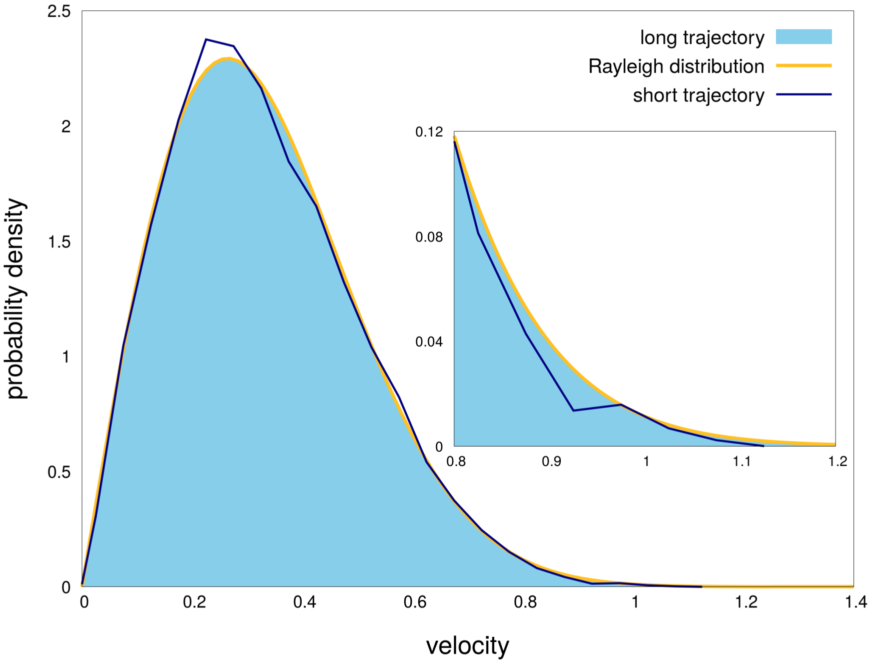


Figure 3.5 – Comparison between the exit distribution of velocities over ∂A, obtained by brute force (3.3×10^{12} timesteps), and the estimator, done with a smaller trajectory of 10^7 timesteps.

This hypothesis can indeed be confirmed by comparing the distribution of velocities on the ensemble of initial conditions used for AMS with the reference distribution (see figure 3.5): this is the direct comparison between the samples of v_Q^{ex}, obtained by brute force (long trajectory of 3.3×10^{12} timesteps), and the 8867 samples used for AMS, obtained with a much smaller trajectory of 10^7 timesteps. Figure 3.5 also shows a zoom of this comparison on the tail of the distribution. One can see that the procedure used to sample the initial points fail to sample higher velocities if the trajectory is too short. This is the source for all the problems observed here and also in [1], as will become clear below. More precisely, the fact that all the AMS starts with a fixed set of 8867 samples implies an undersampling of the high velocity tail, which in turn implies an underestimation of the probability to reach B before A. This is also in accordance with the results in [23] where we showed that a very god sampling of v_Q^{ex} is required to get correct estimations of the transition time on a more complicated system.

In [23], this issue was solved by redrawing new initial conditions before each AMS run with a preliminary simulation. Therefore, we performed AMS runs, were before each one of them a dynamics is run until a total of a thousand loops (between A and back to A) is made. This generates an ensemble of $1,000$ points for the initial condition. Each preliminary simulation starts from the end point of the last preliminary simulation. Figure 3.6 show the results obtained with this experience, which are consistent with the reference results. The AMS simulations are run until the convergence of the probability estimator was reached. For that, it was necessary to run 150,000 AMS simulations (see figure 3.7), which means a total of $1,5 \times 10^8$ initial conditions were used. This shows how difficult it is to sample the queue on the distribution ν_Q^{ex}.

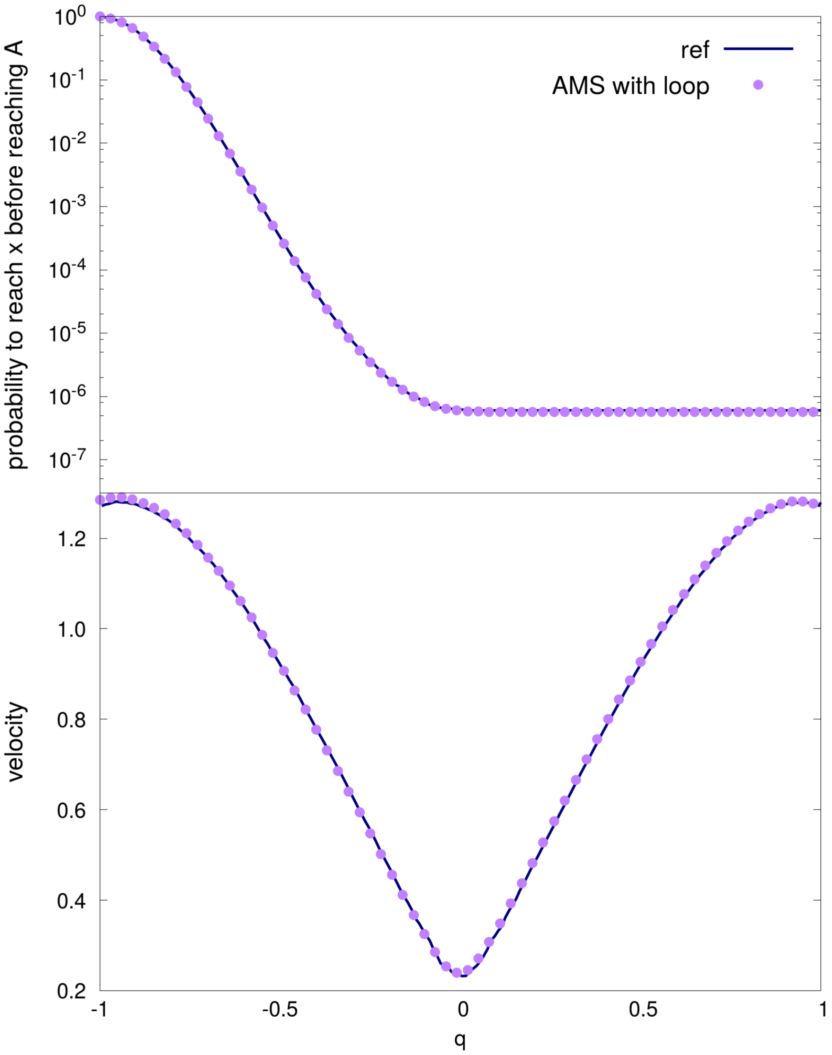


Figure 3.6 – Estimators of the probability $\mathbb{P}^{\nu_Q^{ex}}(\tau_q < \tau_A)$ to reach q before coming back to A, and the mean velocity along the reactive trajectories. The AMS result was generated using a fixed set of 8867 points generated with a 10^7 timesteps free dynamics.

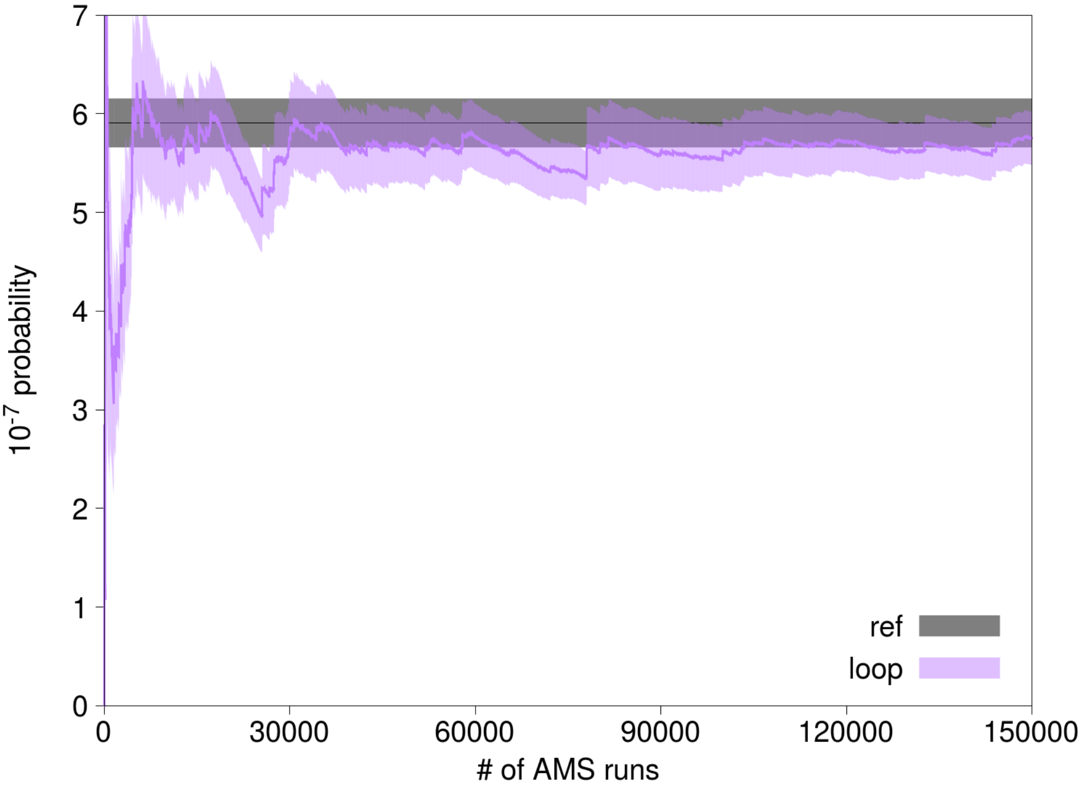


Figure 3.7 – Convergence of the probability estimated by AMS when using the Rayleigh distribution to sample the initial points.

3.3.2 Correct distribution for the initial conditions

In order to prove that the errors in the results were caused by the sampling of the initial conditions, and not by AMS, the experience was repeated using another strategy to sample the initial points. More precisely, we calculated an analytical expression for the exit equilibrium distribution μ_A^{ex}, which is possible in our simple setting.

Let us recall that (q_n, p_n) is the position and momentum of the particle at time $n\Delta t$, discrete approximate solution for the Langevin dynamics with the potential $V(q)$ (see equation (3.1)). Let us introduce $v_n = p_n/m$ the velocity at the same time, to simplify the notation. Because we aim at calculating the exit distribution from A, called μ_A^{ex}, only the points where the particle is inside A at time n, and outside at time $n+1$, are concerned. Therefore, the following relation holds for those points:

$$q_n < -1 < q_n + v_n \Delta t. \tag{3.8}$$

Denoting by $\phi(v)$ a test function, its expected value under μ_A^{ex} is:

$$\mathbb{E}^{\mu_A^{ex}}(\phi(v)) = \lim_{T\to+\infty} \frac{\sum_{n=1}^{T} \phi(v_n)\mathbb{1}_{q_n<-1<q_n+v_n\Delta t}}{\sum_{n=1}^{T} \mathbb{1}_{q_n<-1<q_n+v_n\Delta t}} \tag{3.9}$$

Let us consider the limit when T goes to infinity to substitute the sums by integrals, and let us call $\mu_{\Delta t}$ the invariant measure of the chain $(q_n, v_n)_{n\geq 0}$. Then:

$$\mathbb{E}^{\mu_A^{ex}}(\phi(v)) = \frac{\int \phi(v)\mathbb{1}_{q<-1<q+v\Delta t}\mu_{\Delta t}(dv, dq)}{\int \mathbb{1}_{q<-1<q+v\Delta t}\mu_{\Delta t}(dv, dq)}. \tag{3.10}$$

Now, using the fact that:

$$\mu_{\Delta t} \xrightarrow[\Delta t\to 0]{} e^{-V(x)-\frac{mv^2}{2}} dvdq,$$

and that, in our setting $m = 1$, we have:

$$\mathbb{E}^{\mu_A^{ex}}(\phi(v)) \approx \frac{\int \phi(v)e^{-\frac{v^2}{2}} \int \mathbb{1}_{q<-1<q+v\Delta t}e^{-V(q)}dqdv}{\int e^{-\frac{v^2}{2}} \int \mathbb{1}_{q<-1<q+v\Delta t}e^{-V(q)}dqdv}. \tag{3.11}$$

One can now take the limit for $\Delta t \to 0$, fixing the position in $q = -1$. This gives the distribution only over the velocities, that we will call $\mu_A^{ex,v}$, so that $\mu_A^{ex} = \delta_{-1}(dq) \times \mu_A^{ex,v}$. By taking the limit $\Delta t \to 0$ in (3.11), one then obtains:

$$\mathbb{E}^{\mu_A^{ex,v}}(\phi(v)) = \frac{\int \phi(v)\mathbb{1}_{v>0}ve^{-\frac{v^2}{2}}dv}{\int \mathbb{1}_{v>0}ve^{-\frac{v^2}{2}}dv}. \tag{3.12}$$

Hence, the distribution of velocity is the Rayleigh distribution:

$$\mu_A^{ex,v} = \mathbb{1}_{v>0}ve^{-\frac{v^2}{2}}dv. \tag{3.13}$$

Let us now run AMS with initial velocities sampled according to the Rayleigh distribution. Before each run, a thousand velocities are sampled according to the Rayleigh distribution and used as initial points, with the position at $q = -1$. Let us emphasize that we here use a new ensemble of initial conditions for each AMS run. Figure 3.8 shows the probability $\mathbb{P}^{\mu_A^{ex,v}}(\tau_q < \tau_A)$, as a function of q. The graph from the top presents the obtained mean velocity along the reactive trajectories, (see equation (3.7)). Both are consistent with the reference results. This shows that, not only a correct sampling of the equilibrium distribution was reached, but also that AMS does not fail. Therefore, this confirms that the problems with the results from section 3.3.1 were caused by the bad sampling of the initial condition distribution.

Figure 3.9 shows the convergence of the probability estimator $\mathbb{P}^{\mu_A^{ex,v}}(\tau_B < \tau_A)$. Even if the number of AMS run is smaller than in the previous experience, it is again large (100,000). This means that the

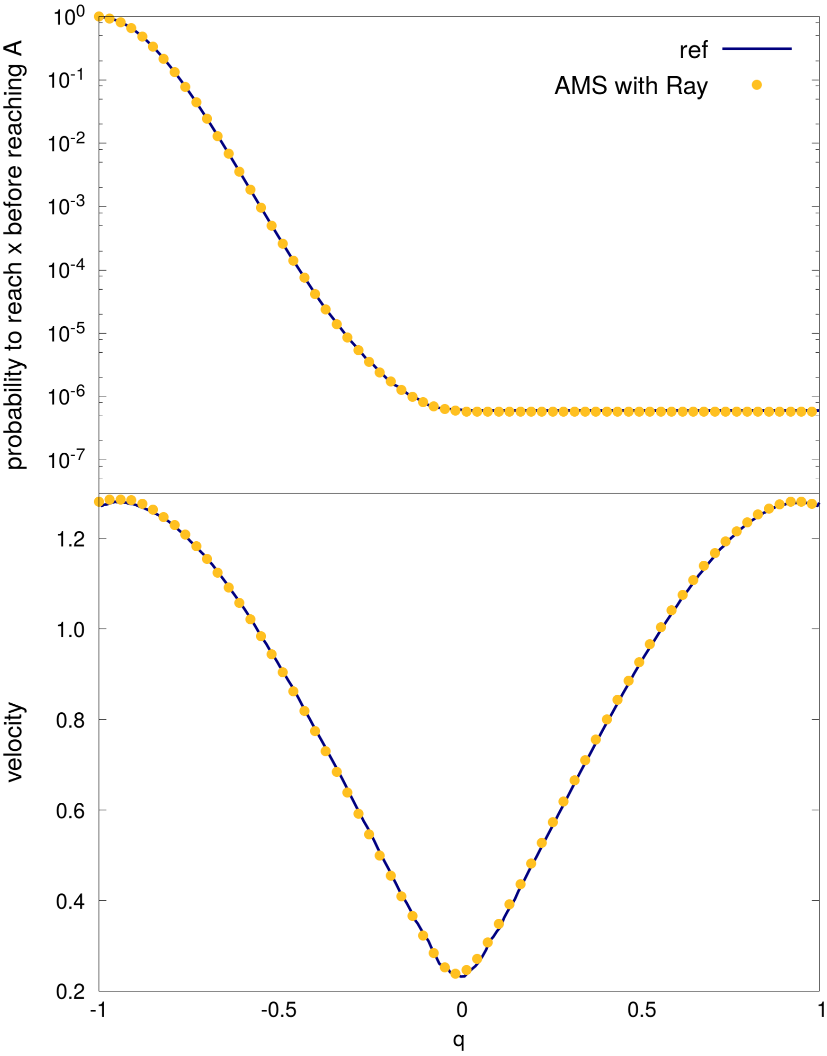


Figure 3.8 – Results obtained using the Rayleigh distribution to sample the initial points (to be compared with figure 3.4).

total number of velocities sampled was 10^8. This again demonstrates the necessity to sample a high number of initial points to ensure the correct sampling on the tail of the Rayleigh distribution. Thus, despite of the success in reproducing the reference results, this was achieved at a high computational cost. To address this issue, we propose to use an importance sampling procedure, introduced in the next section for the one dimensional problem.

3.3.3 Importance Sampling for the initial condition

Let us denote by $P(v) = \mathbb{P}^{v}(\tau_B < \tau_A)$, the probability to reach B before A, starting with initial velocity v and initial position $q = -1$. The estimator of the probability $\mathbb{P}^{\mu_A^{ex,v}}(\tau_B < \tau_A)$ can be obtained as the

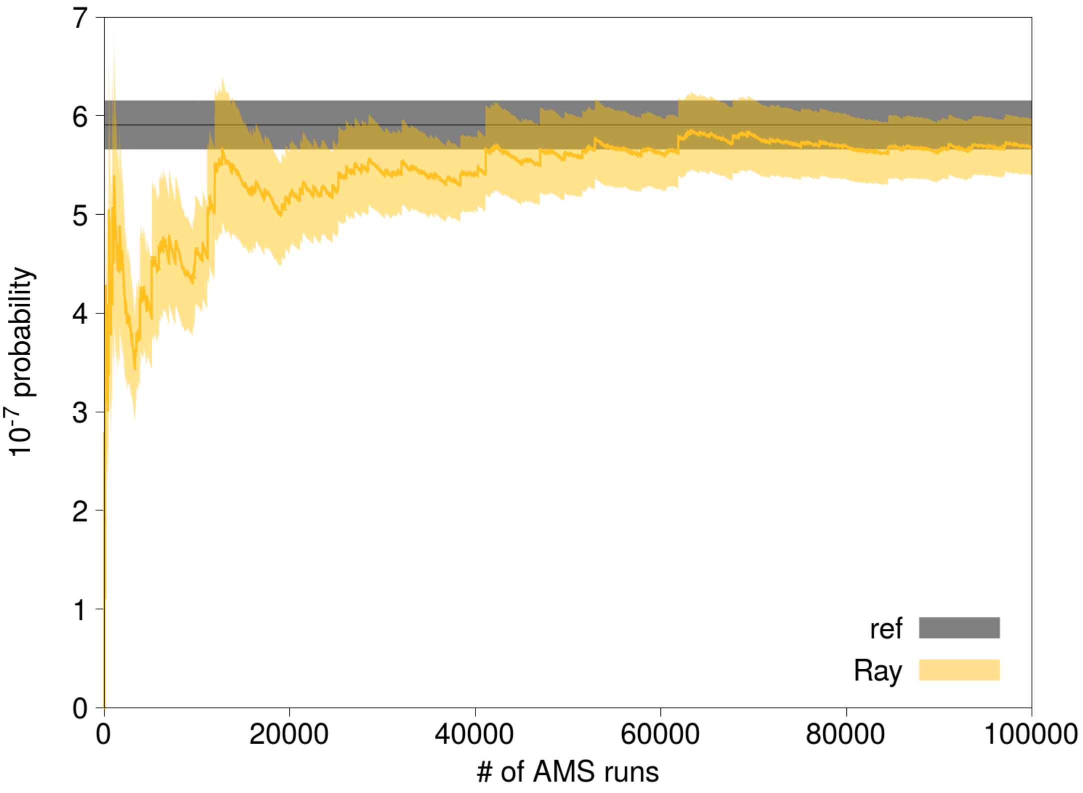


Figure 3.9 – Convergence of the probability estimated by AMS when using the Rayleigh distribution to sample the initial points.

integral of $P(v)$, under the distribution $\mu_A^{ex,v}$, as:

$$\mathbb{P}^{\mu_A^{ex,v}}(\tau_B < \tau_A) = \int_0^{+\infty} P(v)\mu_A^{ex,v}(v)dv.$$

The AMS method calculates this integral as the expected value $\mathbb{E}^{\mu_A^{ex,v}}(P(Y))$, with $Y \sim \mu_A^{ex,v}$. One possible solution to reduce the computational cost is to employ an importance sampling technique on the initial conditions. Let us call $f(v)$ the importance function. The integral then reads:

$$\mathbb{P}^{\mu_A^{ex,v}}(\tau_B < \tau_A) = \int_0^{+\infty} \frac{P(v)}{f(v)} f(v)\mu_A^{ex,v}(v)dv.$$

If we suppose that $\forall v \geq 0, f(v) \geq 0$ and $\int_0^{+\infty} f(v)\mu_A^{ex,v}(v)dv = 1$, then $f(v)\mu_A^{ex,v}(v)dv$ can be seen as a distribution. Thus, introducing a random variable Z with law $f(v)\mu_A^{ex,v}(v)dv$, the expected value writes:

$$\mathbb{P}^{\mu_A^{ex,v}}(\tau_B < \tau_A) = \mathbb{E}^{\mu_A^{ex,v} f}\left(\frac{P(Z)}{f(Z)}\right). \tag{3.14}$$

Let us recall that AMS attributes a weight to each initial trajectory. The expected value from equation (3.14) can then be calculated with AMS by sampling the initial points according to $\mu_A^{ex,v} f$, and computing the initial weights using the function f. More precisely, the weight at iteration 0 of replica n that has as a velocity v_n, is:

$$w_n^0 = \frac{1}{Nf(v_n)}.$$

It is known that the importance function that optimizes the variance of the estimator, called the optimal importance function, is the integrand, namely $P(v)$ in our setting. Let us call $\mu_{\text{opt}}(v)$ the associated optimal distribution, defined as:

$$\mu_{\text{opt}}(v) = \frac{\mathbb{1}_{v>0} v e^{-\frac{v^2}{2}} P(v)}{\int_0^{+\infty} u e^{-\frac{u^2}{2}} P(u) du}.$$

The function $P(v)$ is estimated with AMS for a set of values of v between 0 and 3. For each value, a thousand AMS simulations are run, with 1000 replicas. The function $P(v)$ is then approximated by a piecewise affine function (see figure 3.10).

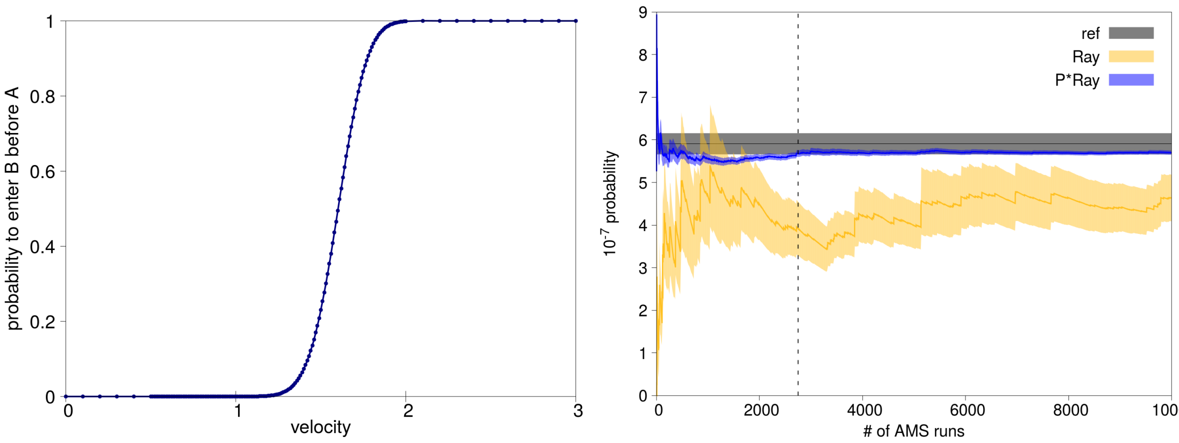


Figure 3.10 – Left: Estimation of $P(v)$ with AMS. Each point of the function was obtained by running a thousand AMS simulations with 10^3 replicas, where the initial condition was a fixed point with position $x = -1$ and velocity in the interval $v \in [0,3]$. Right: Convergence of the probability estimator for $\mathbb{P}^{\mu_A^{ex,v}}(\tau_B < \tau_A)$, obtained with AMS using the optimal importance sampling function (to be compared with figure 3.9, partially reproduced here in yellow).

To test the computational gain implied by the importance sampling, we run 10^4 AMS simulations. For each AMS run, 1000 initial points are sampled according to $\mu_{\text{opt}}(v)$. Notice from figure 3.10 that the convergence is reached at only 2,750 runs (dashed line), and compare it with results from figure 3.9, where 100,000 runs were made. This means the computational time was divided by 36, with respect to AMS from unbiased initial conditions.

In front of this highly satisfactory result, we now try to come up with a new method, based on the usage of importance sampling, but that does not require to know the optimal importance function and thus could be apply to more general problems. This is developed in the following section.

3.3.4 An adaptive importance sampling technique

For more complex systems it is impossible to compute the optimal importance function, so another method is need. Equation (3.15) (see also [23], equation (28)) estimates the probability to reach B when starting with the velocity in interval C_l. This yields a piecewise constant approximation of $P(v)$. Notice that equation (3.15) involves all the $(N_i)_{i\in[1,M]}$ trajectories simulated over the M AMS runs, including the killed ones. The interval of velocities $[0,3]$ is split into L cells and $\widetilde{P}(C_l)$ is defined over each cell $(C_l)_{l\in[1,L]}$ as:

$$\widetilde{P}(C_l) = \frac{\sum_{i=1}^{M}\sum_{n=1}^{N_i} w_{n,i}\mathbb{1}_{\tau_B^{n,i}<\tau_A^{n,i}}\mathbb{1}_{v_0^{n,i}\in C_l}}{\sum_{i=1}^{M}\sum_{n=1}^{N_i} w_{n,i}\mathbb{1}_{v_0^{n,i}\in C_l}}. \tag{3.15}$$

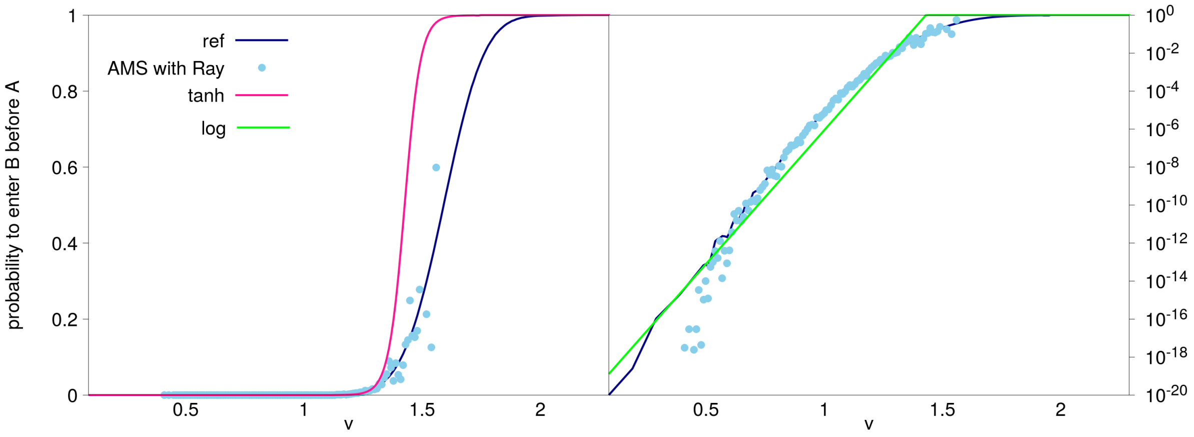


Figure 3.11 – Values of the function $\widetilde{P}(C_l)$ obtained using the equation (3.15), from 10^4 AMS with 1000 replicas each , fitted using functions $P_{\log}$ and $P_{\tanh}$.

Figure 3.11 shows the function $\widetilde{P}(C_l)$ estimated with (3.15), using the trajectories obtained from 10^4 AMS runs, using the Rayleigh distribution to sample the initial conditions. Thanks to our previous knowledge of $P(v)$, a fit procedure adequate to its form can be used. We will discuss more general procedures in section 3.4. This allows us to construct estimators of $P(v)$ using a small quantity of points. Two different functions are used to fit the points of the function estimated with AMS. The first is a hyperbolic tangent and the second a truncated exponential:

$$\begin{aligned} P_{\tanh}(v) &= \frac{\tanh(av+b)+1}{2} \\ P_{\log}(v) &= \min\left(e^{(av+b)},1\right). \end{aligned} \tag{3.16}$$

Both fit were done by applying the arctanh function (for $P_{\tanh}$) or the logarithm function (for $P_{\log}$) to the data and then performing a linear least square fit. Let us call v_l the center of interval C_l, and $S\subset[1,L]$ the indices for which the values estimated using (3.15) are positive, i.e. $\forall l\in S, \widetilde{P}(C_l)>0$. For the hyperbolic tangent, the fit was done over the points $\left(v_l, \operatorname{arctanh}\left(2\widetilde{P}(C_l)-1\right)\right)_{l\in S}$. To obtain the parameters for the truncated exponential function, the fit was done using $\left(v_l, \ln\left(\widetilde{P}(C_l)\right)\right)_{l\in S}$.

Using these fitting procedures, we end up with an adaptive algorithm to estimate $P(v)$ on the fly and to efficiently sample the initial points. The first guess for the probability function is $P_0(v) = 1, \forall v \in [0,3]$. The algorithm consists in the following steps until the desired convergence of the probability estimator is reached.

- **Sampling of the initial condition**
  Iteration s starts with the importance function $P_s(v)$. The distribution to sample the initial condition is defined as:
  $$\mu_s(v) = \frac{ve^{-\frac{v^2}{2}}P_s(v)}{\int_0^3 ue^{-\frac{u^2}{2}}P_s(u)du}, v \in [0,3].$$
  The set $(v_n^s)_{1\le n\le N}$ of velocities is sampled according to μ_s. Each point receives the weight:
  $$\alpha_n^s = \frac{\int_0^3 ue^{-\frac{u^2}{2}}P_s(u)du}{P_s(v_n^s)N}.$$

- **Running AMS**
  The set $(-1, v_n^s)_{1\le n\le N}$ is fixed as the initial conditions for the next K AMS runs. For each replica n, the associated AMS weight at the initialization of the replicas (see section 3.2.4) is:
  $$w_n^0 = \alpha_n^s.$$
  During each AMS run, the results for the sum at the numerator and the denominator of equation (3.15) are updated.

- **Updating the importance function**
  After completing K AMS simulations, the points $\left(\tilde{P}_s(C_l)\right)_{1\le l\le L}$ are computed. The positive estimators among those points are fitted, giving the new importance function $P_{s+1}(v)$. The iteration counter is incremented by one ($s := s+1$) and algorithm returns to the "Sampling of the initial condition" step.

Two sets of simulations were run with the described algorithm, one for each of the two fitting procedures to compute the importance function (see (3.16)). A fixed number of one thousand iterations were performed. Figure 3.12 shows the convergence for this algorithm with the distribution to sample the initial points updated every 10 AMS, i.e. $K = 10$. Although both fitting functions gives consistent results, $P_{\log}(v)$ converges faster, almost at the same speed as the optimal importance sampling.

Figure 3.13 shows the evolution of the importance function as a function of the iterations of the algorithm. Notice that both fitting functions converge. Even though the final importance functions are not in complete agreement with the reference function $P(v)$, the computational cost is divided by 21 for the tanh, and by 164 for the log compared to the results obtained using the Rayleigh distribution. This shows that one does not need a precise estimation of the function $P(v)$ to obtain very significant computational gains.

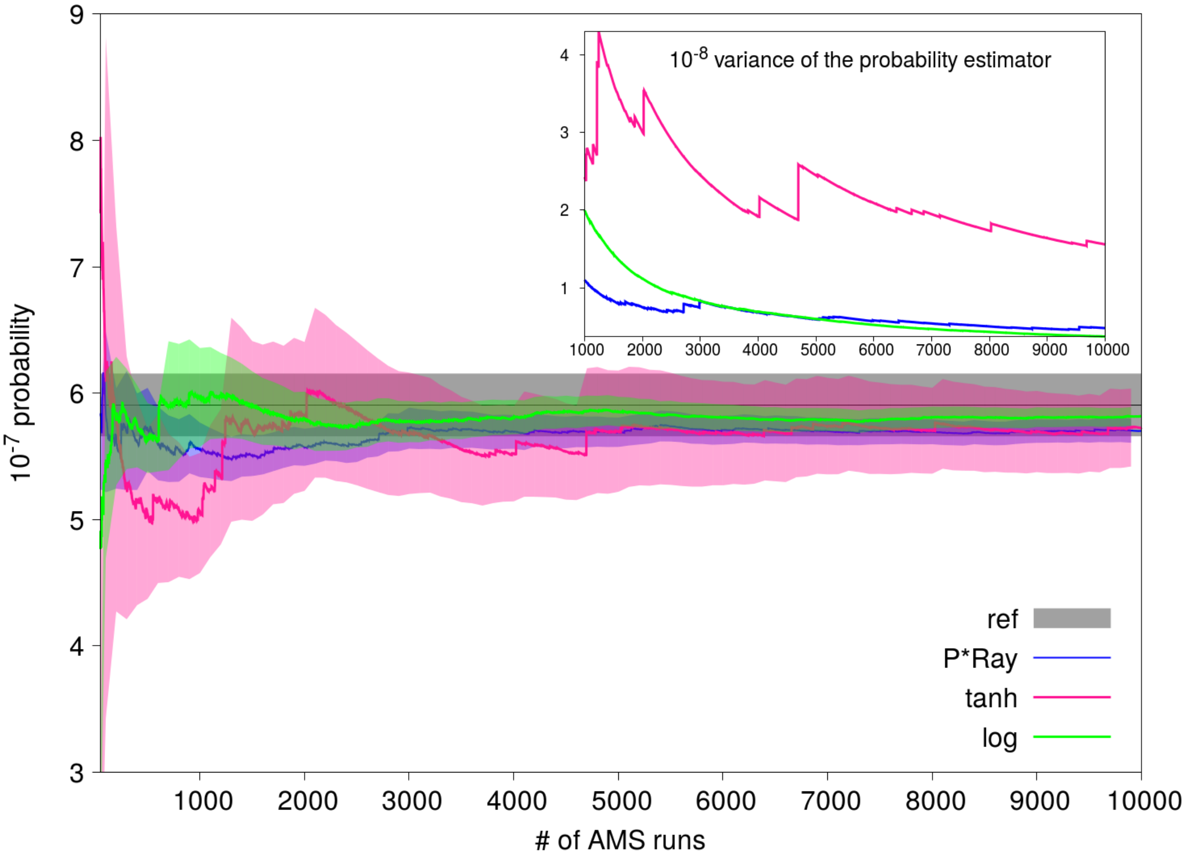


Figure 3.12 – Results for the adaptive importance sampling algorithm with $K = 10$.

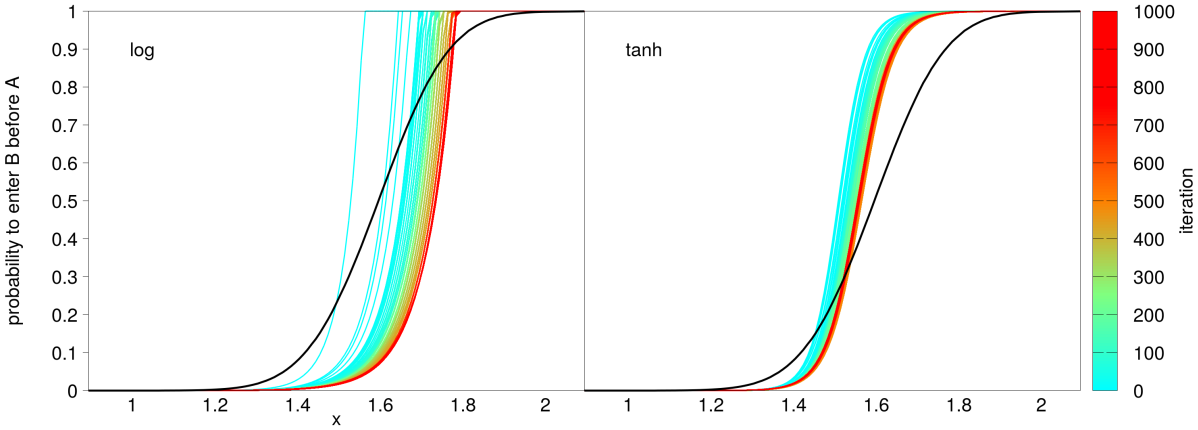


Figure 3.13 – Evolution of the importance function during the algorithm iterations, using both functions $P_{\log}$ and $P_{\tanh}$ for the fit.

3.4 Conclusion and Perspectives

The study over the one dimensional potential allowed us to exhibit the difficulty to sample initial points when using AMS to calculate the transition time. It is not only hard to correctly sample the initial conditions, but also to do it efficiently.

Using the estimation of the committor function calculated with AMS, we were able to propose an adaptive method to efficiently sample the initial condition, and reach convergence of the AMS estimator with a computational cost divided by 21 and 164, depending on the fitting procedure used. Moreover, our results show that a precise estimation of $P(v)$ is not needed to enable a more effective sample of the Rayleigh distribution. The proposed algorithm is then a potential candidate for applications on multidimensional systems, were the estimation of $P(v)$ is harder.

To adapt the method to more general situations, two difficulties need to be overcome. First, for a given importance function, the question of how to sample the biased quasi-stationary distribution need to be addressed. Typically, the importance function will only depend on a few collective variables, which are those used to define the states. For alanine-dipeptide, these would be two dihedral angles. Then, to sample the biased quasi-stationary distribution, one could think of two techniques. The first consists in drawing from the ensemble of samples distributed according to the quasi-stationary distribution, obtained as usual from a dynamics run in the neighborhood of A, each sample being weighted with the biasing function. The second, more involved, would be to modify the dynamics in order to directly sample the distribution of interest. To achieve this, techniques based on the weighted ensemble method could be used[47, 48].

The second difficulty is to adapt the fitting procedure to a more general setting. This should be possible for example if A is defined using up to four collective variables, by parameterizing the boundary of A, and by building some estimate of P using this parameterization. Otherwise, high-dimensional interpolation techniques could be used to build sensible approximation of P from values obtained at a few points. We intend to test these ideas to the alanine dipeptide case shortly.

Chapter 4

Elucidating mechanisms through the clustering of reactive trajectories

4.1 Introduction

Elucidation of transition mechanisms has been a research topic for the last three decades. Despite of that, the literature on the subject does not provide a clear definition of what should be the transition mechanism[24–27]. Moreover, many of the previous work assume that there is only one possible mechanism, which is false in complex molecular systems. The transition tubes method introduced by Vanden-Eijnden in [16] was the first method to consider more than one mechanism, but these tubes are not uniquely defined.

Clustering is a data analysis method that assembles the data in Voronoi cells. Those are defined by their centers and the distance over the data space. By performing a clustering of the reactive trajectories, the representative paths from each cluster can be considered as a possible mechanism of reaction. In addition, the clustering technique not only enables the existence of more than one mechanism, but also associates a probability to each one.

In this chapter we present two different clustering techniques to find transition mechanisms. In the first method, the clustering is performed over the original trajectories, and the centroids belong to the data ensemble. For the second, a data transformation is done in order to reduce the dimension of the problem. We apply these techniques to two problems: a bi-channel potential in two dimensions and a molecular toy model. The work presented in this chapter was made in collaboration with Jacques Printems (LAMA-UPEM).

4.2 Methods

The clustering is performed in order to elucidate possible mechanisms of a metastable transition. These transitions are rare, and therefore need the use of an appropriate method to be simulated, otherwise the computational cost is impractical. The chosen method is the adaptive multilevel splitting (AMS), which gives an ensemble of weighted trajectories sampling the reactive trajectories ensemble (see Chapter 2 for a complete description of the method). This method accelerates the sampling by splitting the transition into a sequence of conditional events, that are more probable and thus easier to simulate. This procedure produces a set of branched trajectories. We will call A the state of origin and B the target, such that the reactive trajectories links A to B.

Let us call $\left(X_{n,t}\right)_{n\in[1,N],t\in[0,T_n]}$ the ensemble of discrete reactive trajectories sampled with AMS, and $(w_n)_{n\in[1,N]}$ their weights. An estimator of the expected value of any path functional F defined on the set of reactive trajectories can be obtained as:

$$\mathbb{E}\left(F\left(X_{t\in[0,\tau_B]}\right)\mathbb{1}_{\tau_B<\tau_A}\right) \sim \frac{1}{N}\sum_{n=1}^{N} w_n F\left(X_{n,t\in[0,T_n]}\right),$$

where τ_A (resp. τ_B) is the first time to enter A (resp. B). The average thus only involves reactive trajectories, for which $\tau_B < \tau_A$. Considering the problem to be in dimension d, $X_{n,t} \in \mathbb{R}^d$. We will here use the notation $\|\cdot\|$ for the Euclidean norm in $\mathbb{R}^d$.

Two different strategies were used for the classification. The first one is the Lloyd's algorithm, where the representative trajectories, also called centroids, are considered to be among the original ensemble. This is explained in section 4.2.1. Section 4.2.2 then presents the second strategy, where an initial data transformation is performed, allowing for a dimensionality reduction. The final centroids were found using the Kohonen's algorithm, followed by the Lloyd's algorithm. Contrary to the first method, in this second strategy, the final representative trajectories are described as a linear combination of the data, and thus do not belong to the data ensemble.

4.2.1 Clustering over the original trajectories

To perform a clustering, the data space is partitioned into Voronoi cells, where each element belongs to the cell whose center it is closest to. Thus, a notion of distance over the data space needs to be defined. One difficulty is to define a distance between trajectories that have different lengths. Two different strategies were employed. The first was to use two versions of the Fréchet distance, which consists in calculating the minimum of a function over all the possible time reparameterizations, allowing to compare trajectories with different lengths. The second strategy was to extend the trajectories' lengths, by considering them stationary from their end point up to the maximum trajectory time among the data ensemble.

To give a proper definition of those distances, let us introduce the trajectories' reparameterizations. Let us call $\gamma_{n,m} = \max(T_n, T_m)$, the maximum length between two trajectories X_n and X_m. The sequences $P = \left(p_0,\dots,p_{\gamma_{n,m}}\right)$ and $Q = \left(q_0,\dots,q_{\gamma_{n,m}}\right)$, are reparameterizations of respectively the trajectories X_n and X_m, if and only if:

$$\begin{aligned}&p_0 = q_0 = 0,\;\; p_{\gamma_{n,m}} = T_n,\;\; q_{\gamma_{n,m}} = T_m\\ &\forall\;\; i = 0,\dots,\gamma_{n,m}-1,\;\; p_{i+1} = p_i \text{ or } p_{i+1} = p_i + 1,\;\; q_{i+1} = q_i \text{ or } q_{i+1} = q_i + 1.\end{aligned}$$

In other words, $P : i \in \{0,\dots,\gamma_{n,m}\} \to \{0,\dots,T_n\}$ and $Q : i \in \{0,\dots,\gamma_{n,m}\} \to \{0,\dots,T_m\}$ are non-decreasing surjective maps. We will denote by $R_{n,m}$ all possible reparameterizations (P,Q) of the trajectories X_n

and X_m.

Now, the discrete Fréchet distance is defined as the minimum over all the reparameterizations of the maximum Euclidean distance between the trajectories[49]:

$$\delta_F^{max}(X_n, X_m) = \min_{(P,Q)\in R_{n,m}} \max_{i\in[0,\gamma_{n,m}]} \|X_{n,p_i} - X_{m,q_i}\|. \tag{4.1}$$

The second distance is a variant of the first, where the L^∞-norm is replaced by a L^1-norm:

$$\delta_F^1(X_n, X_m) = \min_{(P,Q)\in R_{n,m}} \sum_{i=0}^{\gamma_{n,m}} \|X_{n,p_i} - X_{m,q_i}\|. \tag{4.2}$$

To define the third and last distance, let us introduce the maximum trajectory duration over the ensemble of trajectories $(X_n)_{n\in[1,N]}$:

$$T_{\max} = \max_{n\in[1,N]}(T_n). \tag{4.3}$$

For two trajectories X_n and X_m, the stopped process distance is defined as, assuming without loss of generality that $T_n < T_m$:

$$\delta_{\text{stop}}^1(X_n, X_m) = \sum_{t=0}^{T_n} \|X_{n,t} - X_{m,t}\| + \sum_{t=T_n+1}^{T_m} \|X_{n,T_n} - X_{m,t}\| + (T_{\max} - T_m)\|X_{n,T_n} - X_{m,T_m}\|. \tag{4.4}$$

Let us now discuss how to deal with another difficulty when devising clustering techniques over the set of reactive trajectories, namely the high dimension of the systems of interest. Because the computational bottleneck is the computation of the distance, a clustering that uses all the degrees of freedom would be too costly. We therefore perform the analysis over the trajectories projected on a low dimensional space, namely a set of internal variables that describe the transition. Yet, once the clustering is done, there is an interest in knowing the behavior of other degrees of freedom for the representative trajectories. Hence, the Lloyd's algorithm was performed by searching the center of the cells among the ensemble of trajectories. Therefore, the representative trajectories belong to the original data set, and thus has all the dimensions.

The clustering is made using a Lloyd's algorithm, searching to best represent the data set using a fixed number L of Voronoi cells. Those are uniquely described by their centers, which belong to the original ensemble of trajectories. Thus, we searched for the ensemble of L indices of trajectories, called C, that minimizes the sum of distances from the data to its residence cell, as described below:

$$C \in \underset{\substack{S\subset[1,N] \\ |S|=L}}{\operatorname{argmin}} \left\{ \sum_{n=1}^{N} w_n \min_{s\in S} \delta(X_n, X_s)^2 \right\}. \tag{4.5}$$

Here w_n is the weight of trajectory n. The distance δ between two trajectories is either δ_F^{max}, δ_F^1 or δ_{stop}^1.

The minimization is done by giving a guess for the centers, and then iteratively defining the cells and their new centers, until a fixed point is reached, i.e. the centers remain unchanged between two successive iterations. Let us call $C = (c_l)_{1\le l\le L}$ the indices of the centers, and I_l the ensemble of indices of the elements in cell l:

$$I_l = \left\{ n \in \{1,\dots,N\} \;\middle|\; l \in \underset{1\le j\le L}{\operatorname{argmin}} \delta\left(X_n, X_{c_j}\right)^2 \right\}. \tag{4.6}$$

After the elements of each cell are obtained, the centers are updated as the element that is closest to all the other elements from the cell:

$$c_l \in \underset{i\in I_l}{\operatorname{argmin}} \left\{ \sum_{n\in I_l} w_n \delta\left(X_n, X_i\right)^2 \right\}. \tag{4.7}$$

To initialize the algorithm, the first guess is a simple random sample of L indices between 1 and N, giving $C^0 = \left(c_l^0\right)_{1\le l\le L}$. Equation (4.6) is used to define the elements of each cell, i.e. the ensembles $\left(I_l^0\right)_{1\le l\le L}$. New centers $C^1 = \left(c_l^1\right)_{1\le l\le L}$ are computed, using the the elements of each cell, via equation (4.7), that are then used to obtain the cells $\left(I_l^1\right)_{1\le l\le L}$, and so on. The algorithm stops at iteration q if $C^q = C^{q-1}$. This is the so-called Lloyd's algorithm[50, 51].

After the classification is done, the probability of each cell is computed as:

$$\forall\, l \in \{1,\dots,L\},\quad P_l = \frac{\sum\limits_{n\in I_l} w_n}{\sum\limits_{n=1}^{N} w_n}.$$

4.2.2 Clustering over projected trajectories

Let us now make precise the second clustering technique that will be tested[52]. It can be separated into two steps. The first is the projection of the data over a basis, that was done through principal component analysis (PCA) over the centered trajectories, followed by a dimensionality reduction of the problem. The second step is the clustering over the projected data, that was made using one step of the Kohonen's algorithm[53], and then refining the result through the Lloyd's algorithm[50].

To perform a principal component analysis, one needs to build a matrix of the centered trajectories. The first encountered problem is the same as in the last section: the fact that the trajectories do not have the same lengths. This can be solved by considering the trajectories constant from their final timestep T_n to $T_{\max}$ (see equation (4.3)). This however introduces null eigenvalues in the PCA. In order to avoid this issue, a small white noise is added to the stationary part. This noise only influences the smaller eigenvalues, that will be ignored once the dimension is reduced. Let us denote by $\left(Y_{n,t}\right)_{n\in[1,N],t\in[0,T_{\max}]}$ the new set of trajectories defined by:

$$Y_{n,t} = \begin{cases} X_{n,t} & 0 \le t \le T_n \\ \left(1 + \frac{G^n}{1000}\right) X_{n,t} & T_n < t \le T_{\max} \end{cases}, \tag{4.8}$$

where G^n are independent reduced centered Gaussian random variables.

The goal here is to write the data in the basis of the covariance operator. In order to have a covariance matrix, the process needs to be centered. Consider then the centered trajectories $\left(Z_{n,t}\right)_{n\in[1,N],t\in[0,T_{\max}]}$, where $Z_n = Y_n - \overline{Y}$. Here $\overline{Y}$ is the average trajectory:

$$\overline{Y} = \frac{\sum_{n=1}^{N} w_n Y_n}{\sum_{n=1}^{N} w_n}.$$

Calling $Z^i_{n,t}$ the value for the i^{th} coordinate at time t of trajectory n, the matrix of data $\mathbf{M}$, of size $d(T_{\max}+1) \times N$, is constructed as:

$$\mathbf{M} = \begin{bmatrix} Z^1_{1,0} & \cdots & Z^1_{N,0} \\ \vdots & & \vdots \\ Z^1_{1,T_{\max}} & \cdots & Z^1_{N,T_{\max}} \\ \vdots & & \vdots \\ Z^d_{1,0} & \cdots & Z^d_{N,0} \\ \vdots & & \vdots \\ Z^d_{1,T_{\max}} & \cdots & Z^d_{N,T_{\max}} \end{bmatrix}, \tag{4.9}$$

Let us call $\mathbf{W} \in \mathbb{R}^{N\times N}$ the diagonal matrix of the AMS weights of the trajectories, i.e. $\mathbf{W}_{n,n} = w_n$. The principal component analysis is done by diagonalising the symmetric matrix $\mathbf{C} = \mathbf{M}\mathbf{W}\mathbf{M}^{\intercal}$. Notice that the covariance weighted matrix is of size $d(T_{\max}+1) \times d(T_{\max}+1)$. Hence, the PCA being the computational bottleneck of this method, its computational cost will only depend on the maximum size of the trajectories and the dimension d, but not their number.

The diagonalisation procedure gives λ, the vector of ordered eigenvalues, and $\mathbf{U}$, the matrix whose lines are the associated eigenvectors. In order to reduce the dimension of the problem, only the N_{dim} first eigenvectors are used, creating the reduced matrix $\mathbf{U}_{\text{dim}}$. The choice of N_{dim} is further discussed in the results section. Let us denote by $\mathbf{\Lambda}_{\text{dim}}$ the diagonal matrix of the square root of the first N_{dim} eigenvalues. The projection of the data over this reduced basis gives the new data matrix $\mathbf{V}$, of size $N_{\text{dim}} \times N$, as:

$$\mathbf{V} = \mathbf{\Lambda}^{-1}_{\text{dim}} \mathbf{U}_{\text{dim}} \mathbf{M}.$$

With this data transformation, the n^{th} column of $\mathbf{V}$, called V_n, corresponds to the n^{th} projected trajectory.

Once the data is transformed, the second step is to perform the quantization with a fixed number of L cells. This means the data $\mathbf{V}$ will be represented by L vectors of size N_{dim}, with an associated probability. The data space is described by Voronoi cells, and thus a definition of distance is needed. Let us denote by $\|\cdot\|_\lambda$ the norm over the reduced space, defined by:

$$\|V_n\|^2_\lambda = \sum_{i=1}^{N_{\text{dim}}} \lambda_i \|V_{n,i}\|^2.$$

The clustering is done by making a first extensive search using one step of the Kohonen's algorithm[53], and then performing Lloyd's algorithm[50] until convergence is reached. The quantization error we aim to minimize with this two step procedure is the sum of the distances from the points of a cell to its center. Denoting by I_l the ensemble of indices of trajectories from cell l, those are defined as:

$$\{I_1,\dots,I_L\} \in \underset{\substack{\{S_1,\dots,S_L\} \\ \bigcup_{l=1}^{L} S_l=[1,N] \\ \forall i\neq j, S_i\cap S_j=\emptyset}}{\operatorname{argmin}} \left\{ \sum_{l=1}^{L}\sum_{n\in S_l} w_n \|V_n-\tilde{V}_l\|_\lambda^2 \,\middle|\, \tilde{V}_l=\frac{\sum_{n\in S_l} w_n V_n}{\sum_{n\in S_l} w_n} \right\}. \tag{4.10}$$

Notice that the centroids are vectors of size N_{dim}, and are denoted by $\big(\tilde{V}_l\big)_{l\in[1,L]}$.

One can now use the original data to define the centroids using all the dimensions. Indeed, each center of a cell is a linear combination of vectors that corresponds to the trajectories projected on a basis. The same procedure can be done using the original data from matrix **M**. Then, the centroids of the original Y process are defined by adding the mean trajectory, as:

$$\tilde{Y}_l=\frac{\sum_{n\in I_l} w_n Z_n}{\sum_{n\in I_l} w_n}+\overline{Y}.$$

The error made within this quantization procedure, called the distortion, can be computed as the sum of two terms. The first source of error is the dimensionality reduction. The second one is the error made by describing the data by a few Voronoi cells. This gives:

$$\mathbb{E}\big(\|Y-\tilde{Y}\|_{L^2}^2\big)=\sum_{n=N_{\text{dim}}+1}^{dT_{max}}\lambda_n+\mathbb{E}\big(\|V-\tilde{V}\|_\lambda^2\big).$$

4.3 Results

Both methods were applied to two different systems. The first is a potential in 2D and the second is our molecular toy model, alanine dipeptide.

4.3.1 Double channel 2D potential

For the second numerical experience, we used the overdamped Langevin dynamics with a potential in two dimensions.

$$dX_t=-\nabla V(X_t)dt+\sqrt{2\beta^{-1}}dW_t, X_t\in\mathbb{R}^2$$

In this equation $\beta^{-1}=k_B T$. The numerical solution was obtained through the explicit Euler-Maruyama scheme:

$$\forall n\in\mathbb{N}, X_{n+1}=X_n-\nabla V(X_n)h+\sqrt{2h\beta^{-1}}G_n, \tag{4.11}$$

where $(G_n)_{n\in\mathbb{N}}$ are i.i.d. centered Gaussian random vectors in $\mathbb{R}^2$ with identity co-variance matrix. The timestep h is 0.01.

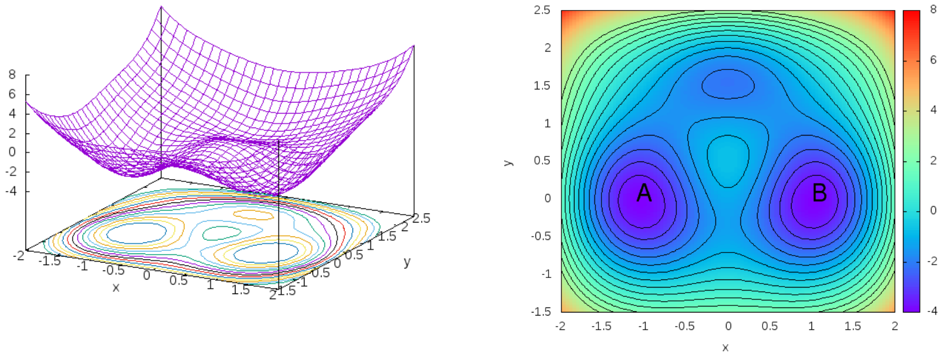


Figure 4.1 – The 2D double channel potential. Zones A and B are defined as the regions were the potential is lower than -3.5.

The 2D potential function V is given by[38, 54, 55]:

$$V(x,y) = 3e^{-x^2-\left(y-\frac{1}{3}\right)^2} - 3e^{-x^2-\left(y-\frac{5}{3}\right)^2} - 5e^{-(x-1)^2-y^2} - 5e^{-(x+1)^2-y^2} + \frac{x^4+\left(y-\frac{1}{3}\right)^4}{5}. \tag{4.12}$$

The potential landscape is symmetric with respect to the y-axis and has three wells, where one is less deep than the others (see figure 4.1). We then consider the two most significant wells as states A and B, defined as:

$$\begin{aligned} A &= x \in (-\infty, 0] \cap \{(x,y) | V(x,y) < -3.5\} \\ B &= x \in [0, +\infty) \cap \{(x,y) | V(x,y) < -3.5\} \end{aligned} \tag{4.13}$$

The two saddle-points around $(x, y) = (\pm 0.7, 1)$ are lower in energy that the saddle-point around $(x, y) = (0, -0.4)$. There are two different ways to transition from A to B. The first transition goes through the shallow well around $(x, y) = (0, 1.5)$, and the second one is directly crossing the higher energy barrier at the bottom. Notice that the latter channel goes through a higher saddle point, and will thus be preferred only if the temperature is high. For small temperature, the particle will preferably take the path through the upper shallow well. Thus, the reactive trajectories can be separated into two clusters, that will represent either the upper or lower path, whose probabilities will depend on the temperature.

To generate the reactive trajectory ensemble, the AMS method was used. A total of a thousand AMS simulation were run using 100 replicas, and the following definition for the reaction coordinate.

$$\xi(x,y) = \begin{cases} |(x,y) - (-1,0)| - |(x,y) - (1,0)| & x < 0 \\ 5 - |(x,y) - (1,0)| & x \geq 0 \end{cases} \tag{4.14}$$

The parameter k was set to unity and the last level z_{max} as 4.6. To see the effect of the temperature, two different values were used for the parameter β, 1.67 and 6.67. The trajectories that crosses $x = 0$ for the first time at $y < 0.5$ are considered to be in the bottom pathway, and the others in the top. Using this definition, when $\beta = 1.67$, 35.9 % of the trajectories are in the upper channel. For $\beta = 6.67$, the upper channel represents 55.1 % of the trajectories. These results are in accordance with previous computations made in [38].

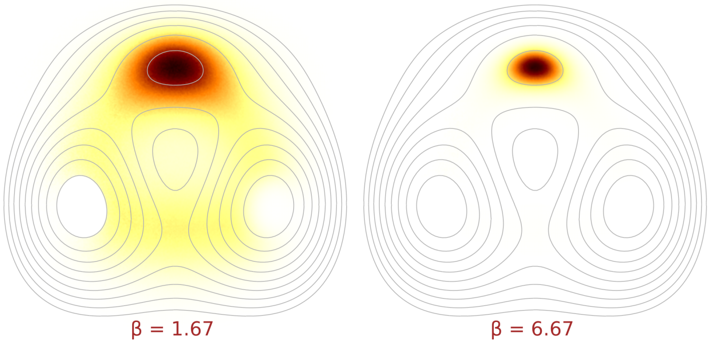


Figure 4.2 – Density of the reactive trajectories obtained with AMS.

Figure 4.2 shows the density of reactive trajectories for the two values of β. The density for $\beta = 1.67$ shows the two possible reaction paths. Notice that, for the smaller temperature ($\beta = 6.67$), most of the time is spent in the upper shallow minimum.

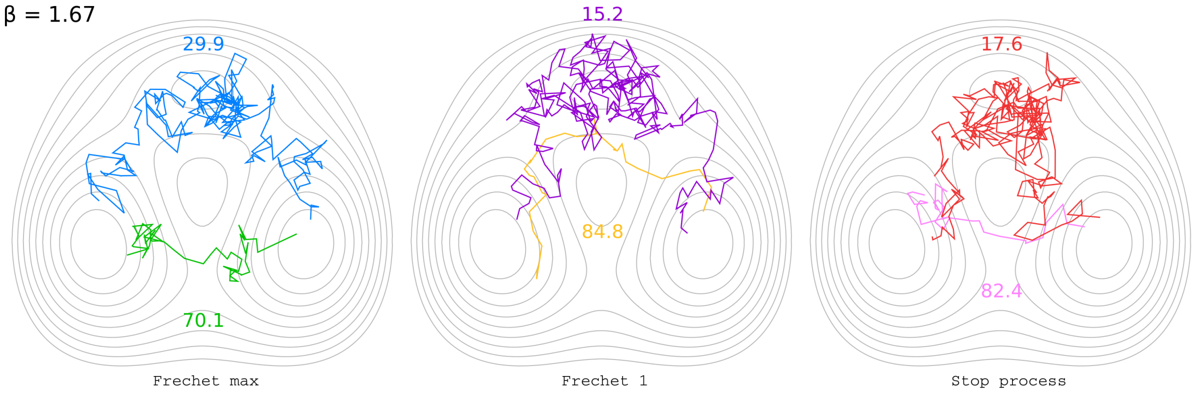


Figure 4.3 – Results for the clustering over the original data with $\beta = 1.67$ and for different choices of distance between paths.

Figure 4.3 shows the representative trajectories obtained with the clustering over the original data with $\beta = 1.67$. The trajectories obtained with the Fréchet max distance are consistent with the two mechanisms, and with the probabilities of reference. The second distance fails to separate the mechanisms. The stop process distance gives a larger proportion for the trajectories that passes by the bottom path.

For $\beta = 6.67$, the results for the clustering over the original data is presented in figure 4.4. The results with Fréchet max are consistent with the reference, with consistent proportions. Both Fréchet 1 and stop process distances fails to separate the two mechanisms. The trajectories seems to have been separated by their duration, and not by the region of the space they explore. Notice from the definition of those distances (see (4.1), (4.2) and (4.4)) that both Fréchet 1 and stop process distances depend on the size of the trajectories, but Fréchet max does not. Because at a low temperature the duration of the trajectories varies more than at a higher temperature, the distances δ_F^1 and δ_{stop}^1 are not able to separate the trajectories passing through the two channels.

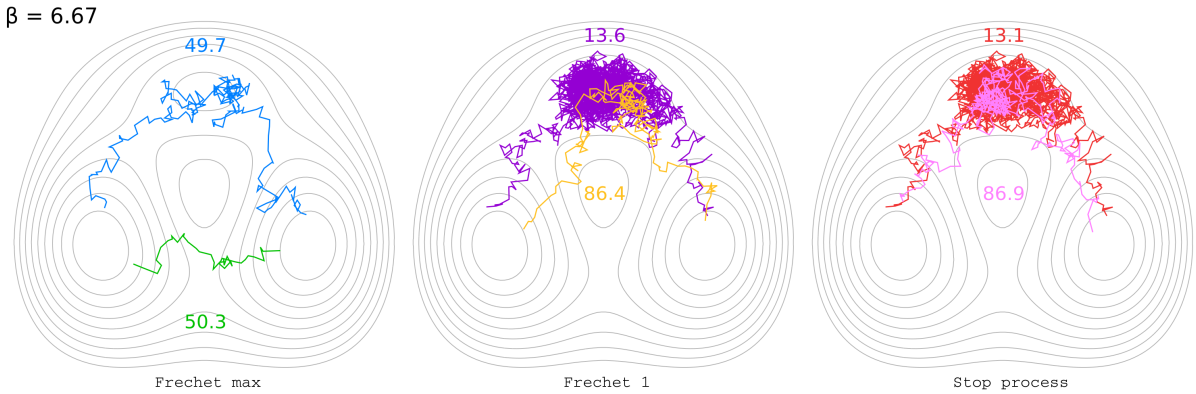


Figure 4.4 – Results for the clustering over the original data with $\beta = 6.67$.

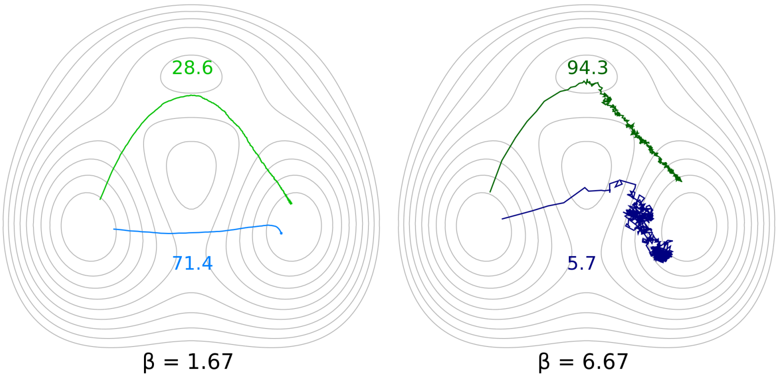


Figure 4.5 – Representative trajectories found with the clustering done over the projected data.

The results for the clustering using the projected data is presented in figure 4.5. The trajectories and weights for $\beta = 1.67$ are consistent with the results found with the other technique, and also the reference. For $\beta = 6.67$, the method fails to predict the correct probabilities. This last result requires more investigation for better understanding.

4.3.2 Alanine Dipeptide conformational change

The alanine dipeptide is a small biomolecule that has two stable conformations that can be defined by two dihedral angles φ and ψ (see figure 4.6). The graph on figure 4.6 shows the reaction coordinate used and also the definitions of states A and B. The dynamics was run using the NAMD program[10]. Two sets of AMS simulations were run using two different initial conditions.

Figure 4.7 shows the obtained flux of trajectories (see Chapter 2, equation (21)). The flux maps suggest the presence of one reaction mechanism for the first initial condition, and two for the second.

In order to determine the number of mechanisms, the two clustering techniques described above were applied using different number of cells. Figure 4.8 shows the results for the clustering over the projected trajectories, using from one to three cells, and dimension one. It is important to mention that

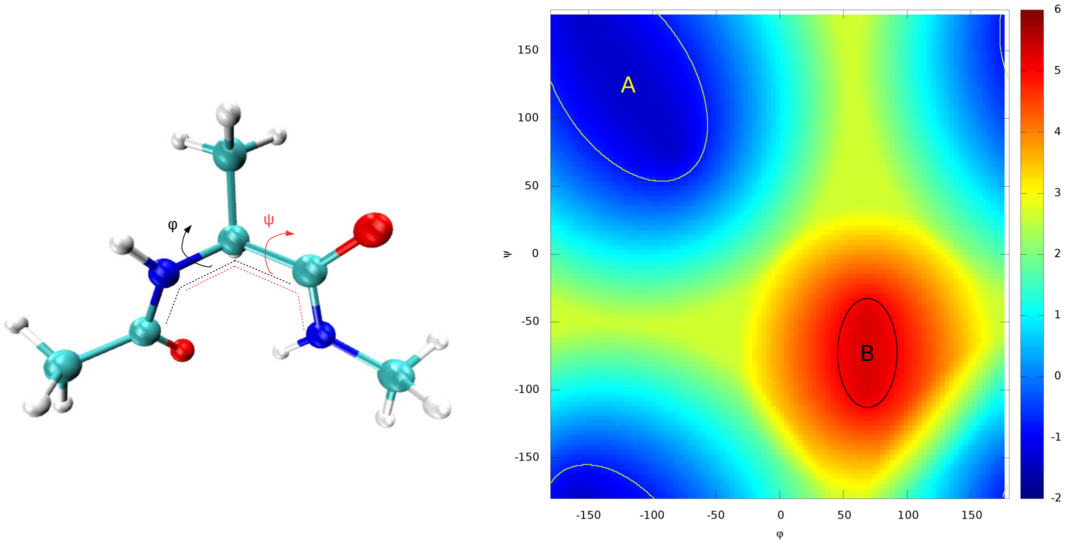


Figure 4.6 – Left: the alanine dipeptide molecule and the dihedral angles used to distinguish the two stable conformations. Right: the reaction coordinate and the definitions of states *A* and *B*.

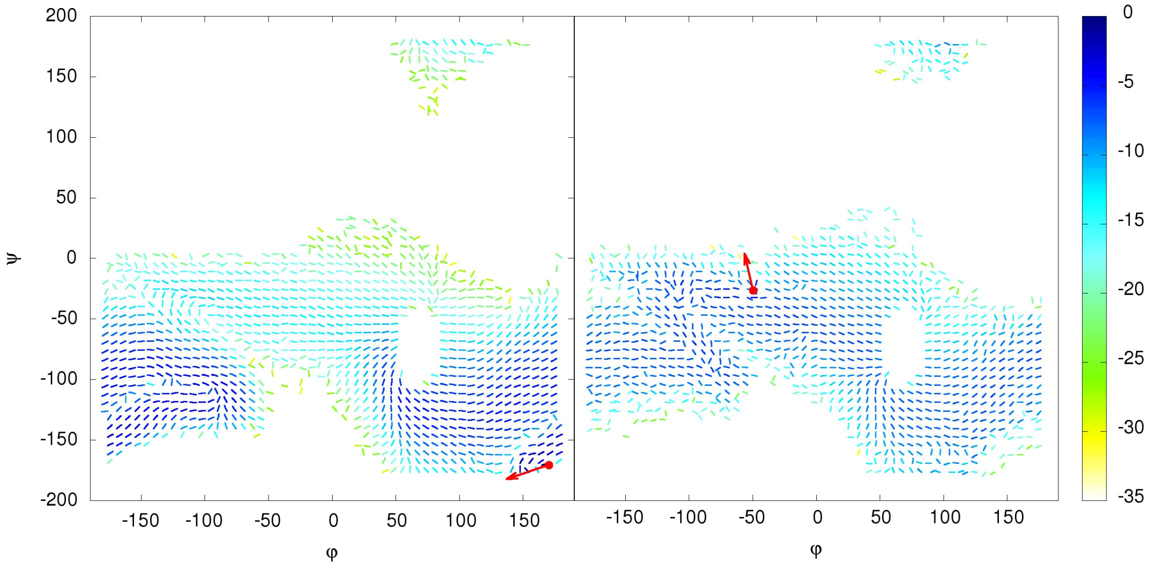


Figure 4.7 – Flux of reactive trajectories obtained with two different initial conditions, where the positions and velocities are fixed for all atoms, whose projection is given by the red vectors.

the results using two dimensions are equivalent to those with one dimension, and therefore the second dimension is not needed to correctly distinguish the trajectories of this system. For the first initial condition, the clustering with one cell gives a representative trajectory compatible with the result for the flux. For two cells the clustering finds another path that has a low probability of occurrence, which

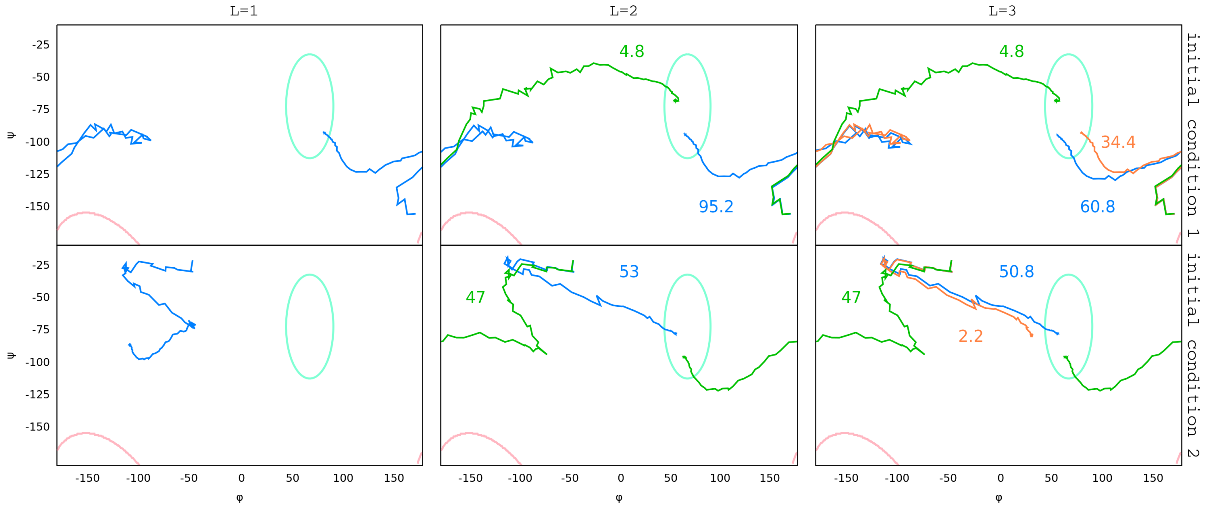


Figure 4.8 – Representative trajectories for the clustering over the projected data for the two initial conditions and $L \in \{1,2,3\}$.

is consistent with the flux map. With three cells, the trajectory with larger weight is divided into two that represent the same transition mechanism. This indicates the presence of only two mechanisms. For the second initial point, the clustering with one cell gives a trajectory that is not reactive, and thus can not be considered as a representation of the data ensemble. With two cells, two trajectories consistent with the flux map were found. The most probable cluster is split into two when using three cells, indicating again the presence of two reaction mechanisms.

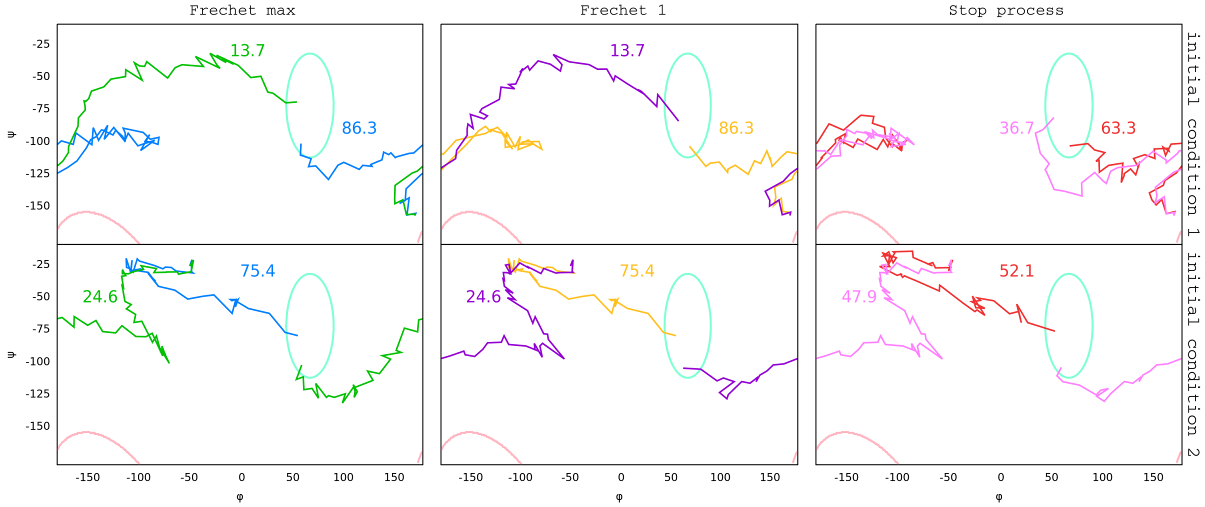


Figure 4.9 – Representative trajectories obtained for the clustering over the original data, using the three different definitions for the distance.

Figure 4.9 shows the results obtained when performing the cluster over the original trajectories for the three different distances. The results are consistent with the ones found with the previous clustering technique, but the weights are different, and only the stop process distance failed to find the trajectory of smaller probability for the first initial condition. The two Fréchet based distances predicted the same weights.

4.3.3 Conclusion and Perspectives

The two clustering techniques show the capacity to find the possible transition mechanisms. When performing the clustering over the original data, the Fréchet max distance gives results which are more consistent with what is expected.

The clustering over the projected data give reliable results, and demands less computational cost than the other technique. However, there is an additional parameter to choose, namely the reduced dimension, which will depend on the system. Also, the final trajectories does not belong to the original data ensemble. One could however apply this clustering method over the projected data, but imposing the centers of the Voronoi cells to be among the ensemble, as in the first clustering method. We intent to test this variant in a near future.

Part II

Applications

Chapter 5

AMS tutorial for NAMD

This chapter contains the written AMS tutorial for NAMD (2018).

Adaptive Multilevel Splitting Method: Isomerization of the alanine dipeptide

Laura J. S. Lopes, Christopher G. Mayne, Christophe Chipot, Tony Lelièvre
CERMICS, École des Ponts ParisTech, Université de Lorraine, University of Illinois at Urbana-Champaign

In this tutorial, we show how to apply the Adaptive Multilevel Splitting (AMS) method to the isomerization of the alanine dipeptide in vacuum. Section 5.1 gives a description of the AMS algorithm and the proper way to set up AMS simulations, in the context of the NAMD program, for any system using the scripts provided with this document. An application of the method to the case example is showcased in section 5.2. More precisely, we show how to obtain the transition probability starting from one fixed point (Section 5.2.2), and the transition time (Section 5.2.3). Using the results obtained in these sections, a description of how to calculate the flux of reactive trajectories is given in Section 5.2.4. Results of the simulations for sections 5.2.2 and 5.2.3 are provided, so that the reader can straightforwardly go to Section 5.2.4 if desired.

Completion of this tutorial requires:

- Files from `AMS_tutorial.zip` provided with this document
- NAMD version 2.10 or later
- Optional: Gnuplot

5.1 The Adaptive Multilevel Splitting method

The Adaptive Multilevel Splitting (AMS) method is a splitting method to sample reactive trajectories [21, 38, 56]. The goal here is to accelerate the transition between metastable states, which are regions of the phase space where the system tends to stay trapped. This method is particularly interesting because the positions of the intermediate interfaces, used to split reactive trajectories, are adapted on the fly, so they are not parameters of the algorithm. The AMS method was already efficiently applied to a large scale system to calculate unbinding time [11].

Section 5.1.1 presents the AMS algorithm as implemented in the Tcl script `ams.tcl`, provided with this document. In section 5.1.2 we show how to set up AMS simulations for any system.

5.1.1 The AMS algorithm

Let us call A and B the source and target regions of interest, and assume that A is a metastable state. This means that starting from a point in the neighborhood of A, the trajectory is most likely to enter A before visiting B. The goal is to sample reaction trajectories that link A and B. In practice, these regions are defined using a set of internal variables of the system.

To compute the progress from A to B one needs to introduce a reaction coordinate ξ. Again, in practice ξ is a real-valued function of internal variables of the system. This function only needs to satisfy one condition: it is necessary that there exists a value of ξ that the system has to exceed to enter B when starting from A. This value of ξ is called $z_{\max}$. Note that the definitions of the zones A and B are independent of the reaction coordinate. Since ξ does not need to be continuous, the former condition can be enforced by making ξ equal to infinity on B. The condition is then satisfied with $z_{\max}$ equal to the maximum value of $\tilde{\xi}$ outside B. In practice, the easiest way is to make ξ equal to $z_{\max}+1$ inside B.

The algorithm, as presented below, estimates the probability to observe a reaction trajectory, that is, coming from a set of initial conditions in a neighborhood of A, the probability to enter B before returning to A. We will denoted this estimator by p_{AMS}. This probability can be used to compute transition times and we will see how in Section 5.2.3.

The three numerical parameters of the algorithm are: (1) the reaction coordinate ξ, (2) the total number of replicas N, and (3) the minimum number k of replicas killed at each iteration. The algorithm starts at iteration $q = 0$ and follows the flowchart below (see also Figure 1 for a schematic representation).

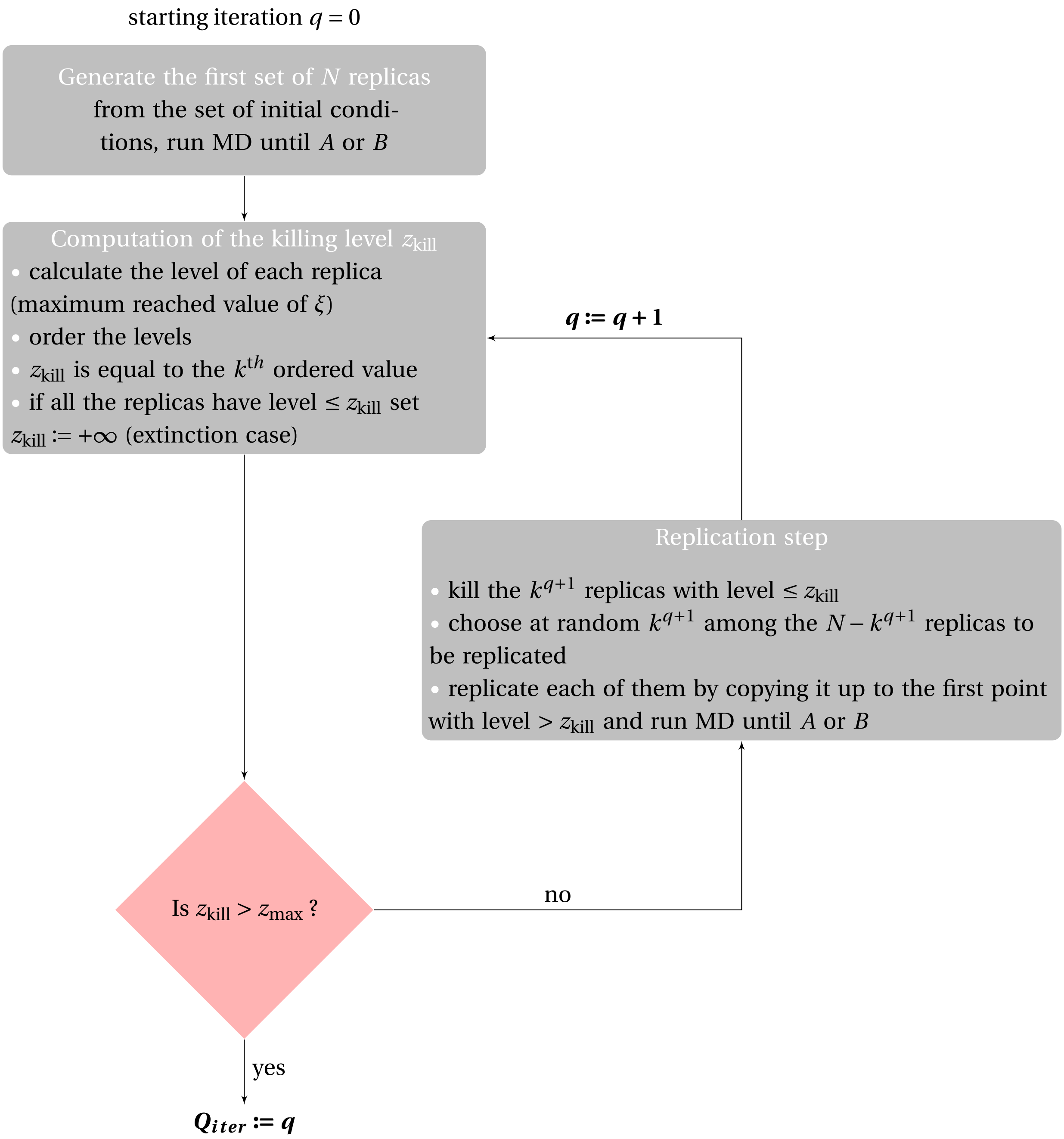
Notice that at the end of iteration q, the estimation of the probability of reaching level z_{kill}^{q}, conditioned to the fact that level z_{kill}^{q-1} has been reached, (where by convention $z_{\text{kill}}^{-1} = -\infty$) is:

$$p^{q} = \frac{N - k^{q+1}}{N}. \tag{5.1}$$

Therefore, denoting r the number of replicas that reached B at the last iteration of the algorithm, the estimator of the probability of transition is:

$$p_{\text{AMS}} = \frac{r}{N} \prod_{q=1}^{Q_{iter}} p^{q-1} = \frac{r}{N} \prod_{q=1}^{Q_{iter}} \left(\frac{N - k^{q}}{N} \right), \tag{5.2}$$

where by convention $\prod\limits_{q=1}^{0} = 1$. For example, if all the replicas in the initial set reached B, $r = N$ and, thus, $p_{\text{AMS}} = 1$. In case of extinction $r = 0$, because no replica reached B, and thus $p_{\text{AMS}} = 0$.


starting iteration $q = 0$
Generate the first set of N replicas from the set of initial conditions, run MD until A or B
Computation of the killing level z_{kill}
• calculate the level of each replica (maximum reached value of ξ)
• order the levels
• z_{kill} is equal to the k^{th} ordered value
• if all the replicas have level $\leq z_{\text{kill}}$ set $z_{\text{kill}} := +\infty$ (extinction case)
$q := q + 1$
Replication step
• kill the k^{q+1} replicas with level $\leq z_{\text{kill}}$
• choose at random k^{q+1} among the $N - k^{q+1}$ replicas to be replicated
• replicate each of them by copying it up to the first point with level $> z_{\text{kill}}$ and run MD until A or B
Is $z_{\text{kill}} > z_{\text{max}}$?
no
yes
$Q_{iter} := q$

As the algorithm runs, all the points that can possibly be used in future replication steps must be recorded. In order to decrease the computational cost and the memory use, this is only done every $K_{\text{AMS}} = \Delta t_{\text{AMS}}/\Delta t$ timesteps. It is indeed useless to consider the positions of the trajectory at each simulation time step, as no significant change occurs over a 1 or 2 fs timescale.

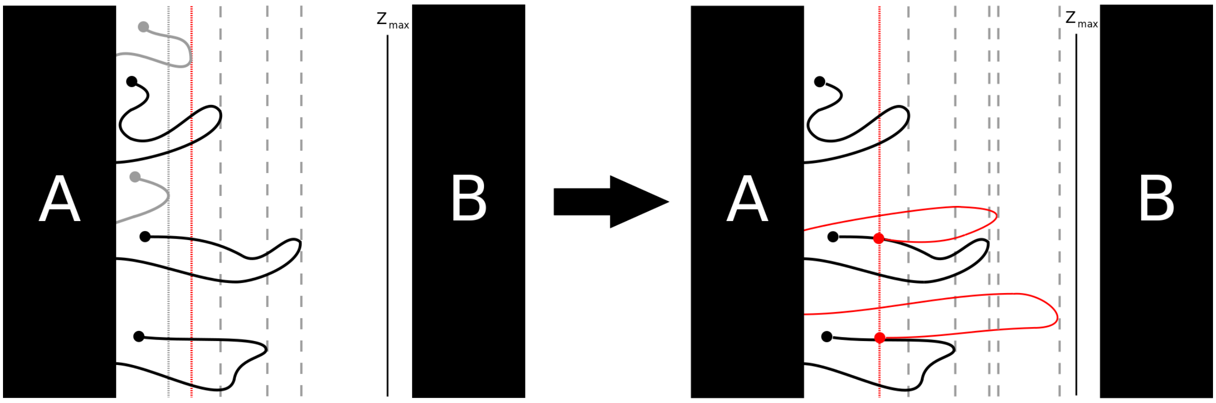


Figure 5.1 – First AMS iteration with $N = 5$ and $k = 2$. Both lower level replicas (in gray) are killed. Two of the remaining replicas are randomly selected to be duplicated until level z^0_{kill} (red line) and then continued until they reach A (typically more likely) or B.

5.1.2 Setting up AMS simulations

The AMS method is implemented in NAMD through a Tcl script sourced by the user in the configuration file, where the AMS functions are directly called. To run the algorithm the user must provide a set of files (see Section 5.1.2), and should have run the first set of replicas. A set of Tcl and bash scripts and simple C programs are provided to automate the process. As will be seen in the Section 5.1.2, the user will only need to provide one input file.

All the scripts and programs needed to run an AMS simulation can be found in the `smart` directory. The algorithm implementation is located in file `ams.tcl`. A script called `smart_parallel.sh` automates all the AMS runs, and it is the only script that the user will call directly. To utilize this script it is necessary to set a few variables.

1. Open script `smart/toall_path.sh` to edit it.
2. Set the variable `smart_path` as the path to all the smart files (i.e. to directory `smart`)
3. Set the variable `amsscript` with the `ams.tcl` script location.
4. Provide the NAMD executable file location through variable `namd`.
5. Close file `toall_path.sh`.
6. Open the terminal and type:
   ```
   export toall_smart="/path/to/tutorial/files/smart/toall_smart.sh"
   ```
   The export command not only defines a variable but makes its value visible for all the scripts that will be run in this same terminal session. To make it visible for all the sessions, just include this line into `/home/user/.bashrc` file.
7. Open file `common/namd.conf` to edit.
8. Set the variable `path` to the path to directory `common`.

The user files to provide

In addition to the basic NAMD files to run MD, it is necessary to provide a group of additional files to set up the AMS simulations. Some of these are Tcl scripts that should contains the definitions of a few procedures that will be called by the AMS Tcl script. If the reader is not familiar with Tcl language, procedure is the equivalent of function in Tcl. Nevertheless, it is not necessary to program in Tcl for this tutorial, as all these files are given in the `common` directory. The additional files in the `common` directory are:

- `dihedral_20.colv`: a Colvars [12] configuration file with the definition of the collective variables that will be used to calculate the regions A and B and the reaction coordinate ξ;
- `inzone.tcl`: a Tcl script with a procedure called `zone` that should return -1 if in region A, 1 if in B and 0 otherwise, using a set of the collective variables defined in the previous file;
- `coord.tcl`: a Tcl script with a procedure called `ams_measure` that has to return the value of the reaction coordinate ξ (also using the collective variables);
- `variables.tcl`: a Tcl script with a procedure called `variables` that returns a list of internal coordinates values used to visualize the reactive trajectories after the AMS run. In the case of alanine dipeptide this script only prints the collective variables defined in `dihedral_20.colv`. This scripts introduces flexibility to the analysis of the reactive trajectories, that can be made using different internal coordinates as the ones used to define A, B and ξ.
- `namd.conf`: a typical NAMD configuration file without any run step that will be the base to build all the NAMD configuration files for the AMS simulations.

9. Open file `common/inzone.tcl` and set the correct path to the executable file `zones_CRI`.
10. Do the same with file `common/coord.tcl` for the path to `coord_CRI`.

Preparing an input file

The `smart_parallel.sh` is the only script that the user will call. This script only needs one simple bash file as an entry, that should define the following variables:

- `initfile`: name of NAMD basic configuration file (`namd.conf` in this tutorial)
- `numinst`: number of AMS instances, i.e. the number of requested AMS runs.
- `outdir`: root directory to save all the AMS instances directories. If this directory exists, the script will look for results from previous run and will perform the missing AMS runs to complete `numinst` simulations.
- `parallel`: number of AMS instances that can be run in parallel.

- numrep: number of replicas for each AMS run (parameter N)
- amstype: single (all the replicas are initiated from the same point), mult (replicas are initiated from a set of numrep points) or var (the initial conditions will vary)
- zmin: minimum value for the reaction coordinate (only used if amstype == var)
- zmax: maximum value of ξ ($z_{\max}$)
- timelimit: simulation time limit in hours. This time limit prevents AMS runs from being abruptly killed if its duration exceeds a time limit from a queue system, and facilitates subsequent restart of the simulations afterwards.
- icprefix: prefix for coor, vel and xsc files from initial condition. If amstype == mult the files have to be named prefix.n (were $n = 0, ..., \texttt{numrep} - 1$).
- zone: name of Tcl script that contains the procedure zone (see inzone.tcl from the previous list).
- measure: name of Tcl script with the definition of the procedure ams_measure (see coord.tcl).
- variables: name of Tcl script with procedure variables (see variables.tcl).
- amssteptime: number of time steps between two computations of the reaction coordinate (this is the parameter K_{AMS} mentioned above).
- tokill: minimal number of replicas to kill at each iteration (parameter k)
- getpaths: on or off. If this variable is set to on, all the sampled trajectories will be given in text format files, built using the variables proc.
- charmrunp: number of processors to employ for the MD (if 0 the command will be namd2)
- removefiles: yes or no. If this variable is set to yes, all the AMS files will be removed after the run. If getpaths == on, all the trajectories will be obtained and will not be erased. Attention, if getpaths == off, and removefiles == yes it will be impossible to obtain the trajectories after the run.

5.2 Applying AMS to the alanine dipeptide isomerization in vacuum

We chose the alanine dipeptide isomerization in vacuum ($C_{eq} \rightarrow C_{ax}$ transition) to illustrate how to utilize the AMS method. The reader will find the precise definitions of regions A and B and of the reaction coordinate ξ in Section 5.2.1. The hands-on part of the tutorial starts in Section 5.2.2, where we show how to obtain the transition probability, starting all the replicas from the same point. In Section 5.2.3 the theoretical underpinnings of the equation used to calculate the transition time using AMS results is given, as well as the guidelines as how to set up these simulations. Finally, armed with the results obtained in Sections 5.2.2 and 5.2.3 we will calculate the flux of reactive trajectories in Section 5.2.4.

5.2.1 Definitions of A, B and ξ

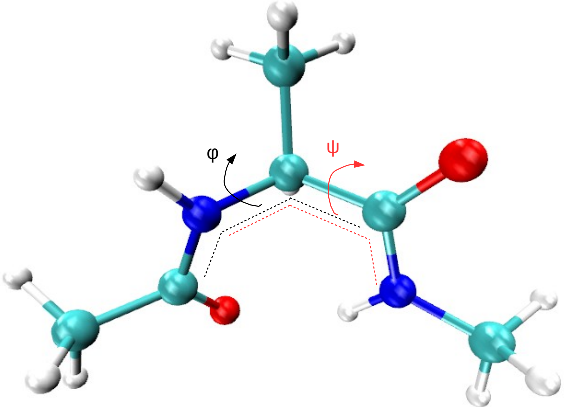


Figure 5.2 – The dihedral angles φ and ψ used to distinguish between the C_{eq} and C_{ax} conformations.

All the definitions will be based on the two dihedral angles φ and ψ (see Figure 5.2). The regions A and B are defined as two ellipses that cover the most significant wells on the free energy landscape. The reaction coordinate is a measure of the distances from the two ellipses:

$$\xi(\varphi,\psi) = \min(d_A, 6.4) - \min(d_B, 3.8). \tag{5.3}$$

In equation (5.3), d_A (resp. d_B) is the sum of the Euclidean distances to the foci of the ellipse A (resp. B). The contour plot of the function ξ is given on Figure 5.4. We will employ $z_{\max} = 4.9$ in our simulations.

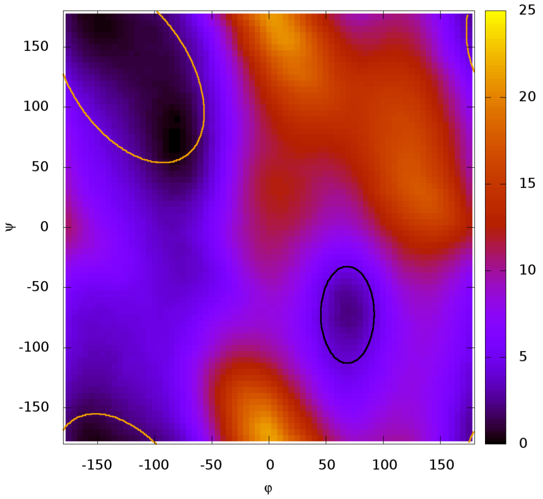


Figure 5.3 – The free energy landscape [14] with the definition of zones A (yellow) and B (black).

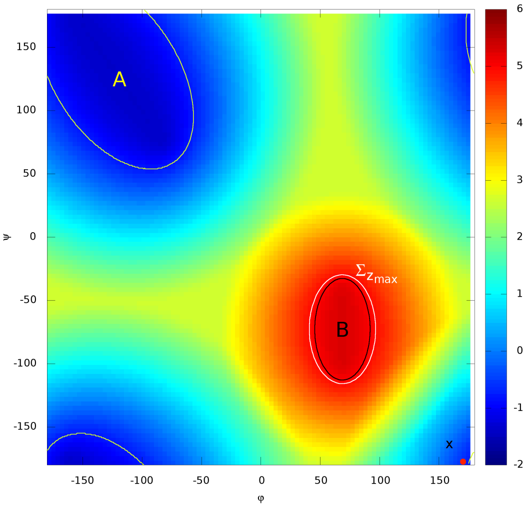


Figure 5.4 – Contour plot of ξ, with regions A and B and the surface $\Sigma_{z_{\max}}$ ($z_{\max} = 4.9$).

5.2.2 Calculating the probability with AMS

In this section, we will see how to compute the probability to enter B before A, starting from a single fixed point. All the necessary files are located inside the directory `1-point`. For these simulations, the first set of replicas are trajectories that starts in a fixed point and finishes at A or B. This is indicated to the script `smart_parallel.sh` through the variable `amstype`, that should be set to `single`. We will start the replicas from the extended system and binary coordinate and velocity files with the prefix `point`.

1. Open the unfinished input file `1-point/point.par` to edit.
2. Set the auxiliary variable `path` with the path to all the tutorial files.
3. Set `amstype` to `single`. This means a simulation where all the replicas start from one single point.
4. Use the variable `icprefix` to give the prefix of the files from the starting point (`point` in this tutorial).

The results of this section will be used to calculate the flux in section 5.2.4. To obtain a net flux it is necessary to have approximately 9000 trajectories, and thus we need that $\texttt{numinst} \times \texttt{numrep} > 9000$. This is because in the case of extinction, no reactive trajectory will be sampled, so we overestimate the number of trajectories. In this tutorial, $\texttt{numinst} = 100$ and $\texttt{numrep} = 100$. We also have to tell our script to get the final trajectories, invited to so we can calculate the flux via the `getpaths` variable. If the reader is not interested in completing this tutorial and only want the calculation of the probability, set this variable to `off`.

5. Set `numinst = 100`.
6. Set `numrep = 100`.
7. Set the variable `getpaths` to `on`.

There is no special interest in obtaining the `dcd` trajectories for the alanine dipeptide case. Thus, to decrease disk space usage, all the files will be deleted at the end of the run.

8. Set the variable `removefiles` to `yes`.

The alanine dipeptide in vacuum is a really small system, so it is not necessary to run MD in parallel. However, we recommend running the AMS simulations in parallel and this should be adapted to the computer architecture at hand. Please, keep in mind that each one of the AMS simulations of this section takes about five minutes to complete (using 100 replicas). So a good estimation of the total time in minutes needed to complete all the 100 runs is:

$$\text{total time} = 5 \times \frac{\texttt{numinst}}{\texttt{parallel}}.$$

For example, using notebook with Intel core i7 processor, one can utilize `parallel = 8`, so the total time will be about one hour.

9. Set `charmrunp` to 0.

10. Set the `parallel` variable to the number of cores at hand.

Now, the final input file should look like this:

```
path="/path/to/tutorial/files"
outdir=$path"/1-point/ams"
tokill="1"
amstype="single"
numinst="100"
numrep="100"
zmax="4.90"
timelimit="240"
icprefix=$path"/1-point/point"
zone=$path"/common/inzone.tcl"
measure=$path"/common/coord.tcl"
variables=$path"/common/variables.tcl"
initfile=$path"/common/namd.conf"
amssteptime="20"
parallel="8"
getpaths="on"
charmrunp="0"
removefiles="yes"
```

Notice here that we are using `amssteptime = 20`. If the reader is guiding himself through this tutorial to run simulations with another system, be careful when choosing this parameter. First, using `amssteptime = 1` is always an option, but this will make the simulations slow. Second, if `amssteptime > 1`, it is necessary to satisfy one important condition: it should be small enough, so that if the system passes through A, at least one point inside A will be computed. Thus, we recommend to run a small preliminary simulation to evaluate the mean time the system stays inside A.

11. Run the script:
    ```
    ../smart/smart_parallel.sh point.par
    ```

Running the script will block the screen showing what instance has already been launched. At the end of the run the probability estimation is given, as well as the total wall clock time spent, and other four files with the same name as the `outdir` variable, followed by:

- `cputime`: list of the CPU time of each AMS run
- `runtime`: same but with the wall-clock times

- `proba`: list of estimated probabilities
- `T3`: list of MD steps of the sampled reaction trajectories. This will be used in section 5.2.3.

The `smart` directory contains an executable file named `media`. The argument for this program is a file with numbers in one column, and their average value and standard deviation will be computed.

12. To see the final estimated value for the probability, type:
```
../smart/media ams.proba
```

Compare the obtained result to the reference DNS value: $(2.076 \pm 0.357) \times 10^{-4}$.

Performing the simulation in this section using smaller values for `numrep` and/or `numinst` leads to a larger confidence interval. If the reader wish to make it smaller, it is possible to run the script again with a larger value of `numinst`. The script will not overwrite the previous results; instead it will run the remaining instances to complete the `numinst` AMS runs.

5.2.3 Obtaining the transition time using AMS results

As already mentioned, it is possible to calculate the transition time using the probability obtained with AMS by using a specific set of initial conditions, which we will now see how to obtain.

The transition time is the average time of the trajectories, coming from B, from its first entrance in A, until the first subsequent entrance in B [15, 16]. As A is metastable, the dynamics tends to make loops between A and its neighborhood before visiting B. To correctly define those loops, let us fix an intermediate value z_{min} of the reaction coordinate, defining a surface $\Sigma_{z_{\text{min}}}$ that corresponds to the region in which ξ is equal to z_{min}.

If A is metastable and $\Sigma_{z_{\text{min}}}$ is close to A, the number of loops made between A and $\Sigma_{z_{\text{min}}}$ before visiting B will then large. After going through some of them, the system reaches an equilibrium. When this equilibrium is reached, the first hits of $\Sigma_{z_{\text{min}}}$ follow a so-called quasi-stationary distribution μ_{QSD}. Here, we call the first hitting points of $\Sigma_{z_{\text{min}}}$ the first points that, coming from A, have a ξ-value larger than z_{min}. Using as an initial condition points distributed according to μ_{QSD}, it is possible to evaluate the probability p to reach B before A, starting from $\Sigma_{z_{\text{min}}}$ at equilibrium with AMS. As A is metastable, the number of loops needed to reach the equilibrium will be small compared to the total number of loops followed before going to B. Thus, the time spent to reach the equilibrium can be neglected.

Let us now consider an equilibrium trajectory coming from B that enters A and returns to B. The goal is to calculate the average time $(\mathbb{E}(T_{AB}))$ of this trajectory. A good strategy is to split this path in two: the loops between A and $\Sigma_{z_{\text{min}}}$, and the reaction trajectory, i.e. the path from A to B that does not come back to A after reaching $\Sigma_{z_{\text{min}}}$ [16]. Neglecting the first time taken to go out of A, one can define as T^k_{loop} the time of the k^{th} loop between two subsequent hits of $\Sigma_{z_{\text{min}}}$, conditioned to have visited A between them, and as T_{reac} the time of the reaction trajectory. If the number of loops made before visiting B is n, the time T_{AB} can be obtained as:

$$T_{AB} = \sum_{k=1}^{n} T^k_{\text{loop}} + T_{\text{reac}}. \tag{5.4}$$

At each passage over $\Sigma_{z_{\min}}$ there are two possible events: (i) first enter A, or (ii) first enter B. Using the probability p from the previous paragraph, the average number of loops before entering B is $1/p-1$. This leads us to the final equation for the expected value of T_{AB}:

$$\mathbb{E}(T_{AB}) = \left(\frac{1}{p} - 1\right)\mathbb{E}(T_{\text{loop}}) + \mathbb{E}(T_{\text{reac}}). \tag{5.5}$$

Attention !

The calculations of this section needs several hours of computer time. This is due to the difficulty to correctly sample the initial conditions, as explained below. The reader following this tutorial in a NAMD hands-on workshop is invited to skip to Section 5.2.4 and use the provided results for this section.

It has been shown that a good way to sample μ_{QSD} is to change the set of initial conditions at each run [57]. To do so, the user has to provide the value of z_{min}. A small simulation before each AMS run is performed and the first `numrep` trajectories between $\Sigma_{z_{\min}}$ and A are used as the first set of replicas. This is done just by setting the variable `amstype` to `var`. All the simulations will start from a point inside of A (files with prefix `A`).

The sampling of μ_{QSD} is not easy, and thus it is necessary to use more replicas and run more AMS simulations, compared with the simulations in Section 5.2.2), in order to get the desirable results. Go to the directory `2-time` for this part of the tutorial.

1. Copy the input file of the previous section and rename it `time.par`. A few editions are necessary.
2. Set the variable `amstype` to `var`.
3. Set the variable `numrep` to 500.
4. Set `numinst = 1000`.

First, it is necessary to provide the variable `zmin`. The choice of this parameter may be delicate. The closer $\Sigma_{z_{\min}}$ to A, the smaller the probability p to estimate. On the other hand, if $\Sigma_{z_{\min}}$ is too far from A, it will be harder to sample the loops between A and $\Sigma_{z_{\min}}$, and the underlying assumption of quasi-equilibrium before transiting to B will not be satisfied, which will imply a bias on the estimate of the transition time by formula (5.5). Moreover, the time needed in the initialization step will be larger. In this tutorial we will set `zmin = -0.6`, but we invite the reader to change this parameter and compare the results.

5. Set `zmin` to -0.6.
6. Change the variable `outdir`, otherwise the script will not run any new simulation.
7. Run the script:
   ```
   ../smart/smart_parallel.sh time.par
   ```

When using `amstype` as `var`, the script will create two more output files: `ams.T1` and `ams.T2`. To obtain the transition time the user will run the provided program `ams_time` in directory `smart`. The argument for this program is a file that contains, in this exact order: the probability and the obtained values for T_1, T_2 (whose sum is equal to $\mathbb{E}(T_{\text{loop}})$) and T_3 (equal to $\mathbb{E}(T_{\text{reac}})$). All of these values have to be provided with the confidence interval and it is possible to obtain them utilizing the executable file `media`, just like in the previous section.

8. Run this command line with files `ams.proba`, `ams.T1`, `ams.T2` and `ams.T3` (in this exact order), and redirect the output in a file named `for_time`.
   ```
   ../smart/media ams.proba >> for_time
   ```

9. Run the following command line:
   ```
   ../smart/time_ams for_time
   ```

Compare the obtained result to the reference value of: (309.5 ± 23.8) ns.

5.2.4 Calculating the flux of reactive trajectories sampled with AMS

Using a set of reaction trajectories obtained with the AMS method, each trajectory i can be associated with a vector $(\boldsymbol{\theta}^i_t)_{t\in[0,\tau^i_B]}$ with the two dihedral angles at each point. The (φ,ψ) space is split into L cells $(C_l)_{1\le l\le L}$. The flux of reactive trajectories in each cell is then defined up to a multiplicative constant by (compare with equation of Remark 1.13 in reference [15]):

$$J(C_l) = \sum_{i=1}^{n} \sum_{t=0}^{\tau^i_B - 1} \left(\frac{\boldsymbol{\theta}^i_{t+1} - \boldsymbol{\theta}^i_t}{\Delta t} \right) \mathbb{1}_{\boldsymbol{\theta}^i_t \in C_l}. \tag{5.6}$$

The parameter L should be given by the user. In this tutorial $L = 50 \times 50$.

A program that calculates the reactive trajectories flux using the expression above is provided in the `smart` directory. The user only needs to provide a file containing the list of files with the trajectories sampled by AMS. Such a file is actually given by the `smart_parallel.sh` script, and the user should find it inside the `outdir` directory under the name `paths_list`. Please note that the provided program only calculates the flux in two dimensions.

1. In the terminal, `cd` directory `3-flux`

2. Calculate the flux with the results from Section 5.2.2
   ```
   ../smart/flux ../1-point/ams/paths_list point.flux 50 20
   ```
   The last value corresponds to the `amssteptime`.

3. The same for Section 5.2.3
   ```
   ../smart/flux ../2-time/ams/paths_list time.flux 50 20
   ```

If the reader is performing only this Section of the tutorial, use the provided results located in directory `example_results`.

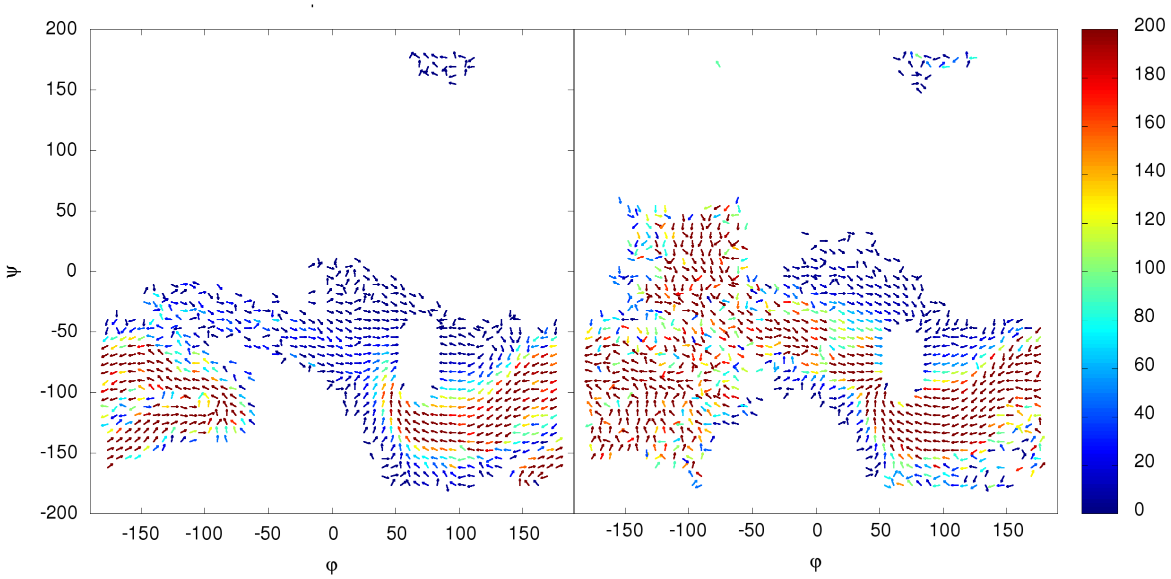


Figure 5.5 – Example of results for the flux of reactive paths.

The flux file contains five columns, that corresponds to the vector position, the vector direction (unit vector) and size. The files `point.flux` and `time.flux` can then be plotted using the program the user prefers. If the reader has access to Gnuplot program, we provided a script file, named `make_plot`, used to make Figure 5.5.

4. In side directory `3-flux`, type:
   `gnuplot make_plot`.

5. Change the variable `cutoff` and repeat the previous step until a desirable result is achieved.

The flux of reactive trajectories can give an idea of the preferable paths from A to B. They strongly depend on the initial conditions. Notice that `time.flux` was calculated using variable initial conditions created by sampling loops between A and $\Sigma_{z_{\text{min}}}$, as explained in Section 5.2.3. Thus, if corresponds to the flux of reactive trajectories at equilibrium.

Chapter 6

β-Cyclodextrin-ligand unbinding mechanism and kinetics: influence of the water model

Laura J. S. Lopes*, Jérôme Hénin•, Tony Lelièvre*

* *CERMICS, École des Ponts ParisTech, 6-8 avenue Blaise Pascal, 77455 Marne-la-Vallée, France*

• *LBT, Institut de Biologie Physico-Chimique, 13 rue Pierre et Marie Curie, 75005 Paris, France*

In this paper we analyze the mechanism and kinetics of ligand unbinding from β-cyclodextrin, with ligands 2,3-diazabicyclo[2.2.2]oct-2-ene and 1-hydroxymethyl-2,3-diazabicyclo[2.2.2]oct-2-ene. In particular, we show the influence of the water model, TIP3P and TIP4P/2005. Adaptive multilevel splitting was used to simulate the unbinding transition. Results show that the unbinding mechanism remain the same for both water models, but the time of the process is affected.

6.1 Introduction

Water models have been developed over the last four decades and remain an important field of study. However, there is not yet a classical model able to describe all the water properties with precision[58]. Nevertheless, there is a huge need for those models to describe aqueous environment, especially in the biophysics field.

Models of the TIPnP family, where n stands for the number of Coulomb or Lennard-Jones sites, are among the most used nowadays. These empirical models were fitted to reproduce some properties of the liquid at standard conditions. TIP3P, commonly used for simulations of bio-systems, was fitted to reproduce density and dimerization energy[30]. From the same family, TIP4P is a variant, where the oxygen charge is displaced to a fourth site, introduced in an attempt to better describe the second peak from the oxygen-oxygen radial distribution function. This last model has a more recent version, TIP4P/2005, were the fit was redone to better describe some thermodynamic properties of water, in both liquid and solid states[31].

Despite of being an apparently better model, TIP4P/2005 yields worse predictions for the water dielectric constant and specific heat, compared to TIP3P[59]. Also, it is more computationally expensive, as it has four sites instead of three. This can increase the computational cost by up to 33%, if we consider that the most abundant molecule present in the simulation of an aqueous environment is water. Therefore, depending on the properties one wishes to compute and the computational resources available, the best choice between these models may vary.

Despite the extensive literature in water model comparison, there is a steady interest in reproducing thermodynamic properties, and less work regarding kinetic properties. In this work, we propose a study of the influence of the water model in the unbinding process of the β-cyclodextrin with two ligands[29]. Because of the metastable property of the unbinding process, the adaptive multilevel splitting method was used to calculate the unbinding time and sample reactive trajectories.

β-cyclodextrin and ligands

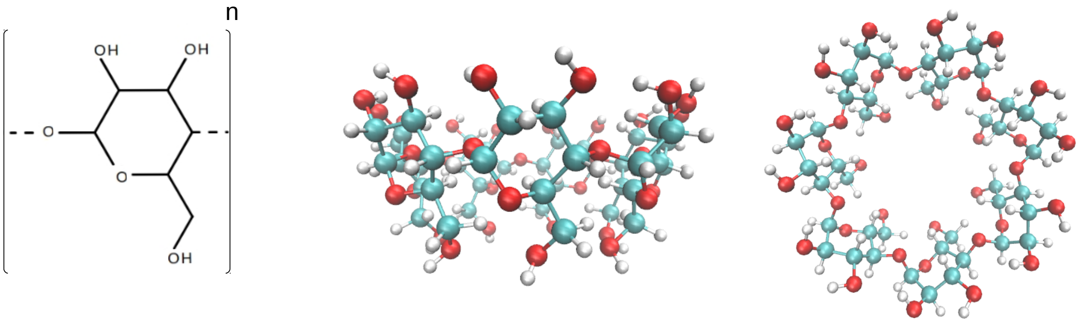


Figure 6.1 – The cyclodextrin unit and β-cyclodextrin (n=7).

Cyclodextrins are a family of molecules formed by glucopyranose units, forming a cyclic structure (see figure 6.1). Its conformation creates a hydrophobic interior environment, and a hydrophilic exterior. Consequently, cyclodextrins form inclusion complexes with several hydrophobic compounds. A variety of applications arise from this trap property, hence its use in different industries[28]. For this reason, cyclodextrins are a well-studied system, and many experimental results can be found in the present literature.

We simulated the unbinding process for the β-cyclodextrin with ligands 2,3-diazabicyclo[2.2.2]oct-2-ene and 1-hydroxymethyl-2,3-diazabicyclo[2.2.2]oct-2-ene[29], in water. Both ligands have a bicyclic structure, and only differ by the presence of a hydroxymethyl group on the second (see figure 6.2). Their common hydrophobic structure keeps them trapped in β-cyclodextrin, and make their escape into an aqueous environment a metastable transition. To simplify the discussion, we will refer to the pure bicyclic ligand as ligand I, and the other as ligand II.

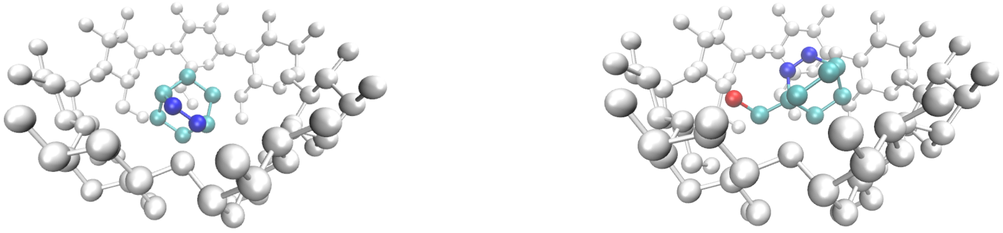


Figure 6.2 – β-cyclodextrin with ligands I and II.

6.2 Methods

The four independent systems, consisting of the β-cyclodextrin with ligand, I or II, using either water model TIP3P or TIP4P/2005, were built in a periodic box of size 42 × 42 × 35 Å, with NaCl at 0.15 M. The simulations were carried out using the NAMD program[10], with the CHARMM36 force field[5]. However, the uncommon chemical form of the ligands demanded a specific force field parameterization. This was done with the help of the Force Field Toolkit (FFTK) plugin from VMD[7, 8].

The first guess for the parameters were provided by the CGenFF program (version 1.0.0, force field 3.0.1)[60]. Those parameters were then refined using *ab initio* calculations, done with Gaussian (HF and MP2 with 6-31G*). Following the FFTK protocol, geometry optimization was done first, in order to predict the equilibrium structure. Next, point charges were fitted over the result of the distance optimization between the water and the ligand, at every donor/acceptor atom. The Hessian was calculated to obtain the force constants for the bond and angle harmonic potentials. Last, relaxed energy surfaces of all dihedrals were computed and fitted, giving rise to the torsion parameters.

As already mentioned, the unbinding process consists in a metastable transition, and thus a rare event. Hence, the use of a naive Monte Carlo approach is not possible, and a specific method is needed. The adaptive multilevel splitting (AMS) is a rare event method that rests on the splitting of the event process as a strategy to reduce computational time. This split is made on the fly, creating a succession of more probable events that are easier to simulate. For the systems with TIP3P water model, we also decided to compute a brute force estimation for the unbinding time, in order to evaluate the time gain when using AMS.

To use the AMS method, one first needs to give a precise definition for the bound and unbound states. In addition, a reaction coordinate has to be provided, in order to follow the progress towards the unbound state. Taking advantage of the shape of β-cyclodextrin, the position of the ligand was described using cylindrical coordinates, represented in figure 6.3, obtained with the Colvars plugin for NAMD/VMD[12]. To reduce computational cost, only a set of carbon atoms were used to define those coordinates, marked in black. Both the bound and unbound states, and also the reaction coordinate, were defined using the cylindrical coordinates r and z. Colvars was also used in the analysis of the reactive trajectories.

To define the bound states, a 1 ns simulation was run for each system with the ligand initially inside the cage molecule. We decided to define the states as rectangles in the (r, z) coordinates. Figure 6.4 shows the histogram for the z coordinate, that overlaps for both water models. Thus, the same definitions

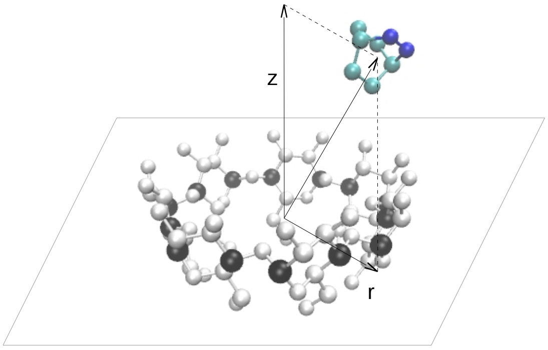


Figure 6.3 – The cylindrical coordinates used to describe the position of the ligand with respect to the β-cyclodextrin. Fitting a plane to a set of carbon atoms (in black), and considering its geometrical center the origin, vectors **r** and **z** are defined.

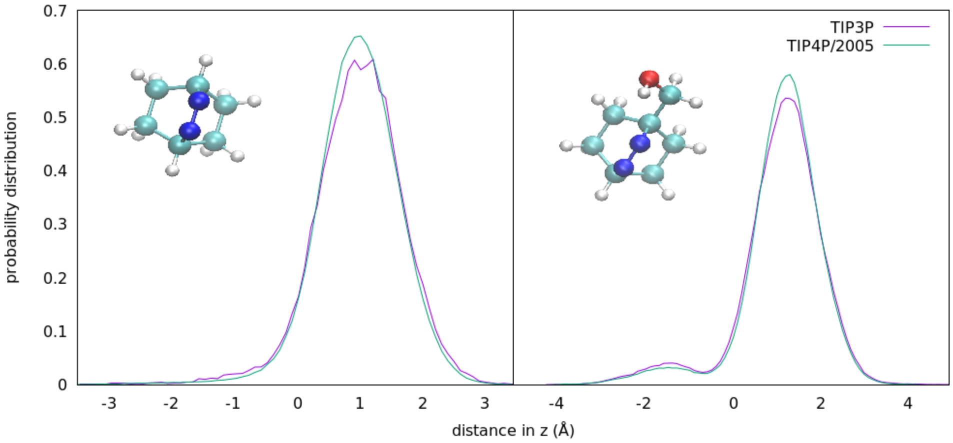


Figure 6.4 – Histogram of the *z* position for both ligands for a 1 ns trajectory. These histograms are used to define the bound state.

were used for TIP3P and TIP4P/2005. For each ligand, the mean value of *z* was considered the center of the bound state for that coordinate. The interval was set such that 80% of the total simulation time lies inside the bound state. For the coordinate *r* the goal was to radially cover all the inside of the β-cyclodextrin. This gave us the following definitions of the bound state:

$$B_I = \{r \in [0,3]\} \cap \{z \in [0.15, 1.55]\}$$
$$B_{II} = \{r \in [0,3]\} \cap \{z \in [0.45, 2.05]\}.$$

The unbound state was defined using the cutoff distance for the charge interactions, and hence is the same for both ligands:

$$U = \{r \in [19, +\infty]\} \cup \{z \in [-\infty, -16] \cup [15, +\infty]\}$$

The reaction coordinate was defined as the maximum of three affine functions, on *r* and on positive

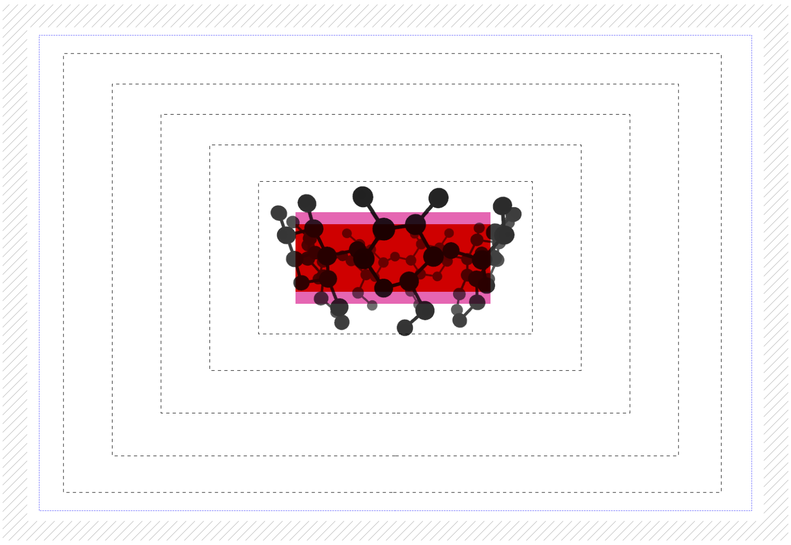


Figure 6.5 – Schematic representation of the bound state (red); the reaction coordinate level sets (dashed black lines); $\xi_{max} = 15.9$ (blue line); and the unbound state (hashed region). In pink is region Σ, used to sample the initial points for AMS.

and negative regions of z as follows:

$$\begin{aligned} \xi_I &= \max\left[r-3, \frac{16(z-1.85)}{13.15}, \frac{-16(z+0.25)}{15.75}\right] \\ \xi_{II} &= \max\left[r-3, \frac{16(z-2.45)}{12.55}, \frac{16(0.05-z)}{16.05}\right] \end{aligned} \tag{6.1}$$

Its value was set to constant over the border of the unbound states. Figure 6.5 show the reaction coordinate level sets. The bound state is represented by the red region, and the unbound by the hatched. The only condition the reaction coordinate has to satisfy to be used by AMS, is the existence of a value through which one has to pass when going from the bound to the unbound state. This last level, called ξ_{max}, equal to 15.9 for both ligands, is represented in blue in figure 6.5.

6.2.1 The Adaptive Multilevel Splitting Method for ligand unbinding from β-cyclodextrin

Starting from a set of N points (position and velocity for all atoms distributed according to some initial condition), in the neighborhood of the bound state, AMS gives an estimation for the probability to reach the unbound state. This probability will be used to calculate the transition time, discussed in the next section. All AMS simulations for our case were run using 50 initial points ($N = 50$). We will discuss below the choice of the initial condition in order to compute the equilibrium transition time.

The AMS algorithm follows the following steps (see figure 6.6):

0. **Initialization**
   At iteration $n = 0$, the first set of trajectories is generated. From each of the N initial points, a free dynamics is run until the bound or unbound state is reached. At each generated trajectory i

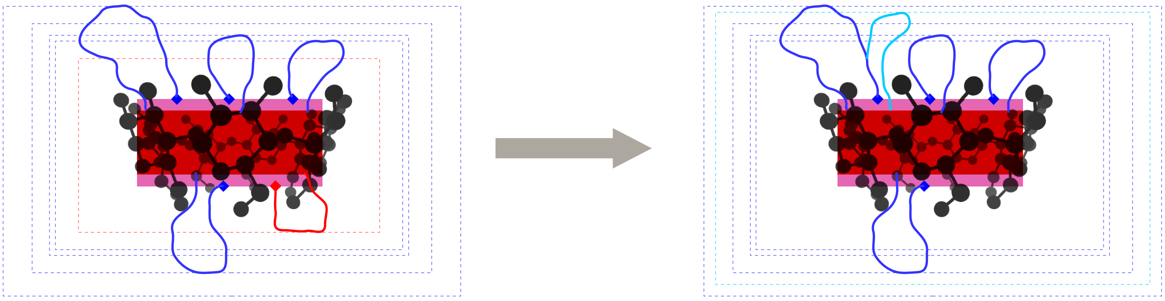


Figure 6.6 – First iteration of the AMS algorithm, with $N = 5$: the trajectory with the lower level (in red) is killed, and its level (dashed red line) is called the killing level; among the surviving trajectories (in blue), one is randomly chosen to be replicated, which means to be copied until the killing level and run independently until the bound or the unbound state is reached (see the light blue trajectory).

is associated a weight $w_{i,0}$. The initial weights $w_{i,0}$ are all equal to $1/N$. The weight represents the probability of obtaining each trajectory, and hence sums to unity at the beginning of the algorithm. The next three steps are then sequentially run until the stopping criterion is satisfied.

1. **Computation of the killing level**
   At the beginning of iteration n, the progress of each trajectory is computed as the maximum reached value of the reaction coordinate, called the trajectory level. Among the levels of all the trajectories, the k^{th} lowest is set as the killing level. The parameter k, that we call the minimum number of trajectories to kill, was here fixed to unity. Let us call k_{n+1} the number of trajectories with level lower or equal to the killing one at iteration n.

2. **Stopping criterion**
   The algorithm stops if one of the following is true:
   (I) the killing level is larger than ξ_{max}. In this case, the total number of iterations is set to n, and the estimated probability is the sum of weights of all particles that reached the unbound state:

$$p_{AMS} = \sum_{i=1}^{N} w_{i,n} \mathbb{1}_{\substack{\text{trajectory } i \\ \text{end in } U}}$$

   (II) the number of trajectories to kill is equal to the total number of trajectories ($k_{n+1} = N$), in which case no one reached the unbound state and thus the estimated probability is zero: $p_{AMS} = 0$.

3. **Replication**
   All the k_{n+1} trajectories with level lower or equal to the killing level are eliminated. Then k_{n+1} trajectories are randomly chosen among the $N - k_{n+1}$ surviving ones to be replicated. Replication consists in copying the chosen trajectory up to its first point with level larger than the killing level, and running until the bound or unbound state is reached.
   All the weights are updated by using the probability to pass this iteration's killing level. This is equal to the portion of trajectories that progressed further, which means the quantity that were

not killed. Therefore:

$$\forall i \in [1, N], w_{i,n+1} = w_{i,n} \frac{N - k_{n+1}}{N}.$$

This ends iteration n. The iteration counter is then incremented by one ($n := n + 1$), and the algorithm goes back to step 1.

Previous work on AMS showed that the expected value of the estimated probability $\mathbb{E}\left(p_{AMS}\right)$ is equal to the actual probability to reach the unbound state before going back to the bound state, starting from the chosen initial condition, and that this holds whatever the choice of the algorithm parameters[22]. Hence, in practice, the final results are mean values of estimated probabilities from independent AMS runs. This enables us to also provide statistical error bounds on the results, by using empirical variances.

The probability estimated by AMS can be used to calculate the unbinding time, see equation (6.2) below. The idea behind it is that, because unbinding is a metastable transition, in a free dynamics the ligand stays a long time doing loop movements inside the β-cyclodextrin. One can then calculate the transition time as a sum of the time spent doing such loops, computed as their duration times their number, and the reactive trajectory duration. This is explained in the following section.

6.2.2 The Transition Time Equation

To correctly describe the loops the ligand makes when trapped in the bound state, we will make use of an intermediate region, called Σ (colored in pink in figure 6.5), that contains the bound state. Let us define as a loop a segment of trajectory between two consecutive passages over the border of Σ, if that segment visits the bound state in between. Notice that, every time the ligand crosses the border of Σ there are two possibilities: return to the bound state or escape and reach the unbound state. This is described by a Bernoulli law. Calling p the probability over $\partial\Sigma$ to reach the unbound state, the mean number of loops the ligand makes before the escape occurs is $(1 - p)/p$.

The probability p can be obtained with AMS, if the initial points are sampled according to the canonical measure conditioned by $\partial\Sigma$. With a short free dynamics it is possible to obtain the mean loop time, denoted here by $\mathbb{E}(T_{\text{loop}})$. Notice that, because AMS samples reactive trajectories, it also gives their mean time, that we will call $\mathbb{E}(T_{\text{reac}})$. The transition time equation then reads:

$$\mathbb{E}\left(T_{\text{trans}}\right) = \left(\frac{1}{p} - 1\right) \mathbb{E}\left(T_{\text{loop}}\right) + \mathbb{E}\left(T_{\text{reac}}\right). \tag{6.2}$$

The estimate of p is only accurate if the initial points for AMS follow an equilibrium distribution over $\partial\Sigma$, hard to sample for a rare event. But, since the probability p is small, the number of loops is large. Therefore, one may assume that, before the transition occurs, the system reaches a quasi-stationary distribution over $\partial\Sigma$. Notice that this distribution is easier to sample, as no transition to the unbound state needs to be observed. We refer to [17] for a proper mathematical justification of equation (6.2), see also Chapter 3.

The way to correctly sample this distribution is to sample N new points before each AMS run[23]. For the β-cyclodextrin–ligand unbinding, the previously mentioned 1 ns dynamics were used. All the crossing points over $\partial\Sigma$ were kept, and before each AMS, 50 of them were randomly chosen for the initial set.

6.3 Results

		TIP3P		TIP4P/2005
		brute force	AMS	AMS
ligand I	total time (µs)	11.7	4.2 (123 runs)	15.8 (192 runs)
	time/traj. (ns)	244	0.68	1.65
	trajectories	48	6135	9577
ligand II	total time (µs)	8.8	3.2 (57 runs)	22.3 (180 runs)
	time/traj. (ns)	518	1.12	2.48
	trajectories	17	2849	8992

Table 6.1 – Total simulation time and number of generated unbinding trajectories: with AMS for the four systems; and with brute force for the systems with TIP3P.

For each system, the AMS simulations were run until the desired convergence of the probability estimator was reached. Table 6.1 shows the total number of AMS needed for each system, and the total simulation time required to run them. An attempt to obtain brute force results was made for the systems with TIP3P, but the convergence of the estimator was not reached. However, those simulations generated a few unbinding trajectories, and the time required to obtain each one of them can be used to make a rough estimate of the AMS computational gain compared with the brute force approach. Using this result, the computational cost is divided by 356 for ligand I, and by 462 for ligand II.

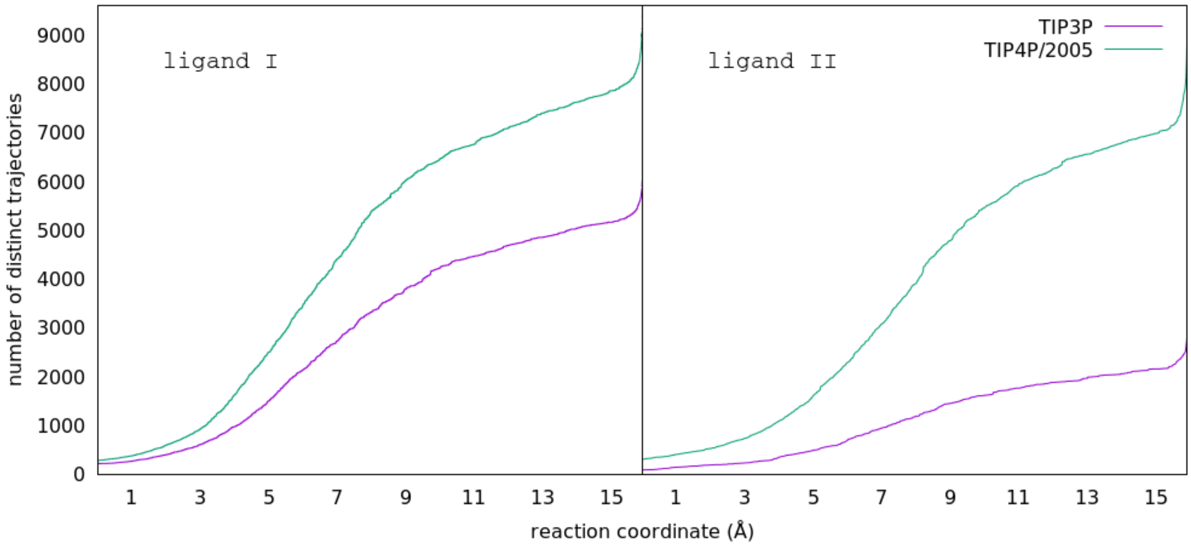


Figure 6.7 – Number of distinct trajectories during the AMS runs. For the TIP3P the brute force approach only generated 48 unbinding trajectories for ligand I and 17 for ligand II.

The unbinding trajectories generated by AMS have common segments, because, by construction, AMS gives trajectories that are branched at the killing levels. This is due to the replication step (see section 6.2.1). This correlation is lost as the trajectories progresses towards the unbound state. Figure 6.7 shows the number of distinct trajectories during the unbinding process. Hence, the further from the bound state the more diversity there is. Consequently, the analysis of the AMS generated trajectories presents less noise for larger values of the reaction coordinate. Notice that, for the unbinding problem, and considering the geometry of the β-cyclodextrin-ligand system, the reaction coordinate isolevel surfaces area increases with the level. This means, the space the ligand explores also increases with the level of the reaction coordinate, so it is larger near the unbound state than near the bound state. Therefore, it is more important to have more diversity in the unbinding trajectories as the ligand progress towards the unbound state, as seen in figure 6.7.

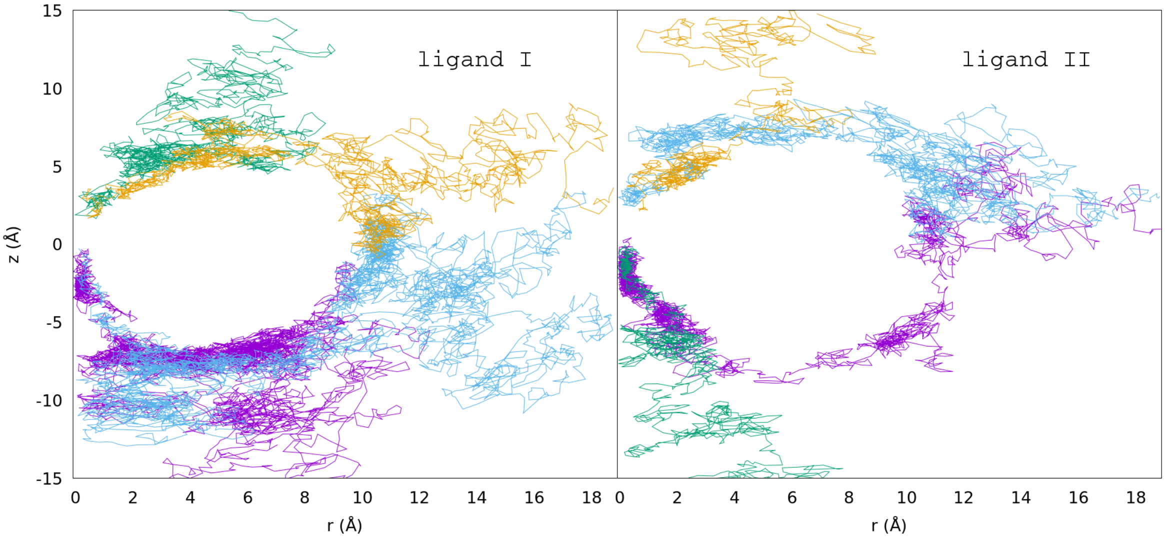


Figure 6.8 – A few example trajectories generated by AMS using the TIP3P water model.

Figure 6.8 shows a few trajectories generated by AMS with the TIP3P water model projected into the $r \times z$ plane. These trajectories shows that the used reaction coordinates does not respect the geometry of the problem. Because the quality of this function only influence the variance of the AMS estimator, one can conclude that the use of a better one would improve the computational efficiency. Notice that the area that is never visited corresponds to the β-cyclodextrin. One could take advantage of this geometry to come up with other possible reaction coordinates. For example, the distance from the β-cyclodextrin area, which can be modeled as an ellipse.

ligand	experimental[29]	TIP3P	TIP4P/2005	TIP4P/TIP3P ratio
I	2.3(5)	0.097(16)	0.95(26)	9.8
II	0.54(9)	0.22(9)	2.1(5)	9.5

Table 6.2 – Transition time in μs, obtained with AMS and equation (6.2).

Results for the unbinding time, obtained using equation (6.2), are presented on table 6.2. The first noticeable result is the difference between TIP3P and TIP4P/2005, that has almost the same ratio for both ligands: a factor of 10. The second is the failure to reproduce the experimental results. It is known that the total reproducibility of real conditions in classical molecular simulations is not guaranteed. Molecular dynamics gives insights about the behavior in an atomic scale, but with no promise for accuracy due to the approximate character of classical force fields. It is also known that the experimental measurement of kinetic quantities is hard, and hence not completely reliable.

In the search for the origin of the disparity in the kinetic result using both water models, a first set of analysis was made to elucidate the structural description of unbinding trajectories, i.e. the mechanism. These results are presented and discussed in section 6.3.1. Because no relevant difference was found, a second set of analysis was proposed, presented in section 6.3.2, comparing kinetic properties of elementary events contributing to the unbinding process.

6.3.1 Unbinding mechanism

To elucidate the mechanism, we computed the probability of contact between the ligand and the β-cyclodextrin, along the unbinding trajectories sampled with AMS. In order to explain the equation used, let us first recall that the weight given by AMS to each trajectory represents its probability of occurrence. The progress of the trajectories was measured using the center of mass distance from the ligand bicycle to the binding site, that was discretized in intervals of size 0.1 Å. The following equation gives the probability of contact between two atoms for the distance interval I_j.

$$PC_j = \frac{\sum_{i=1}^{M} w_i \sum_{t=0}^{T_i} \mathbb{1}_{d_{i,t} \in I_j} \mathbb{1}_{b_{i,t} \leq 2.5}}{\sum_{i=1}^{M} w_i \sum_{t=0}^{T_i} \mathbb{1}_{d_{i,t} \in I_j}} \tag{6.3}$$

Here w_i is the weight of trajectory i, and M is the total number of trajectories. For trajectory i at time t, $d_{i,t}$ is the center of mass distance, and $b_{i,t}$ is the bond distance between the atoms whose contact probability we aim to obtain. Notice from equation (6.3) that a contact is considered to exist when the bond length is inferior to 2.5 Å.

The interactions considered were between the nitrogen, from the ligand, and the hydroxyl groups of the β-cyclodextrin. For ligand II, the hydrogen bonds with its hydroxyl group were also considered. The distance between those groups were computed for each step of every unbinding trajectory.

Let us highlight the existence of two escape exits from the binding site, caused by its ring structure. In order to analyze separately the interaction with the hydroxyl groups from each exit, those were separated in two groups: top and bottom. Figure 6.9 show this separation with a color code. For each group, only the minimum interaction distance at each step was kept, which was then used as $b_{i,t}$ at equation (6.3). Results are presented in figure 6.10.

The probability of contact does not show an important influence of the water model (see figure 6.10). The main difference is for ligand II hydroxyl group at low distances, that presents a higher probability of contact in TIP4P/2005. Besides, results from both water models suggest that these calculated contacts have a significant role in the unbinding mechanism, since they have high probability of occurrence through a large distance range.

For the contact between the ligand nitrogen and the top β-cyclodextrin groups, both ligands exhibit

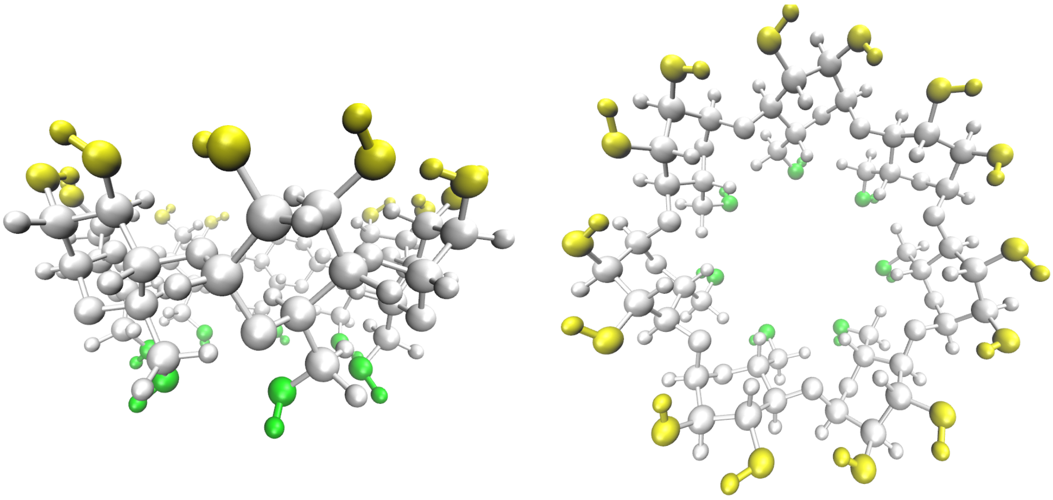


Figure 6.9 – "Top" (yellow) and "bottom" (green) hydroxyl groups of β-cyclodextrin.

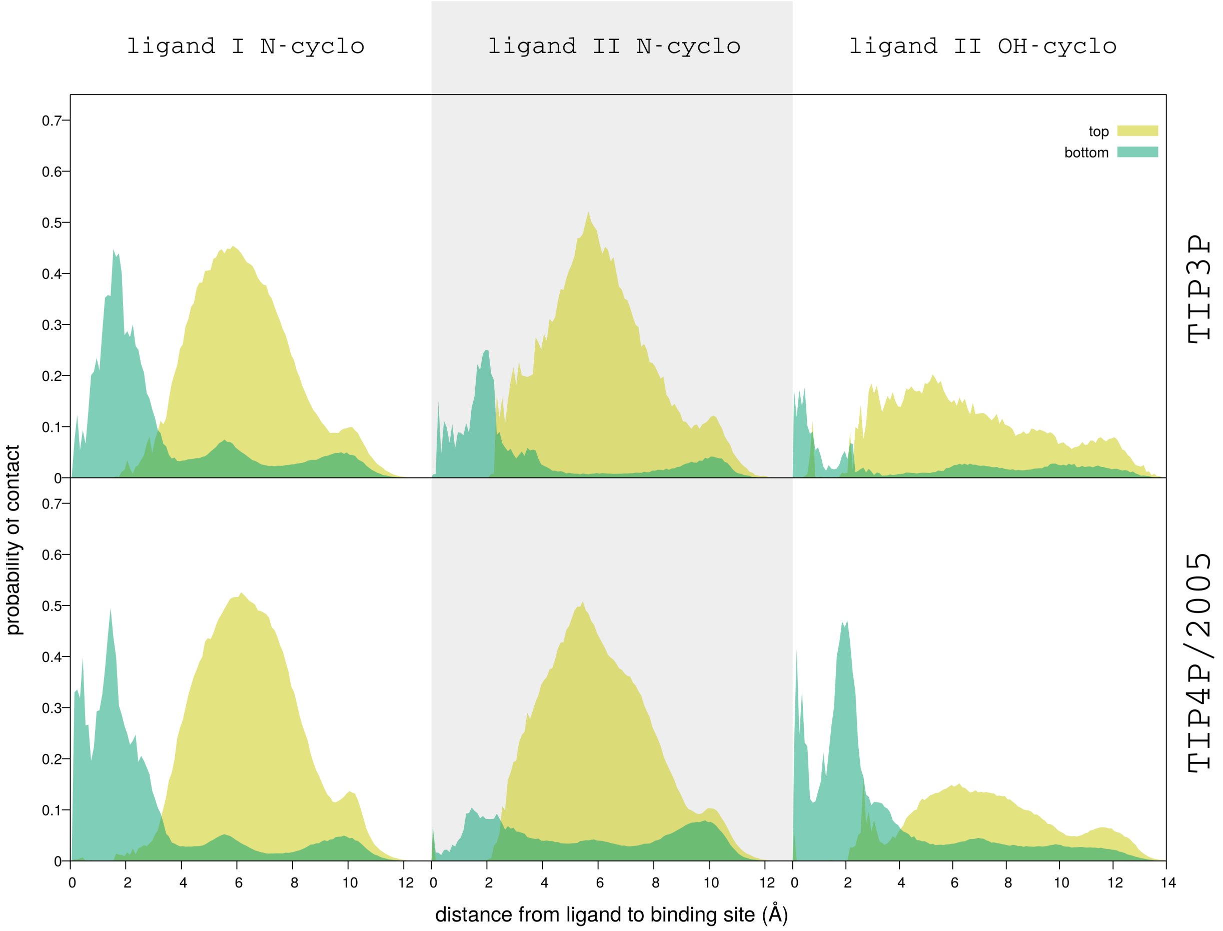


Figure 6.10 – Probability of contact between the ligand and hydroxyl groups top (yellow) and bottom (green) (see figure 6.9) of the β-cyclodextrin with both water models.

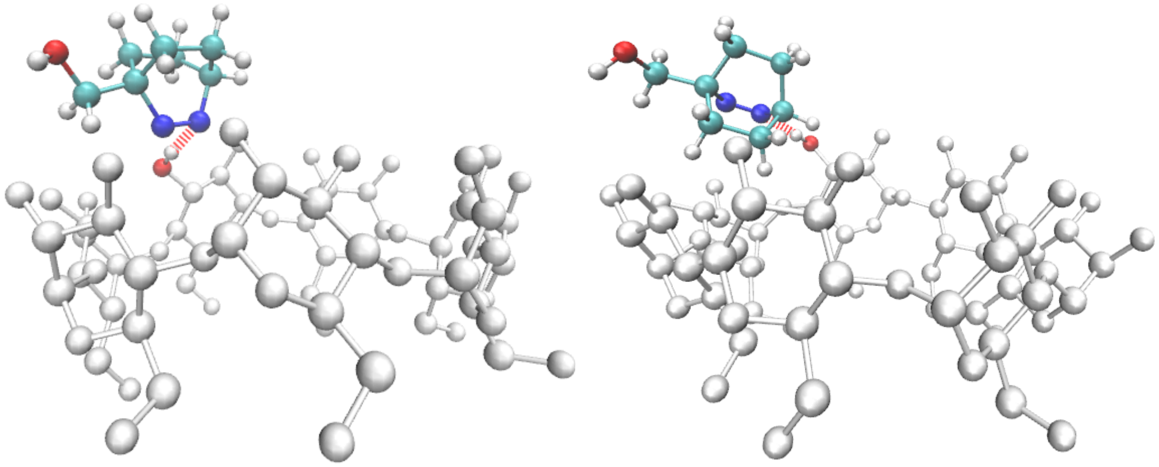


Figure 6.11 – Contact with the nitrogen and the top hydroxyl groups, around a distance of 6 Å, that participate in a pivot exit mechanism.

a peak around 6 Å. This distance is equivalent to the ligand positioned at the upper edge of the β-cyclodextrin, certainly the most favorable contact position (see figure 6.11). But, it is important to mention that the high width of this peak is originated by a pivot exit mechanism, where the ligand rotate around the top border.

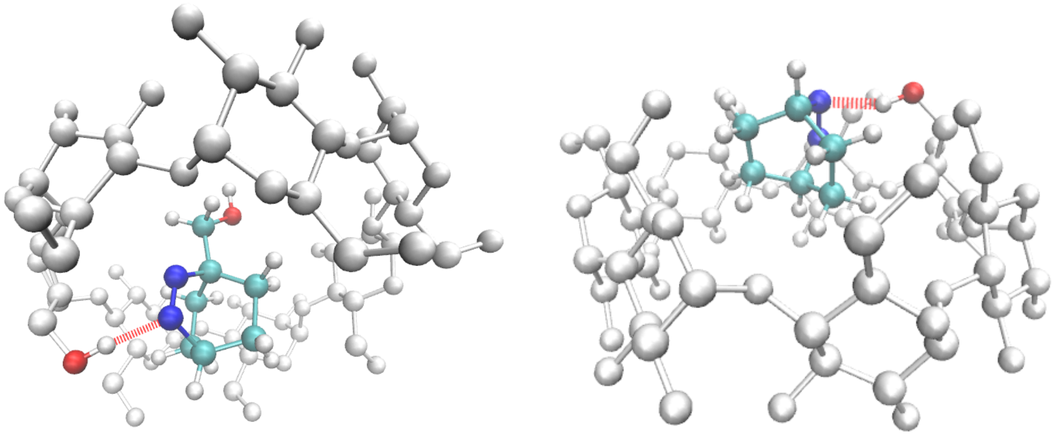


Figure 6.12 – Contact with the nitrogen from both ligands and the bottom hydroxyl groups at low distance.

The lower probability of contact with the bottom groups is caused by the lower number of trajectories where the ligand exits the site via the bottom path. Also, because these hydroxyl groups are connected to a more flexible chain, they can make contact at a lower distance than the top groups (see figure 6.12). This flexibility is also responsible for the wider distribution over the distances.

6.3.2 Understanding the difference in kinetics between the water models

Because the mechanism analysis did not reveal any important qualitative difference in the behavior of the molecules depending on the water model, we compared dynamical quantities. The first one is the duration of the contacts for which the probability was calculated in the previous section. The second is the diffusion coefficient of the ligands in pure water.

For the transition to occur, the ligand has to break all the hydrogen bonds with the β-cyclodextrin. This lead us to investigate the life time of the contacts between both molecules. This quantity is defined as the mean time of all the intervals for which the analyzed bond length is larger than a cutoff. The average is calculated using all the reactive trajectories generated by AMS. Let us denote by $T_{\text{life}}(c)$ the contact life time at a certain cutoff distance c, defined by:

$$T_{\text{life}}(c) = \frac{\sum_{i=1}^{M} w_i \sum_{j=1}^{m_i(c)} \Delta t_j^i(c)}{\sum_{i=1}^{M} w_i m_i(c)}. \tag{6.4}$$

Here M is again the total number of AMS trajectories. The trajectory of index i has weight w_i, and a set of $m_i(c)$ intervals for which the bond length is larger than c.

Figure 6.13 show the results obtained using equation (6.4). The duration of contacts is slightly larger for the TIP4P/2005, but this effect is small. Thus, although this difference goes in the same direction as the obtained transition times, it is not sufficient to explain the kinetics difference in the unbinding process.

Another important part of the transition was then taken into account: the travel between β-cyclodextrin and the unbound state. This is done mostly by passing through a region with only water. The speed of this process is measured by the diffusion coefficient of the ligands in the liquid. To obtain this quantity a set of simulations of each ligand in a periodic box of water, using both models, were performed. The total simulation times vary from 6 to 11 ns. Table 6.3 show the diffusion coefficients, obtained through linear fit of the mean square distance from 20 to 100 ps. This time period is sufficient to see the distance the ligands have to go through in order to reach the unbound state. The coefficients show a difference of order around two between the two models, which is almost the same observed for the self-diffusion coefficient. Although this difference is again in the same direction as the obtained transition times, this alone does not explain the unbinding kinetic difference, and we can then expect that the explanation lies in a combination of the both explored phenomenons. Also, for the ligand to break bonds with β-cyclodextrin and reach a purely aqueous environment, the water solvation layers of both the ligand and the binding site have to be reorganized. The time for this to occur is maybe not

	TIP3P	TIP4P/2005	ratio
ligand I	10.678(4)	5.861(4)	1.82
ligand II	8.906(3)	4.143(3)	2.15
self[59]	5.19	2.08	2.49

Table 6.3 – Diffusion coefficients, in 10^{-5} cm^2/s, calculated for the two ligands in both water models.

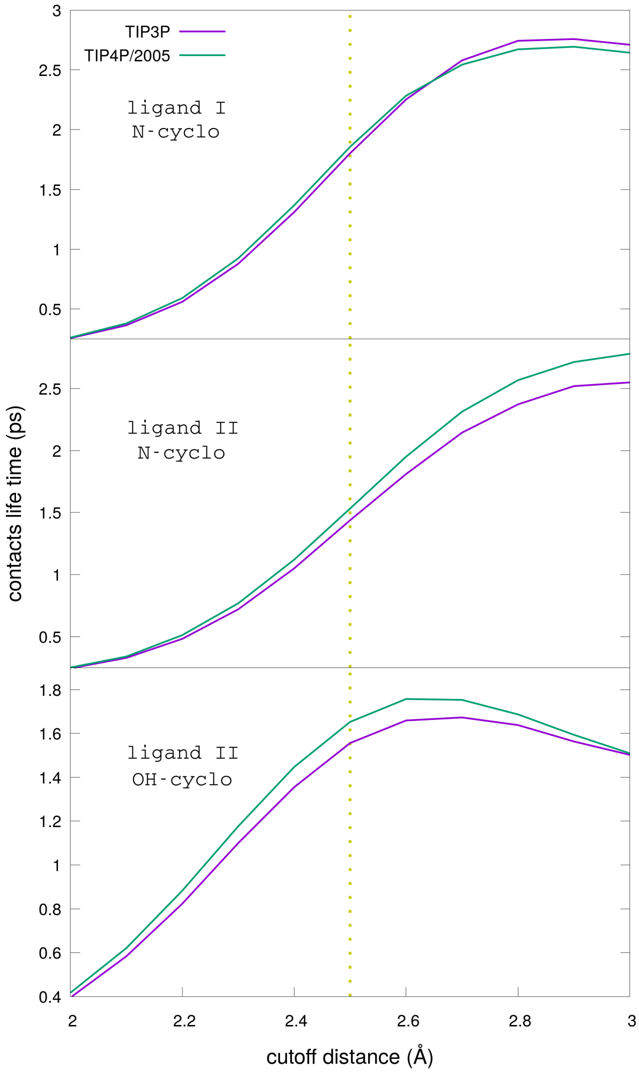


Figure 6.13 – Hydrogen bond life times for ligands I and II. The dashed line represents the cutoff used to calculate the probability of occurrence for the same bonds.

the same for both ligands.

6.4 Conclusion and Perspectives

The analysis of the reactive trajectories shows equivalent exit mechanisms for both water models. This is an indication that there is no disparity in the qualitative behavior predicted with TIP3P and TIP4P/2005. However, the estimated unbinding times obtained with TIP4P/2005 are ten times the ones obtained with TIP3P.

Results for hydrogen bond life time and diffusion coefficient are consistent with the AMS results. The difference in the unbinding time is certainly caused by a combination of different factors. The first is the resistance from the ligands movement inside the liquid, that is measured by the diffusion coefficient. The second is the strength of the hydrogen bonds, indirectly measured by the bond lifetimes, that however showed a low relevance contribution.

Yet, other phenomenons can contribute and new analysis can be made to exhibit them. For example, for the ligand to exit the binding site, the water molecules have to rearrange both to solvate the ligand and replace it inside β-cyclodextrin. Transition state theory analysis would calculate the solvation free energy of both transition states to find if they are favored by one of the water models. Another phenomenon is the shielding caused by the water molecules, that would certainly affect the interaction between the ligand and the β-cyclodextrin. This is measured by the dielectric constant, that is higher for the TIP3P model (82 for TIP3P against 60 for TIP4P/2005[59]). Hence, TIP3P better shields the contacts between both molecules, weakening them. This may explain a quicker escape with TIP3P. This will be the subject of future investigations.

Chapter 7

Ligand unbinding from Heat Shock Protein 90

7.1 Introduction

Heat shock proteins acts as chaperones that preserve cell functions in response to a sudden increase in temperature[61]. Heat Shock Protein 90 (Hsp90), present in humans, participates in a number of processes. Although not all mechanisms are yet elucidated, it is known that Hsp90 acts in the development of some types of cancer. Its overexpression in cancer cells makes of cancerous cells more sensitive to chemical inhibitors of Hsp90. Those act mostly by blocking its N-terminal part, that binds ATP to power the protein's functional cycle.

A common step when developing a new drug is to estimate the affinity between the ligand and the binding site. However, it is known that the drug's efficiency depends on its residence time at the target[32]. Hence, the calculation of the drug-target unbinding time is an important step in drug design.

The unbinding process consists in a escape from a metastable state, and thus a rare event. Some drug candidates have a residence time of a few hours in the N-terminal cavity of Hsp90[32]. It is then necessary to make use of a rare event method to simulate these transitions, as the naive Monte Carlo approach is computationally too expensive. A recent publication obtained a residence time near 39 s for the ligand considered in this chapter, using the biased targeted molecular dynamics (TMD) method[62]. However, such techniques introduce errors compared to what would have been obtained on the original model, which cannot be quantified.

In this chapter we present a project done in collaboration with pharmaceutical researchers at Sanofi, where we used the adaptive multilevel splitting (AMS) algorithm to obtain the unbinding time between a drug candidate the N-terminal domain of Hsp90. As already explained in Chapter 2, this algorithm yields very accurate estimates of unbinding times, and the objective of the work presented in this chapter is to explore the difficulties associated to its use on a large and complicated test case. This work is still in progress, and we present here the results obtained until now. Section 7.2 presents the system set up and the AMS method for this problem. In section 7.3 we present the obtained results as well as the perspectives for future works.

7.2 Set up of the system and numerical method

The crystallographic structure, first provided by Sanofi, then published as Protein Data Bank id 5LR1, is showed in figure 7.1. The ligand (A003498614A) is inside the nucleotide-binding cavity, which is a metastable state.

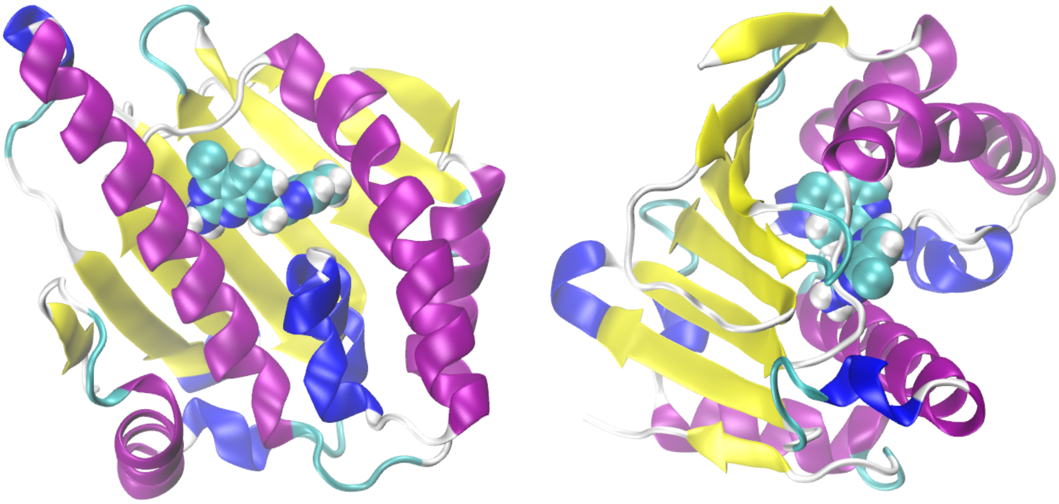


Figure 7.1 – N-terminal part of Hsp90, with ligand inside its cavity (structure PDB 5LR1).

The simulations where carried out using the NAMD program[10], under NPT conditions, at 300 K and 1 atm, Langevin thermostat and barostat settings, and a time step of 2 fs. In order to decrease the computational cost, and also taking advantage of the protein's geometrical form, a truncated octahedron periodic box was used. With the addition of 0.15 M of NaCl, the entire system has 18,732 atoms. The water model was TIP3P and the force field was CHARMM36[5]. A specific force field was parameterized for the ligand, as detailed below.

7.2.1 Custom force field for the ligand

The uncommon structure of the ligand (see figure 7.2) demanded a specific force field parameterization. This was done with the help of the Force Field Toolkit (FFTK) plugin from VMD[7, 8]. The attribution of atom types and first guess for the parameters was made using the CGennFF online platform (version 1.0.0, force field 3.0.1)[60]. The parameters were then refined by a fit to *ab initio* data, obtained with Gaussian (MP2/6-31g(d))[63]. The first quantum data was the optimized geometry.

Two auxiliary molecules where used to fit the charges and the force constants for the bond and angle parameters (see figure 7.3). This was necessary to prevent the appearance of a spurious dipole in the ligand, and to enable the calculation of the *ab initio* Hessian matrix, as the ligand size did not permit this computation with our available memory. Those molecules corresponds to the ligand separated

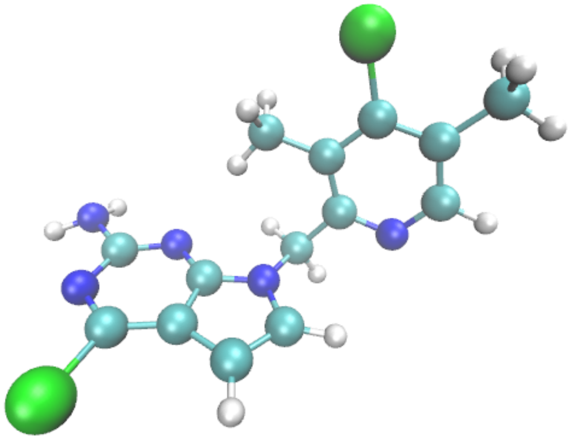


Figure 7.2 – The ligand from structure PDB 5LR1.

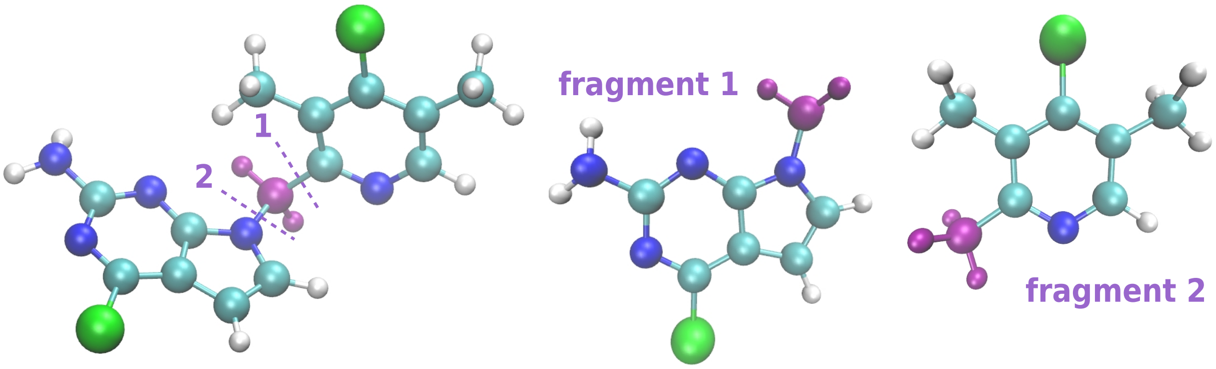


Figure 7.3 – The ligand, and the auxiliary molecules used to fit the charges and the force constants for the bond and angle harmonic potentials.

at the carbon that links the aromatic and bicyclic structures (in purple). Hence, the atoms of those two molecules have the same type as those of the ligand, with the exception of the ones marked in purple or black. Respecting the atom types, the charges and the parameters for bond and angle terms were parameterized for the auxiliary molecules and used for the ligand. To fit the charges, at every donor and acceptor atom the distance from a water molecule was optimized. The charge of the joining methylene carbon was assigned as the average charge between the corresponding methyl carbons in the fragments. The same was done for the charges of the child hydrogens, that were equally divided between the purple ones. The force constants for the bond and angle parameters were obtained via the Hessian matrix.

To generate the torsion parameters, for each dihedral angle a relaxed energy scan was made, where the geometry was optimized for a range of dihedral values around the equilibrium one. The fit of this data revealed that the potential of two dihedrals could not be described as a sum of terms, which was not surprising as they have three common atoms (see figure 7.4), making them highly correlated. A CMAP correction was used for these dihedrals, which is a grid based energy map[9], initially designed to better describe protein backbones. For this cross term, an *ab initio* scan in two dimensions was performed over the $\phi \times \psi$ plane, that was discretized in cells of size 15×15 degrees. The final correction

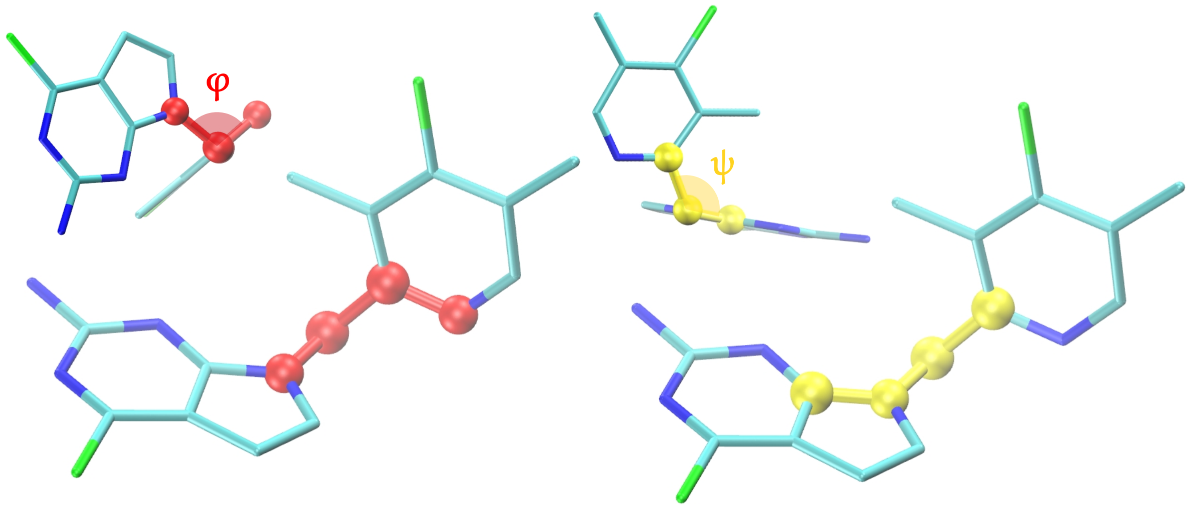


Figure 7.4 – The atoms from dihedrals ϕ (in red) and ψ (in yellow), of the CMAP correction. The molecules on the top represents the view along the central bond for each dihedral.

was obtained through an iterative technique, where the goal is to reproduce the quantum energy surface using the classical force field. Let us call $\left(E^{\mathrm{Q}}_{i,j}\right)_{i\in[1,24],j\in[1,24]}$ the matrix of the quantum scan results.

In this procedure a relaxed classical scan, done over the same values of ϕ and ψ, is obtained. The CMAP correction is the difference between the quantum and the classical scan. However, to ensure the correct relaxation of the other degrees of freedom for every point in the classical scan, this is done iteratively. The first classical scan is done using the CMAP equal to zero for every value of the dihedrals. The first CMAP matrix is then the difference between the quantum and this first classical scan. The scan is repeated and the CMAP recalculated sequentially until convergence is reached. Let us call $\left(E^{\mathrm{CMAP},q}_{i,j}\right)_{i\in[1,24],j\in[1,24]}$ the CMAP correction at iteration q, and $\left(E^{\mathrm{C},q}_{i,j}\right)_{i\in[0,24],j\in[1,24]}$ the results from the classical scan, done with the CMAP correction of iteration q. The CMAP correction is updated as follows:

$$\forall i \in [1,24], \forall j \in [1,24], E^{\mathrm{CMAP},q+1}_{i,j} = E^{\mathrm{CMAP},q}_{i,j} + E^{\mathrm{Q}}_{i,j} - E^{\mathrm{C},q}_{i,j}.$$

The map was considered converged when the maximum absolute value of the correction update was lower than 0.5 kcal/mol. Figure 7.5 show the result of the *ab initio* scan and the final CMAP correction. The negative values near $\phi = -100, 140$ are necessary to describe the stable conformations of the ligand, which includes the equilibrium one.

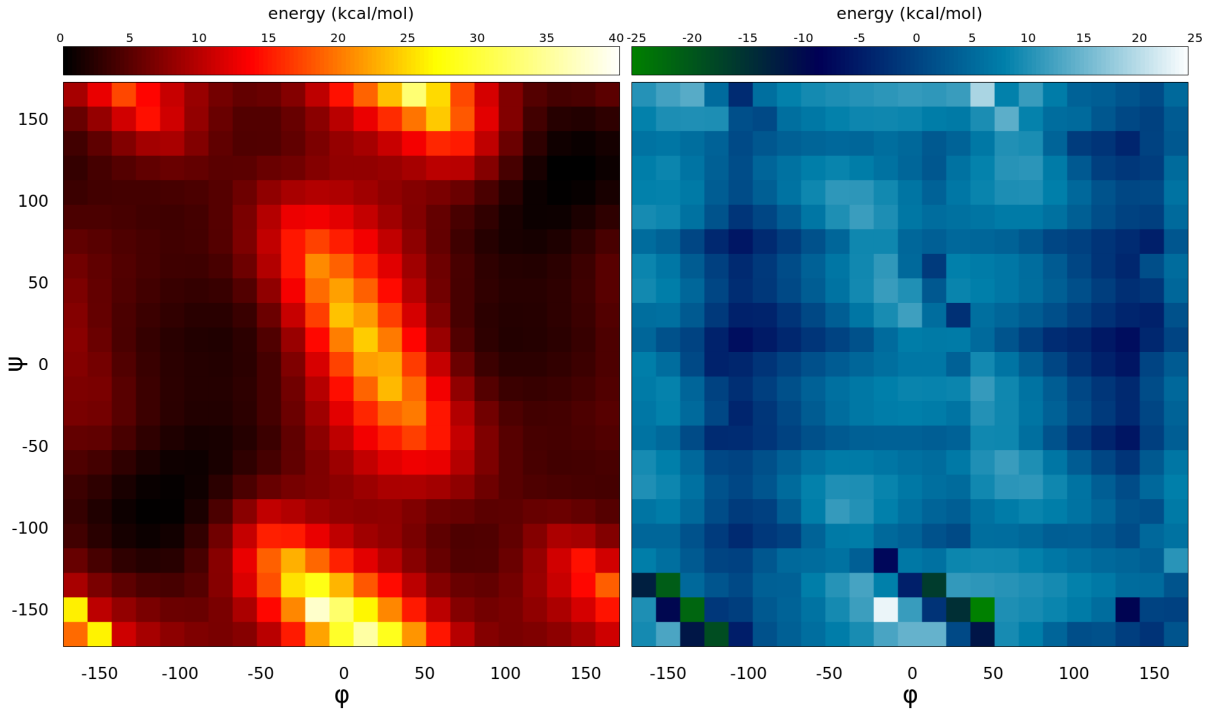


Figure 7.5 – The *ab initio* energy surface for dihedrals ϕ and ψ and the CMAP necessary to obtain the same result with the classical force field.

7.2.2 Calculating the unbinding time with AMS

Our goal is to calculate the unbinding time, or the transition time between the bound and the unbound states, defined using a set of collective variables. The unbinding time is defined as the average duration of trajectories between visits from the bound to the unbound state. More precisely, it is the expected duration of a trajectory from its first entrance into the bound state until its next entrance into the unbound state. Because of the metastable character of the unbinding process, the unbinding trajectory consists essentially in loops between the bound state and its neighborhood. Then, the residence time can be computed as the average loop duration times the number of loops. We refer to Chapter 3 and [17] for a mathematical formalization of this idea.

To correctly describe those loops we will make use of an auxiliary region, which we will call Σ. This region contains the bound state. Let us define as loop a segment of trajectory between two exits from Σ, if that segment visits the bound state. Notice that, every time the ligand crosses the border of Σ there are two possibilities: return to the bound state or escape and reach the unbound state. This is described by a Bernoulli law. Calling p the probability to reach the unbound state, the mean number of loops the ligand makes before the escape occurs is $(1-p)/p$. Let us call $\mathbb{E}(T_{\text{loop}})$ the time of a loop, and $\mathbb{E}(T_{\text{reac}})$ the duration of the reactive trajectory, i.e. between $\partial\Sigma$ and the unbound state. The following equation gives then the unbinding time:

$$\mathbb{E}\left(T_{\text{unb}}\right) = \left(\frac{1}{p}-1\right)\mathbb{E}\left(T_{\text{loop}}\right) + \mathbb{E}\left(T_{\text{reac}}\right). \tag{7.1}$$

The quantities in the right-hand side are computed starting from an initial condition obtained as a quasi stationary distribution of the loop process, as explained below.

Notice that one can obtain $\mathbb{E}(T_{\text{loop}})$ with a relatively short simulation. The probability p and the reactive trajectory time $\mathbb{E}\left(T_{\text{reac}}\right)$ will be obtained by AMS. AMS estimates the probability to reach the unbound state when starting from a fixed set of N points. Hence, to obtain p one has to start from a set of points sampled according to the equilibrium canonical measure conditioned by $\partial\Sigma$.

To obtain the equilibrium distribution over the border of Σ one has to simulate the system for long enough to see the ligand get out from the cavity and come back several times. This is impossible due to the time scales of the process and the limited time scale of atomistic molecular dynamics simulations. But, since the probability p is low, the number of loops is high. Therefore, one may assume that, before the transition occurs, the system reaches a quasi-stationary distribution over $\partial\Sigma$. Notice that this distribution is easier to sample, as no transition to the unbound state needs to be observed.

In order to correctly sample this distribution, new points are drawn before each AMS runs[23]. For Hsp90, once the bound state and the region Σ were defined, an initial simulation was used to collect a set of points over $\partial\Sigma$. Before each AMS, a set of N points was randomly sampled among the total ensemble of possible points.

To run AMS one has to give a reaction coordinate to compute the progress of the trajectories towards the unbound state. This need to be a real-valued function, which we will call ξ, and the only condition over it is the existence of a value of this function that is surpassed when entering the unbound state. Let us call B the bound state and U the unbound state. The condition then reads:

$$\exists \quad \xi_{max} \in \mathbb{R} \text{ such that } \xi(X) > \xi_{max}, \forall X \in U.$$

The AMS algorithm follows the following steps:

0. **Initialization**
   At iteration $n = 0$, the first set of trajectories is generated. From each of the N initial points, a free dynamics is run until the bound or unbound state is reached. At each generated trajectory i is associated a weight $w_{i,0}$. The initial weights $w_{i,0}$ are all equal to $1/N$. The weight represents the probability of obtaining each trajectory, and hence sums to unity at the beginning of the algorithm. The next three steps are then sequentially run until the algorithm reaches the stopping criterion.

1. **Computation of the killing level**
   At the beginning of iteration n, the progress of each trajectory is computed as the maximum reached value of the reaction coordinate, called the trajectory level. Among the levels of all the trajectories, the k^{th} lowest is set as the killing level, where k, the minimum number of killed trajectories, is a tunable parameter of AMS. Let us call $k_{n+1} \geq k$ the number of trajectories with level lower or equal to the killing one at iteration n.

2. **Stopping criterion**
   The algorithm stops if one of the following is true:
   (I) the killing level is larger than ξ_{max}. In this case, the total number of iterations is set to n, and

the estimated probability is the sum of weights of all the particles that reached the unbound state:

$$p_{AMS} = \sum_{i=1}^{N} w_{i,n} \mathbb{1}_{\substack{\text{trajectory } i \\ \text{end in } U}}$$

(II) the number of trajectories to kill is equal to the total number of trajectories ($k_{n+1} = N$), in which case no one reached the unbound state and thus the probability is zero: $p_{AMS} = 0$.

3. **Replication**
   All the k_{n+1} trajectories with level lower or equal to the killing level are eliminated. Then, k_{n+1} trajectories, are randomly chosen among the $N - k_{n+1}$ surviving ones to be replicated. Replication consists in copying the chosen trajectory up to its first point with level larger than the killing level, and running until the bound or unbound state is reached.
   All the weights are updated by using the probability to pass this iteration's killing level. This is equal to the portion of trajectories that progressed further, which means the quantity that were not killed. Therefore:
   $$\forall i \in [1, N], w_{i,n+1} = w_{i,n} \frac{N - k_{n+1}}{N}.$$
   This ends iteration n. The iteration counter is then incremented by one ($n := n + 1$), and the algorithm goes back to step 1.

Previous work on AMS showed that the expected value of the estimated probability $\mathbb{E}\left(p_{AMS}\right)$ is equal to the actual probability to reach the unbound state before going back to the bound state, starting from the chosen initial condition, and that this holds whatever the choice of the algorithm parameters[22]. Hence, in practice, the final results are mean values of estimated probabilities from independent AMS runs. This enables us to also provide statistical error bounds on the results, by using empirical variances.

7.3 Details on the numerical procedures and results

The center of mass distance between the ligand and the residues from the protein cavity (see figure 7.6) was used both as a reaction coordinate and to define the bound and unbound states. To reduce computational cost, this was calculated using only the α-carbons. Figure 7.6 shows the histogram of this distance for a set of five simulations, totaling 174.8 ns. The bound state was defined as the interval $[0, 3]$, and region Σ as $[0, 3.1]$, hence the AMS initial points where at a distance of 0.1Å from the bound state.

To obtain qualitative insight into the unbinding process and pathways, we used a non-equilibrium simulation method likely to provide rapid, yet biased results. This method is the adiabatic bias molecular dynamics (ABMD)[13], in which a wall potential, in the form of a half harmonic function, is added over the reaction coordinate. This potential moves through the simulation, and is located at the maximum value of ξ reached until that moment, preventing its decrease through the simulation and thus generating biased unbinding pathways. Aside, it is useful to probe the presence of additional metastable states along unbinding pathways. If there is no metastable state, the distance between

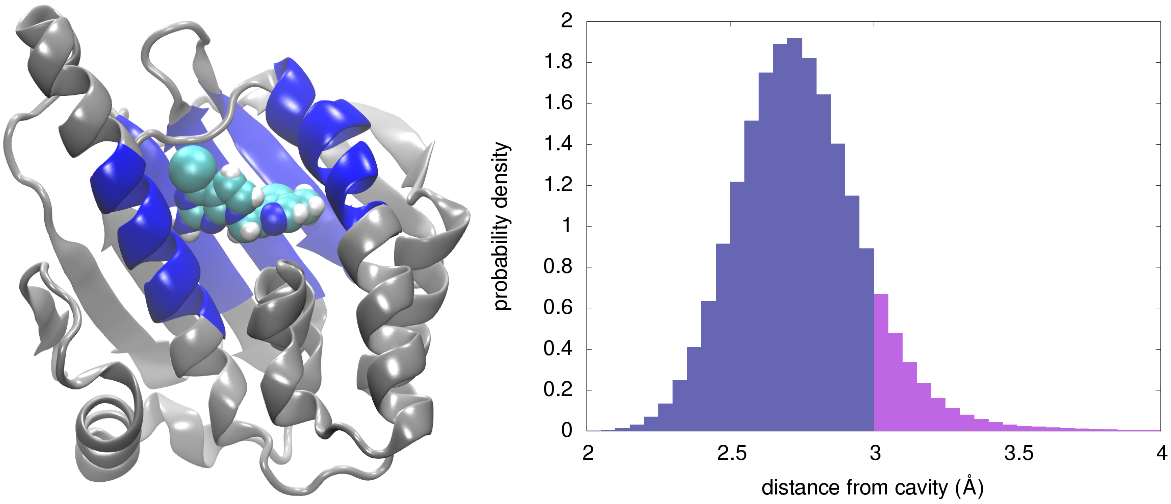


Figure 7.6 – Residues from the Hsp90 cavity (in blue), used to define the bound and unbound state, as well as the reaction coordinate; histogram of the distance between the ligand and the cavity. The bound state is represented in dark purple.

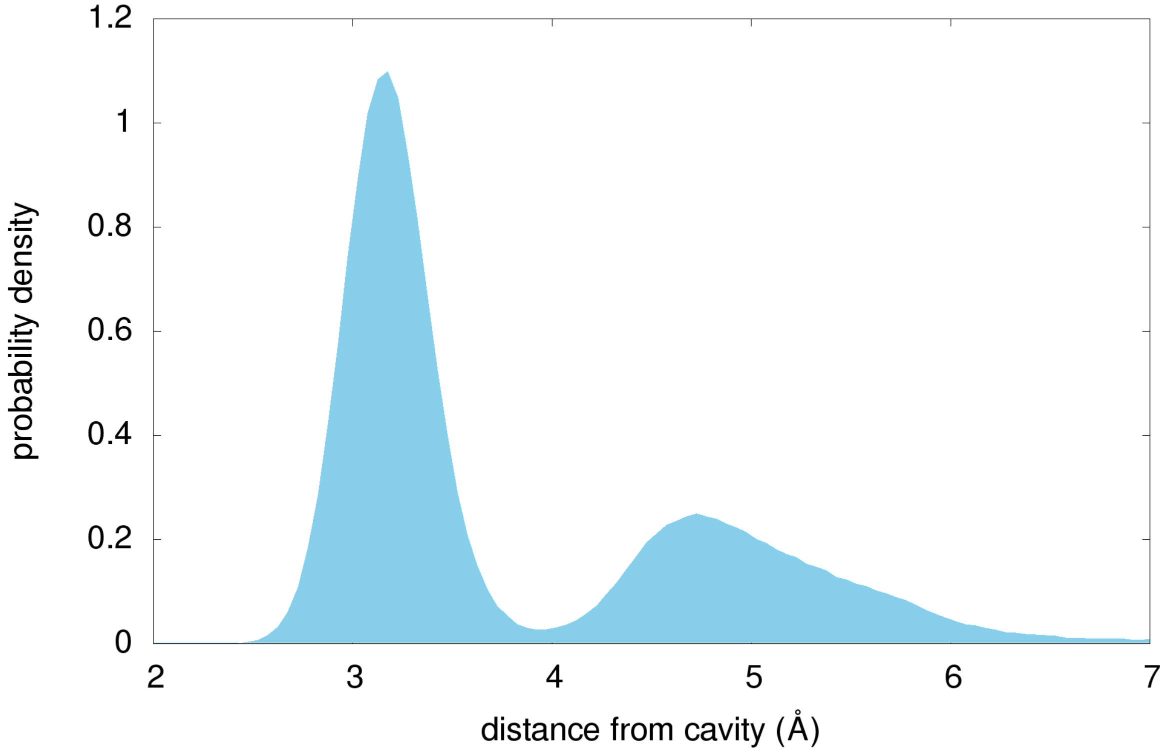


Figure 7.7 – Histogram of the distance from the cavity for the ABMD simulations.

the ligand and the cavity will increase almost linearly with the simulation time. In the presence of a metastable state, the ligand will stay trapped around a value of ξ for a certain amount of time.

A set of 20 ABMD simulations were performed. The results showed that the ligand is unbound from the protein when their distance is larger than 25 Å. The histogram of the distance taken at a fixed time interval along these simulations is in figure 7.7. The peaks of this distribution show the presence of at least two metastable states. Notice that, because of the construction of the added potential, the peaks are at higher values then the centers of the metastable states. Notice that this biased method is not intended to define precisely the metastable states, but simply gives an indication of the presence of additional metastable states.

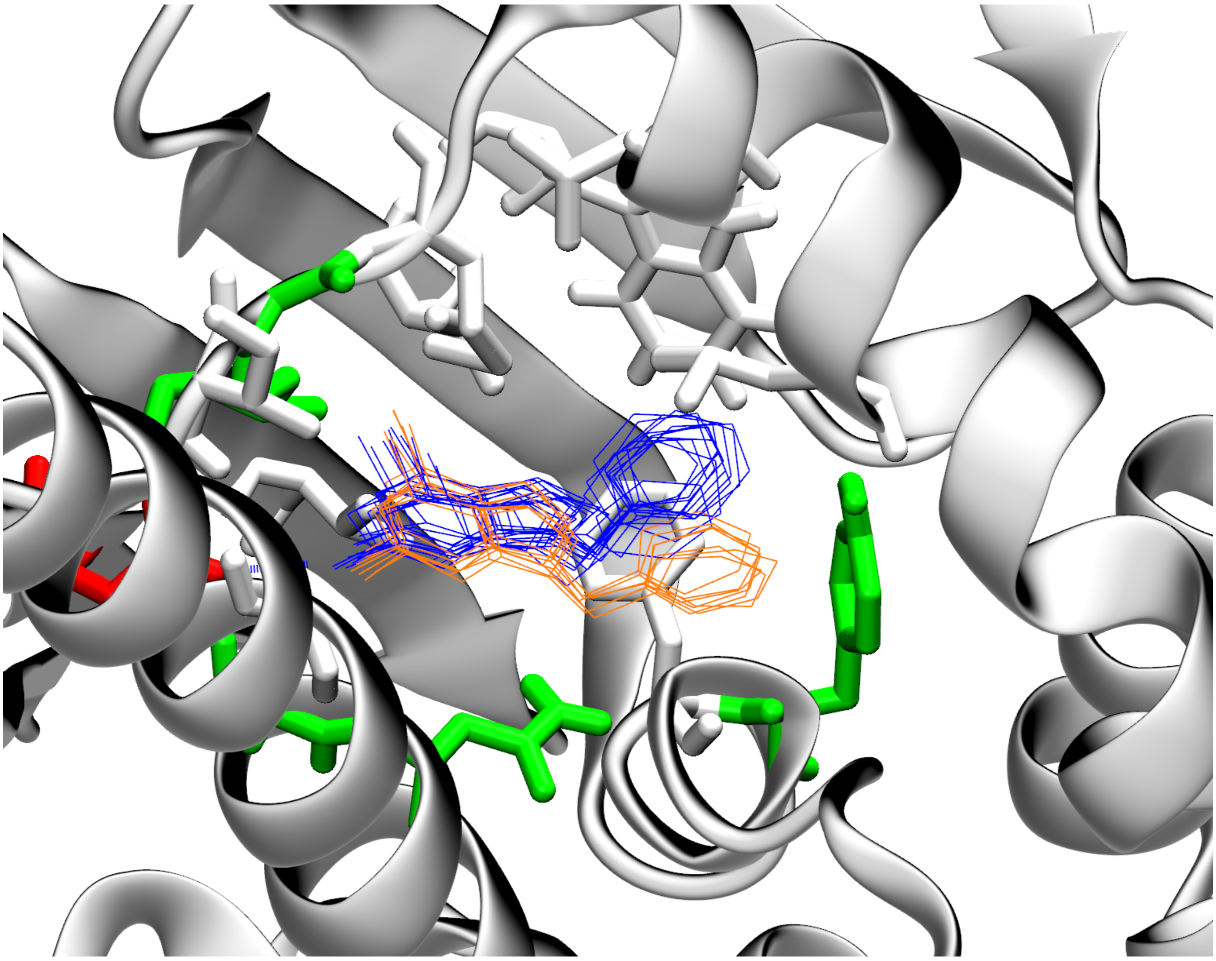


Figure 7.8 – Positions of the ligand in the bound state (in blue), and in the intermediate state (in orange).

The analysis of the generated trajectories revealed that the first peak corresponds to the bound state, and the second peak corresponds to an intermediate state for the unbinding mechanism, where the aromatic ring of the ligand detaches from the protein, but the bicycle remains bound (see figure 7.8). The presence of intermediate states is a not a problem for AMS, if their residence time is low, i.e. escaping from them is not a rare event. This was assumed to be the case here because of the low intensity of the peak in the ABMD result, compared to the one from the bound state.

7.3.1 First AMS results

Two sets of AMS simulations were run, both using $k = 1$, but with different numbers of initial points. For $N = 50$, a total of eight simulations were made. The first seven among those ended with no trajectory reaching the unbound state. Further analysis showed that the ligand was moving to the interior of the protein cavity, where no escape was possible, and yet ξ was increasing. This suggested that the use of a distance as a reaction coordinate was not adapted to the problem. The last simulation had a trajectory that could not reach the bound or unbound state after more than 800 ns, when the killing level was around 7.5 Å. For reference, all the other AMS trajectories simulated until that moment had a mean duration of 86 ps. The same was seen with another simulation, with $N = 250$. This thus indicated the presence of at least one other metastable state that trapped the ligand. From this moment we decided to stop the simulations and analyze the trapped trajectories to decide how to proceed.

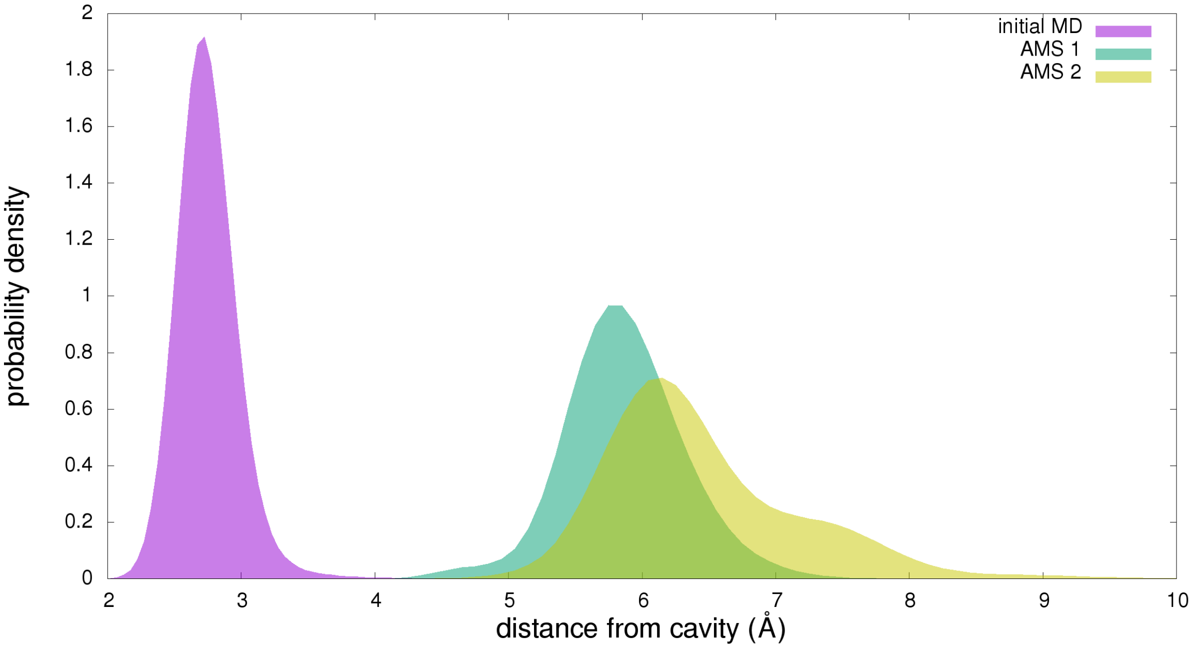


Figure 7.9 – Histogram of the ligand distance from the protein cavity during the last simulated trajectory for two AMS simulations, compared to the initial free dynamics (see figure 7.6). In AMS 1 $N = 50$, and for AMS 2 $N = 250$.

Figure 7.9 shows the histogram of the reaction coordinate, taken at a fixed timestep, for those trajectories, compared to the initial simulation, from figure 7.6. This indicated the presence of not one but two other metastable states, that seem to overlap in terms of distance to the cavity. Those could be either intermediate states, like the one seen in the ABMD simulations, or other bound states that should be taken into account in the AMS simulations. To elucidate the nature of these states, and also define a more suitable reaction coordinate, a set of new analysis and simulations were performed.

7.3.2 Analyzing metastable states to prepare new AMS simulations

In order to identify the new metastable states seen with AMS, a 2D free energy surface was calculated. To separate the intermediate state, from figure 7.8, from the bound state, we decided to look separately at the distance of both the aromatic ring and the bicycle from the cavity center. The free energy was then calculated using the sum and the difference of those distances. This was done using the adaptive biasing force (ABF) method[14, 64], that adds an adaptive force over the chosen reaction coordinates in order to pull the system to the less probable regions, and thus visiting the entire plan.

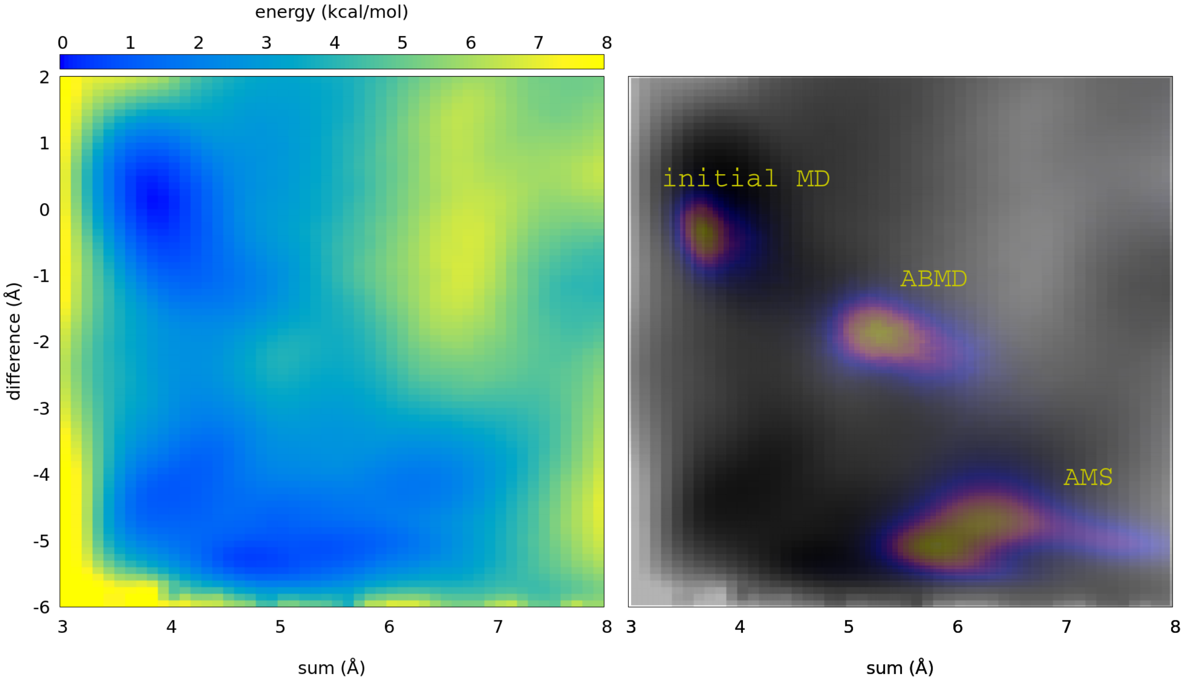


Figure 7.10 – Free energy surface over the sum of the distances of the aromatic and bicyclic structures of the ligand from the protein cavity, and their difference. On the right are the histograms of the trajectories obtained with the initial simulations, ABMD, and AMS, projected over the free energy.

Figure 7.10 show the free energy surface and the histogram within the same coordinates for the initial simulation, the two trajectories generated by AMS, and the trajectories generated by ABMD. The first conclusion is that the intermediate state, seen with ABMD, is not present in the AMS trajectories. The second is that the AMS trajectories cover the second well of the free energy surface, and thus must be included in the bound state. However, the chosen coordinates were not able to separate the new states, which is crucial to correctly define them.

In the search for a more appropriate reaction coordinate, we decided to consider the distance of the ligand from the cavity projected over an axis (see figure 7.11). This way the new reaction coordinate would decrease as the ligand goes deep into the protein cavity. Exploring rotations around this axis and others, using the Collective Variables dashboard in VMD[12], the rotation of the bicyclic ligand structure around a perpendicular axis, was able to separate the three metastable states. Figure 7.12

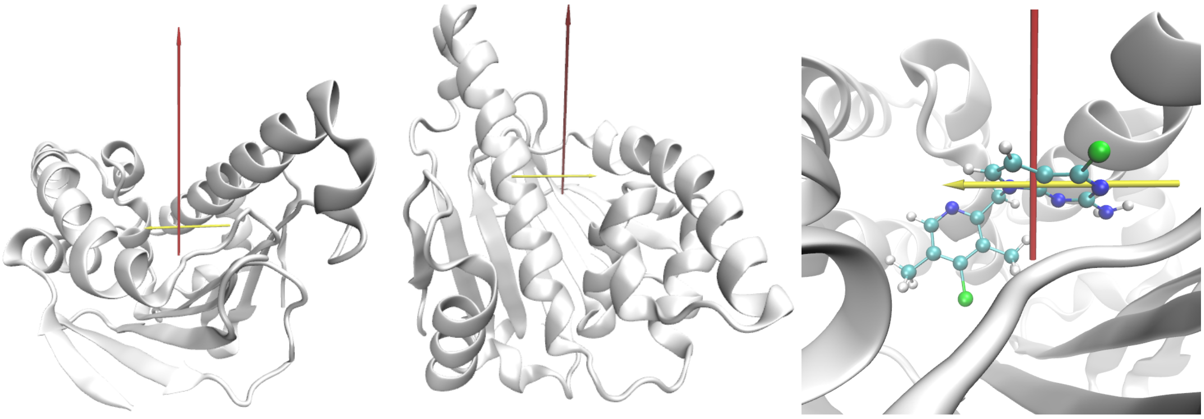


Figure 7.11 – The new coordinates used: the projection of the ligand distance to the cavity, projected onto the red axis; and the rotation of the bicyclic structure of the ligand around the yellow axis.

shows the histogram of both AMS trajectories and the initial simulation over this space.

The new bound state was defined as a union of the three states:

$$\begin{aligned} C &= \{\text{proj} \in [1.9, 2.45]\} \cap \{\text{rot} \in [-7, 7]\} \\ T_1 &= \{\text{proj} \in [2.45, 3.05]\} \cap \{\text{rot} \in [83, 108]\} \\ T_2 &= \{\text{proj} \in [4.25, 5.25]\} \cap \{\text{rot} \in [-93, -80]\}. \end{aligned}$$

To run a new set of AMS simulations we decided to use initial conditions in the border of each one of the defined states C, T_1 and T_2. For state C, the points were sampled using the trajectory generated by the initial simulation. For each of the states T_1 and T_2, the sampling was done using the respective trajectory generated by AMS. A set of four AMS simulations were launched: one for each state, using the parameters $N = 250$ and $k = 1$; and one extra for state C, with $N = 250$ and $k = 100$. Those simulations are currently running.

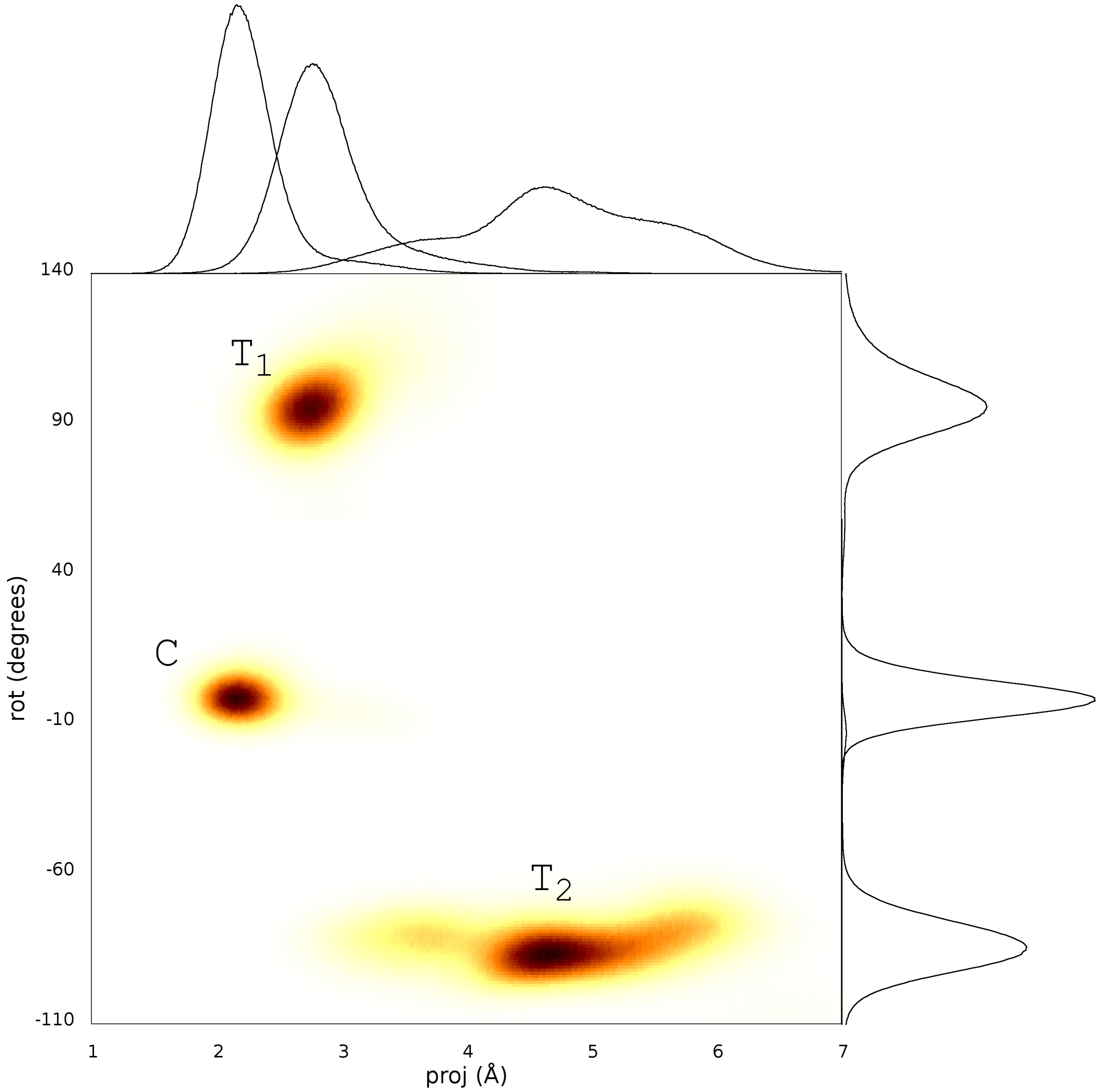


Figure 7.12 – Identification of the three states: seen with the first dynamics, which includes the crystallographic structure (C); and the ones discovered by the AMS simulations, that trapped the ligand (T_1 and T_2).

7.4 Conclusion and Perspectives

Although the first AMS simulations were not able to sample reactive trajectories, new metastable states were found. Those are accessible only after exiting the first bound state, and hence were not visited by the ABMD simulations. Therefore, only AMS was capable of revealing their presence. This shows that the AMS method can also be used in the exploration of unknown metastable states of the system.

The free energy surface showed the bound nature of those new states, indicating the necessity to change the definition of the bound state to run AMS in order to sample reactive trajectories and calculate the unbinding time. The new AMS simulations are currently running. Yet, it is also necessary to calculate a new free energy surface using the new coordinates to compute the probability of each one of the three states. Those are necessary to obtain the unbinding time as a weighted average of the time estimated for each state. This simulation is less expensive than AMS, and can be done after.

Recent published results[62] indicates a lower exit time for a protonated state of the ligand, thus suggesting that protonation would play an important role in the escape process. This can be tested by performing AMS simulations with the protonated ligand. The time of the entire process can be obtained via the estimated AMS unbinding time multiplied by the mean time for the protonation to occur, obtained through the ligand's pKa.

Conclusion and perspectives

In this thesis, we applied the AMS method to different systems, which gave us new understandings about how to use this method. We here present the most important conclusions and perspectives drawn from these simulations and analysis.

The first studied system, the conformational change in alanine dipeptide, suggests that one does not need to provide a reaction coordinate of high quality in order to obtain reliable results. This shows an important robustness of the method.

A major difficulty when using AMS concerns the sampling of the initial conditions. This was firstly seen in Chapter 2, and then further explored in Chapter 3. This problem is common to other rare event methods, because one needs to obtain the probability at equilibrium and it is not possible to reach the equilibrium to correctly sample the initial conditions. Moreover, among an ensemble of initial conditions, typically only a few of them contribute to the rare event of interest. Hence, there are actually two rare events: one related to the sampling of the initial conditions and one related to the sampling of the reactive trajectories starting from this distribution. AMS only attacks the second rare event. Taking a step back and studying a simple one dimensional case, we were able to propose a successful technique to solve the issue raised by the first rare event, linked to the efficient sampling of the initial conditions. Our solution relies on the combination of an importance sampling technique and AMS. On the simple test case we considered, we observed an important computational gain. These results are very promising for applications to large scale systems. We are now working on an implementation for the alanine dipeptide case, that should give insights on the problems we may face with complex systems.

Another difficulty that we exhibit when using AMS on real-life test cases is the definition of the origin state A. This is again a problem that should be encountered with other methods, whenever the definition of the metastable state becomes complicated. The project on Hsp90 (Chapter 7) shows that AMS can be an ally in the exploration of all the states that should be incorporated to properly define the origin state.

In the project with the β-cyclodextrin, it became clear that the reproducibility of kinetic experimental results is not easy. Despite of that, AMS was an important tool to enable the comparison of the water models. The fact that the unbinding mechanism was in essence the same for the two models we tested show some robustness of these water models, but a lot remains to be done to obtain precise quantitative results.

Because AMS samples an unbiased ensemble of reactive trajectories, there is also the possibility to obtain information about the reaction mechanisms. We propose to use clustering techniques to analyze the ensemble of reactive trajectories, and extract reaction mechanisms from it. The centers of the clusters are then treated as representative reaction mechanisms. We intend to test other variants of this methodology and perform applications to more challenging systems in the near future.

In summary, the AMS method is a robust and efficient method to simulate rare events, which relies on sound mathematical foundations, including unbiasedness results and asymptotic variance analysis. The problems seen when applying AMS to molecular dynamics are common to other rare event methods. We were able to explore them and propose solutions that shown to be good candidates for applications to complex systems. We were also able to propose a new way to explore the reaction mechanisms. Finally, the application of AMS to complex molecular systems enabled us to get a better knowledge of the molecular mechanisms on two problems of interest for the pharmaceutical industry.

Bibliography

1. T. S. Van Erp, *Dynamical Rare Event Simulation Techniques for Equilibrium and Nonequilibrium Systems* (John Wiley & Sons, Ltd, 2012), chap. 2, pp. 27–60.

2. B. J. Alder and T. E. Wainwright, J. Chem. Phys. **31**, 459 (1959).

3. F. Stillinger and A. Rahman, J. Chem. Phys. **60**, 1545 (1974).

4. H. C. Andersen, Journal of Computational Physics **52**, 24 (1983), ISSN 0021-9991.

5. R. B. Best, X. Zhu, J. Shim, P. E. M. Lopes, J. Mittal, M. Feig, and A. D. MacKerell, Journal of Chemical Theory and Computation **8**, 3257 (2012).

6. D. A. Case, T. E. Cheatham III, T. Darden, H. Gohlke, R. Luo, K. M. Merz Jr., A. Onufriev, C. Simmerling, B. Wang, and R. J. Woods, Journal of Computational Chemistry **26**, 1668 (2005).

7. C. G. Mayne, J. Saam, K. Schulten, E. Tajkhorshid, and G. J. C., J. Comput. Chem. **34**, 2757 (2013).

8. W. Humphrey, A. Dalke, and K. Schulten, Journal of Molecular Graphics **14**, 33 (1996).

9. A. D. Mackerell, M. Feig, and C. L. Brooks, J. Comp. Chem. **25**, 1400 (2004).

10. J. C. Phillips, R. Braun, W. Wang, J. Gumbart, E. Tajkhorshid, E. Villa, C. Chipot, R. D. Skeel, L. Kale, and K. Schulten, J. Comput. Chem. **26**, 1781 (2005).

11. I. Teo, C. G. Mayne, K. Schulten, and T. Lelièvre, J. Chem. Theory Comput. **12**, 2983 (2016).

12. G. Fiorin, M. L. Klein, and J. Hénin, Mol. Phys. **111**, 3345 (2013).

13. M. Marchi and P. Ballone, J. Chem. Phys. **110**, 3697 (1999).

14. J. Hénin, G. F., C. Chipot, and M. L. Klein, J. Chem. Theory Comput. **6**, 35 (2010).

15. J. Lu and J. Nolen, Probability Theory and Related Fields **161**, 195 (2015).

16. E. Vanden-Eijnden, Lect. Notes Phys. **703**, 439 (2006).

17. M. Baudel, A. Guyader, and T. Lelièvre, work in progress (2019).

18. R. Allen, C. Valeriani, and P. ten Wolde, J. Phys.-Condens. Mat. **21**, 463102 (2009).

19. C. Dellago, P. G. Bolhuis, and P. L. Geissler, *Transition Path Sampling* (John Wiley & Sons, Ltd, 2003), chap. 1, pp. 1–78.

20. E. E. Borrero and F. A. Escobedo, J. Chem. Phys. **129** (2008).

21. F. Cérou and A. Guyader, Stoch. Anal. Appl. **25**, 417 (2007).

22. C.-E. Bréhier, M. Gazeau, L. Goudenège, T. Lelièvre, and M. Rousset, Ann. Appl. Probab. **26**, 3559 (2016).

23. L. J. S. Lopes and T. Lelièvre, J. Comput. Chem. **40** (2019).

24. S. Huo and J. E. Straub, Journal of Chemical Physics **107**, 5000 (1997).

25. G. Li and Q. Cui, Journal of Molecular Graphics and Modelling **24**, 82 (2005).

26. L. Maragliano, A. Fischer, E. Vanden-Eijnden, and G. Ciccotti, Journal of Chemical Physics **125** (2006).

27. R. Zhao, J. Shen, and R. D. Skeel, Journal of Chemical Theory and Computation **6**, 2411 (2010).

28. E. M. Del Valle, Process Biochem. **39**, 1033 (2004).

29. X. Zhang, G. Gramlich, X. Wang, and W. M. Nau, Journal of the American Chemical Society **124**, 254 (2002).

30. W. L. Jorgensen, J. Chandrasekhar, J. D. Madura, R. W. Impey, and M. L. Klein, J. Chem. Phys. **926** (1983).

31. J. L. F. Abascal and C. Vega, J. Chem. Phys. pp. 1–12 (2005).

32. R. Copeland, D. Pompliano, and T. Meek, Nat. Rev. Drug Discovery **5**, 730 (2006).

33. A. Faradjian and R. Elber, J. Chem. Phys. **120**, 10880 (2004).

34. A. Rojnuckarin, S. Kim, and S. Subramaniam, Proc. Natl. Acad. Sci. U. S. A. **95**, 4288 (1998).

35. C. Velez-Vega, E. E. Borrero, and F. A. Escobedo, J. Chem. Phys. **130**, 225101 (2009).

36. T. S. van Erp and P. G. Bolhuis, J. Comput. Phys. **205**, 157 (2005).

37. F. Cérou, B.Delyon, A. Guyader, and M. Rousset, private communication (2018).

38. F. Cérou, A. Guyader, T. Lelièvre, and D. Pommier, J. Chem. Phys. **134**, 054108 (2011).

39. L. J. S. Lopes, C. G. Mayne, C. Chipot, and T. Lelièvre, NAMD tutorial (2018), URL http://www.ks.uiuc.edu/Training/Tutorials/namd/ams-tutorial/tutorial-AMS.pdf.

40. C.-E. Bréhier, T. Lelièvre, and M. Rousset, ESAIM: PS **19**, 361 (2015).

41. J. Hammersley and D. Handscomb, *Monte Carlo Methods*, Methuen's monographs on applied probability and statistics (Methuen, 1964).

42. J. Lu and J. Nolen, Probab. Theory Related Fields **161**, 195 (2015).

43. T. Lelièvre, M. Rousset, and G. Stoltz, *Free energy computations: A mathematical perspective* (Imperial College Press, 2010).

44. T. L. Hill, *Free Energy Transduction and Biochemical Cycle Kinetics* (Dover, New York, 1989).

45. D. Aristoff, ESAIM Math. Model. Numer. Anal. **52** (2018).

46. R. J. Allen, D. Frenkel, and P. R. ten Wolde, J. Chem. Phys. **124**, 194111 (2006).

47. D. Aristoff and D. M. Zuckerman, arXiv e-prints p. arXiv:1806.00860 (2018).

48. D. Aristoff, arXiv e-prints arXiv:1906.00856 (2019).

49. T. Eiter and H. Mannila, Tech. Rep., Christian Doppler Laboratory for Expert Systems, TU Vienna, Austria (1994).

50. S. Lloyd, IEEE transactions on information theory **28**, 129 (1982).

51. G. Pagès, ESAIM: proceedings and surveys **48**, 29 (2015).

52. G. Pagès and J. Printems, in *Handbok of Numerical Analysis, Vol. XV, Special Volume : Mathematical Modeling and Numerical Methods in Finance*, edited by P. G. Ciarlet (Elsevier, North Holland, 2008), pp. 595–648, guest Editors : Alain Bensoussan and Qiang Zhang.

53. T. Kohonen, Biological cybernetics **43**, 59 (1982).

54. S. Park, M. Sener, D. Lu, and K. Schulten, J. of Chem. Phys. **119**, 1313 (2003).

55. P. Metzner, C. Schütte, and E. Vanden-Eijnden, Journal of Chemical Physics **125** (2006).

56. Aristoff, David, Lelièvre, Tony, Mayne, Christopher G., and Teo, Ivan, ESAIM: Proc. **48**, 215 (2015).

57. L. J. S. Lopes and T. Lelièvre, arXiv:1707.00950 [physics.chem-ph] (2017).

58. A. V. Onufriev and S. Izadi, Wiley Interdiscip. Rev. Comput. Mol. Sci. **8** (2018).

59. M. Chaplin, *Water structure and science*, http://www1.lsbu.ac.uk/water, accessed on Janyary 2019.

60. K. Vanommeslaeghe, E. Hatcher, C. Acharya, S. Kundu, S. Zhong, J. Shim, E. Darian, O. Guvench, P. Lopes, I. Vorobyov, et al., J. Comput. Chem. **31**, 671 (2010).

61. F. H. Schopf, M. M. Biebl, and J. Buchner, Nat. Rev. Mol. Cell Biol. **18**, 345 (2017).

62. S. Wolf, M. Amaral, M. Lowinski, F. Vallée, D. Musil, J. Güldenhaupt, M. K. Dreyer, J. Bomke, M. Frech, J. Schlitter, et al., arXiv e-prints arXiv:1907.10963 (2019).

63. M. J. Frisch, G. W. Trucks, H. B. Schlegel, G. E. Scuseria, M. A. Robb, J. R. Cheeseman, G. Scalmani, V. Barone, G. A. Petersson, H. Nakatsuji, et al., *Gaussian 16 Revision C.01* (2016), gaussian Inc. Wallingford CT.

64. E. Darve, D. Rodríguez-Gómez, and A. Pohorille, J. Chem. Phys. **128**, 144120 (2008).