FREIE UNIVERSITÄT BERLIN

EFFICIENT MULTI-SCALE SAMPLING METHODS IN STATISTICAL PHYSICS

Luigi Sbailò

Dissertation
zur Erlangung des Grades eines
Doktors der Naturwissenschaften (Dr. rer. nat.)
am Fachbereich Mathematik und Informatik
der Freien Universität Berlin

Berlin, 2019

Luigi Sbailò, *Efficient multi-scale sampling methods in statistical physics*,
PhD thesis, FU Berlin, 2019.

Betreuer:
Prof. Dr. Frank Noé
Freie Universität Berlin
Fachbereich Mathematik und Informatik
Arnimallee 6
14195 Berlin

Gutachter:
Prof. Dr. Frank Noé, FU Berlin
Dr. Max von Kleist, Robert Koch Institut

Tag der Disputation:
19 Dezember 2019

Abstract

Computer simulations, with the aid of physical modeling, have enabled us to reproduce complex biological processes with steadily increasing level of detail. The different biological components are modeled as objects whose physical interactions are known, then the evolution of the system is evaluated numerically in accordance with the laws of physics. From a partial and fragmented knowledge of a biological system that is acquired with experimental techniques, it is then possible to reconstruct computationally the full picture, and predict states of the system that would not be accessible experimentally. Results that have been achieved are already numerous, spanning from the microscopic reproduction of protein-protein association [Plattner et al. - Nature Chemistry, 2017] to the mesoscopic spatio-temporal definition of metabolic networks [K. Takahashi et al. - PNAS, 2010], and promises are even greater. Detailed simulations can indeed facilitate the comprehension of many protein misfolding diseases, as Alzheimer's disease, and can give a core contribution in the construction of a synthetic cell. The field of computational biology is, in fact, in rapid expansion. This growth is being boosted by the continuous increase in computational power and technological advances in experimental biology, and results that few years ago seemed unreachable are now within our reach. Computational resources and biological knowledge currently available are incredibly vast, but the complexity of biological systems is also enormous. In fact, the most ambitious goals in this field demand a scrupulous and efficient consumption of the computational power required for simulations. The impact of the development of algorithms computationally efficient becomes, in this context, as relevant as never before.

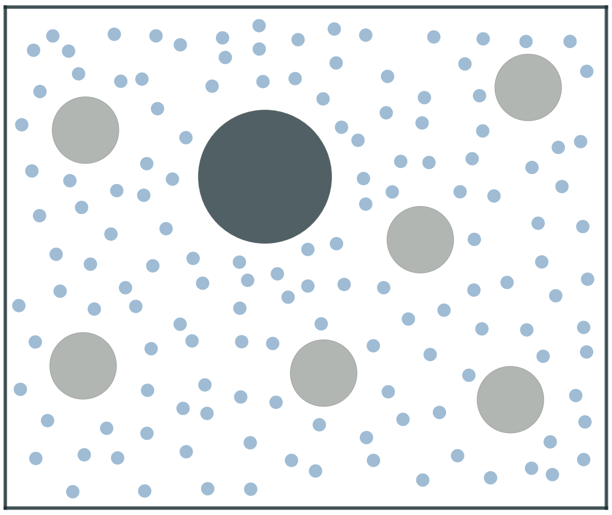
This thesis deals with the development and formalization of algorithms designed for an efficient simulation of biological systems. This work is separated into two different parts, and in each part a different algorithm is investigated. In the first part of the thesis, an algorithm that is used to simulate biological systems at the mesoscopic scale is outlined. The aforementioned algorithm is studied in detail, and several improvements, theoretical, algorithmic and technical, are presented. In the second part of the thesis, a novel sampling method is outlined, which uses deep-learning to accelerate the computation of equilibrium properties of systems defined with atomistic detail. The two parts lead to applications at different scales, and, in the future, methods and concepts developed in this thesis can be useful for the investigation of biological processes defined with mesoscopic or microscopic detail.

Zusammenfassung

Diese Arbeit beschäftigt sich mit der Entwicklung und Formalisierung von Algorithmen zur effizienten Simulation biologischer Systeme. Die Arbeit ist in zwei Teile gegliedert und in jedem Teil wird ein anderer Algorithmus untersucht. Im ersten Teil der Arbeit wird ein Algorithmus vorgestellt, mit dem biologische Systeme im mesoskopischen Maßstab simuliert werden. Dieser Algorithmus wird im Detail analysiert und es werden verschiedene theoretische, algorithmische und technische Verbesserungen vorgestellt. Im zweiten Teil der Arbeit wird eine neuartige Methode zur Erzeugung von Stichproben vorgestellt, welche mithilfe von Deep-Learning die Berechnung von Gleichgewichtseigenschaften von Systemen mit atomistischen Details beschleunigt. Die beiden Teile führen zu Anwendungen in unterschiedlichen Maßstäben. In Zukunft können die in dieser Arbeit entwickelten Methoden und Konzepte für die Untersuchung biologischer Prozesse nützlich sein, die in mesoskopischen oder mikroskopischen Details definiert sind.

" You may be sure this starts with atoms; they are what provide
The base of this unrest. For atoms are moving on their own,
Then small formations of them, nearest them in scale, are thrown
Into agitation by unseen atomic blows,
And these strike slightly larger clusters, and on and on it goes -
A movement that begins on the atomic level, by slight
Degrees ascends until it is perceptible to our sight,
So that we can behold the dust-motes dancing in the sun,
Although the blows that move them can't be seen by anyone."

(Lucretius - De Rerum Natura)



Translation by E. Stallings, Penguin UK, (2007)

Publications

Parts of this thesis have been published or are in preparation for publication:

- L. Sbailò, and F. Noé. (2017) *An efficient multi-scale Green's function reaction dynamics scheme.* J. Chem. Phys. **147**, 184106; https://doi.org/10.1063/1.5010190.
- L. Sbailò, and L. Delle Site. (2019) *On the formalization of asynchronous first passage algorithms.* J. Chem. Phys. **150**, 134106; https://doi.org/10.1063/1.5083147.
- L. Sbailò, M. Dibak, and F. Noé. (2019) *Neural Mode Jump Monte Carlo.* Arxiv:1912.05216 (submitted).

Contents

Chapter 1

Introduction

Latest technological advances in super-resolution microscopy have enabled us to probe intracellular structures with unprecedented resolutions, and this has permitted quantitative measurements of single-molecule localization [1][2][3][4]. However, the experimental investigation of the spatio-temporal correlations of molecules in biological processes is challenging, but many cellular functions can be explained only in terms of the temporal succession of molecular interactions and the exact position where interactions take place [5][6][7][8]. For example, detailed spatio-temporal definition of molecular interactions is crucial for modeling signaling pathway, where an external stimulus on a microscopic receptor triggers a cascade of chemical reactions that ultimately generates a macroscopically observable output [9][10][11]. A biochemical cascade can be reproduced as a particle-based reaction dynamics (PBRD) system where each relevant biomolecule is represented as a particle. The concerted motion of particles in PBRD simulations allows to explain the mechanisms through which information is processed in biochemical networks and to study the kinetics of the investigated system.

PBRD is one of the most detailed types of simulation [12][13][14]. In contrast to other approaches to simulate molecular kinetics, as concentration-based approaches or Gillespie's dynamics [15][16], the trajectory of all interacting particles is resolved, providing a reaction kinetics model with high spatio-temporal detail. However, cells contain an enormous number of molecules, and reaching biologically relevant time scales with detailed simulations of this number of particles remains challenging even with the most powerful processors. Hence, an approximation is required, and only the motion of larger, or mesoscopic, biomolecules is simulated, while the contribution of smaller molecules is approximated as an average effect. At the mesoscopic scale, particles diffuse according to the Langevin equation (see section 2.1 for a more detailed discussion). Particle motion is then integrated from a brute-force discretization of the Langevin equation, and whenever particles get close to each other, reactions can happen.

PBRD simulations have already seen many applications to biochemical networks (some representative examples can be found in Refs. [17][18][19][20]), and there is a wide range of simulation tools for PBRD [21][22], including ReaDDy [23], Smoldyn [24], MCell [25], Cell++ [26], eGFRD [8], mesoRD [27], spatiocyte [28], SpringSaLaD [29], and SR-Sim [30]. However, an evident drawback for detailed simulations like PBRD is the heavy computational effort that is required to simulate the full dynamics of all tracked particles. This problem is particularly evident when dilute systems are simulated, where the time particles require to encounter each other can be long, and investigating the reaction kinetics of the system becomes computationally demanding. Indeed, in this case, most of the computational time is dedicated to the simulation of free diffusion of particles, while only a fraction of it is dedicated to particle interactions. The reason for this is given by the difference between the time scale of reactions and the time that molecules need to find each other when freely diffusing. The exact dynamics of particles during free diffusion is not particularly interesting, while it is critical to know the exact position and time where interactions take place. It would be ideal to skip irrelevant computations during free particle diffusion to instead dedicate computational resources for molecular reactions. This, in fact, is the crux idea of Molecular Dynamics - Green's Function Reaction Dynamics (MD-GFRD) [31], which is a multi-scale PBRD algorithm that generates large spatio-temporal displacements of free particles by computing the exact position and time where the next molecular interaction takes place.

MD-GFRD is an asynchronous algorithm that alternates event-based particle propagations over long time scales and local time-driven particle interactions. Asynchronous event-based particle propagations avoid detailed simulation of the free diffusion, and computational time is mainly dedicated to the time-driven integration of particle interactions. While synchronous time-driven PBRD simulations treat reactions and free diffusion within the same integration scheme, i.e. time discretization of the Langevin equation, MD-GFRD separates the two processes. This distinction allows for the incorporation of a variety of effects during reactions, which are relevant at the molecular level. Not only can reactions be defined with a basic binary reaction scheme [32], but can also include anisotropic reactions [33][34], or interaction potentials [14], and would be a natural place to include the dynamics simulated by kinetic models obtained from all-atom MD, e.g., Markov State Models (MSMs) [35][36][37][38][39] or multi-ensemble Markov models (MEMMs) [40][41]. Differently, isolated non-interacting particles are displaced with an event-based propagation. This is attained with the construction of protective domains around free particles, then first exit-times from the domains are computed and particles are placed onto the domain borders at the sampled times. Each domain contains only one particle, the underlying stochastic dynamics are then analytically tractable and it is possible to solve the relative first exit-time problem. MD-GFRD algorithm has been proven to be several orders of magnitude faster than brute-force time-driven integrations [31][33].

In the first part of this thesis, MD-GFRD is explained from an algorithmic and theoretical perspective. It is shown that the criteria used to select the size of protective domains are relevant for computational performance. A new scheme for the domain size selection is proposed, and it is shown that it provides a tenfold improvement in computational performance. Subsequently, the probability distribution describing the particle position within protective domains is derived analytically. In MD-GFRD, particle position might be updated prematurely before escaping the domain, and the effects of exit-time boundary conditions on the position sampling are investigated. It is demonstrated that exit-time boundary conditions can be ignored only under specific conditions, and the formalization of the algorithm is put on a more solid theoretical ground. Finally, an efficient method to effectively sample the probability distributions used in MD-GFRD is presented.

The second part of this thesis deals with Markov chain Monte Carlo (MCMC), a powerful and versatile method, also considered one of the 10 most important algorithms in the twentieth century [42]. MCMC methods are used to sample probability distributions by constructing Markov chains that, under specific conditions, asymptotically sample the equilibrium distribution [43]. Possible applications are wide and extend to all stochastic systems, naturally also including biological systems. Statistical mechanics can indeed be used to explain processes that take place at the molecular and cellular level in biology, allowing for a stochastic description of these processes. Each component of the system is described as a different atom, and a certain probability is associated to every possible configuration of the atoms. Macroscopic properties are then evaluated from a representative sample of the microscopic configurations of the system. MCMC methods are rather simple

to implement, that have also contributed to increasing their popularity and range of applications. However, these methods suffer from the curse of dimensionality, because, as dimensionality is increased, the fraction of physically meaningful microscopic configurations becomes vanishingly small within the configuration space. Local MCMC can be used to explore locally relevant conformations of the system, but this approach generates highly correlated samples, and many samples are required before convergence to equilibrium is reached. In this thesis, a novel MCMC method - neural mode jump Monte Carlo (neural MJMC) - is presented. This method makes use of deep learning to accelerate convergence of the Markov chain, and it is combined with local MCMC to preserve statistical consistency. It is then applied to a non-trivial multi-dimensional distribution, and it is shown that this method increases the generation rate of independent samples by many orders of magnitude with respect to local MCMC. Neural MJMC is general and transferable, and can be applied to any stochastic system.

Part I

Mesoscopic Scale

Chapter 2

Modeling the motion of mesoscopic particles

Complex biological functions in living organisms are explainable in terms of the coordinated interaction of elementary microscopic components. Microscopic elements in biological organisms are biomolecules, which carry different and specific functionality, and their sizes span different orders of magnitude in length. Proteins are large biomolecules, and they play a fundamental role in living organisms being involved in essentially every aspect of cellular life. The functions of proteins are various, and are mainly determined by their structure. For example, proteins are responsible for signal transduction in living organisms. A sequence of protein interactions permits an external signal to be transmitted through a cell, which ultimately results in a cellular response. Cells contain an enormous number of molecules, but signal transduction, as many other biological functions, can be cast in a simplified model, where only protein interactions are relevant to describe the mechanisms through which information is processed. Proteins diffuse in a bath of smaller microscopic molecules whose exact motion is not relevant and, when close to each other, interact. Their interaction triggers a succession of events that finally produces a macroscopically observable result.

We aim to reproduce in detail the aforementioned processes through computer simulations. Assuming that particle-particle interaction is known and is expressible as an interaction potential, the dynamics of particles are described by Newton's equations. Computers are a natural tool to facilitate the integration of particle dynamics, but even the most powerful processors must cope with limiting factors due to the number of particles to be simulated. Models of biological systems can include trillions of different particles, and a numerical solution of systems of this size is clearly computationally unfeasible. Hence, simplifications based on physically reasonable assumptions are required. A reasonable modeling assumption is then to ignore the exact dynamics of microscopic particles, and to consider their contribution as an average effect on the dynamics of larger mesoscopic particles. The name mesoscopic (from ancient Greek $\mu\epsilon\sigma o\zeta$ - middle) is used to designate particles much larger than microscopic particles, which are treated as a bath, but not large enough to be macroscopically observable. The Langevin equation is perfectly suited for this purpose.

2.1 Langevin Equation

Each molecule in the system is modeled as a spherical particle, and the interaction between particle i and particle j is reproduced with an interaction potential $V_{ij}(r_{ij}, t)$, where r_{ij} is the distance between the particles. The motion of particle i is described by Newton's second equation

$$m\frac{d\vec{v}_i(\vec{x}_i, t)}{dt} = \sum_{j}^{N_{micro}} \vec{F}_{ij}(r_{ij}, t), \tag{2.1}$$

where the force $\vec{F}^{ij}(r_{ij}, t)$ is derived from an interaction potential $\vec{F}^{(ij)}(\vec{r}_{ij}, t) = -\vec{\nabla} V_{ij}(r_{ij}, t)$, and the sum is iterated over all

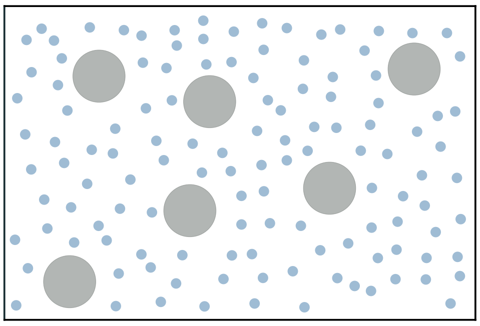


Figure 2.1: Langevin equation is used to describe the motion of mesoscopic particles (grey particles) diffusing in a bath of microscopic particles (blue particles). Biological systems can contain en enormous number of microscopic particles, and solving numerically their motion is often computationally unfeasible. The main assumption of the Langevin equation is that the dynamics of microscopic particles are not numerically integrated, but their contribution is treated as a stochastic effect on the motion of mesoscopic particles.

$j = 1, \ldots, N_{micro}$ particles within interaction distance. The core assumption in the Langevin equation is that only the interaction between mesoscopic particles is resolved (see Fig. 2.1), while the contribution of microscopic particles is treated as an average effect

$$m\frac{d\vec{v}_i(t)}{dt} = \vec{F}_{micro}(\vec{x}_i, t) + \sum_{j}^{N_{meso}} \vec{F}_{ij}(r_{ij}, t), \tag{2.2}$$

where $\vec{F}_{micro}(t)$ is the resulting force exerted by all microscopic particles on the mesoscopic particles, and N_{meso} is the number of mesoscopic particles interacting with particle i. The contribution of microscopic particles is separated into two different components: (i) a deterministic hydrodynamic friction $\vec{F}_{hydr}$, which resists motion; (ii) a stochastic force $\vec{F}_{rand}(t)$ characterizable through a probabilistic description

$$m\frac{d\vec{v}_i(t)}{dt} = \vec{F}_{hydr}(t) + \vec{F}_{rand}(t) + \sum_{j}^{N_{meso}} \vec{F}_{ij}(r_{ij}, t). \tag{2.3}$$

The hydrodynamic friction is described through Stokes' law, which tells us that the fluid contributes with a viscous force $\vec{F}_{hydr}(t) = -\alpha\vec{v}_i(t)$ with opposite direction with respect to the particle velocity $\vec{v}(t)$, where α is the dumping coefficient $\alpha = 6\pi a\eta$, and a, η respectively are the object radius and the fluid viscosity. The motion of mesoscopic particles is then described by the Langevin equation

$$m\frac{d\vec{v}_i(t)}{dt} = -\alpha\vec{v}_i + \vec{F}_{rand}(t) + \sum_{j}^{N_{meso}} \vec{F}_{ij}(r_{ij}, t). \tag{2.4}$$

We assume that fluid is dense with respect to the particle mass, which is reasonable for mesoscopic particles. Hence, inertial effects are safely disregarded, and this brings us to the overdamped Langevin equation, also known for describing Brownian dynamics

$$\vec{v}_i = \frac{1}{\alpha}\vec{F}_{rand}(t) + \frac{1}{\alpha}\sum_{j}^{N_{meso}} \vec{F}_{ij}(r_{ij}, t). \tag{2.5}$$

The position of the mesoscopic particle is observed at time intervals of length long enough to include many random collisions, and the central limit theorem states that the sum of independent random variables tends asymptotically toward a normal distribution. Hence, if we assume that collisions against mesoscopic particles are numerous and uncorrelated, the resulting force exerted by microscopic particles $\vec{F}(t)$ is approximated by a Gaussian process

$$\left\langle \vec{F}_{rand}(t) \right\rangle = 0, \qquad \left\langle \vec{F}_{rand}(t_1)\vec{F}_{rand}(t_2) \right\rangle = \sigma^2\delta(t_1 - t_2), \tag{2.6}$$

where σ is the variance of the Gaussian and gives the strength of the fluctuating force. We have assumed that stochastic fluctuations of the mesoscopic particles are the consequence of microscopic collisions. We also know that the fluid friction is the result of the resistance of the microscopic particles against the movement of the mesoscopic particles. The friction term and the stochastic fluctuations must then be connected, and this, in fact, is the result of the fluctuation-dissipation theorem [44]

$$\sigma^2 = 2k_B T\alpha, \tag{2.7}$$

where T is the temperature and k_B is the Boltzmann constant. Fluctuations increase with the temperature, which is natural if we consider that the temperature measures the kinetic energy of the fluid, and that fluctuations are produced by the momenta of the microscopic particles. Introducing the diffusion coefficient

$$D = k_B T\alpha, \tag{2.8}$$

and redefining the stochastic process as a Gaussian process with unitary variance $\tilde{F}_{rand}(t) = \sigma F_{rand}(t)$, the overdamped Langevin equation becomes:

$$\frac{d\vec{x}_i(t)}{dt} = \sqrt{2D}\tilde{F}_{rand}(t) + \frac{D}{K_B T}\sum_{j}^{N_{meso}} \vec{F}_{ij}(r_{ij}, t). \tag{2.9}$$

A time-driven discretization of the above equation allows for a brute-force integration of the particle dynamics

$$\vec{x}_i(t+dt) = \vec{x}_i(t) + \sqrt{2Ddt}\,\xi(t) + \frac{D}{K_B T}\sum_{j}^{N_{meso}} \vec{F}_{ij}(r_{ij}, t)\,dt, \tag{2.10}$$

where dt is the integration step and ξ is a random number extracted from a normal distribution.

2.2 Multi-scale problem

The computational time that is required to simulate a definite biological time scales linearly with the integration step dt in Eq. (2.10). The length of the integration step gives the efficiency of the simulation, i.e. computational time per unit of biological time, and efficiency is increased by enlarging the time increment. On the other hand, simulations must also be accurate, especially during interactions, and this is achieved by decreasing the length of the integration step. Hence, the length of the integration step should be optimized so as to select the largest possible time increment to avoid missing critical information in the simulation. In PBRD simulations, particles interact when in proximity to each other, and integrations must be sufficiently short to avoid missing particles' interaction during motion. Hence, the choice of the integration step must consider both the diffusivity and the size of the particles. This means that fast particles, i.e. those with a large diffusion coefficient, require a short integration step, and small particles also require a short integration step. Considering that the diffusion coefficient decays linearly with the particle radius [1] (see Eq. (2.8)), the particle size gives a twofold contribution into the choice of the integration step length.

In a multi-scale simulation with many particle types of different size, different time increments are associated to different particle types. Among all possible values, a trivial choice is to select a fixed value for all particles, and this is given by the fastest particles. However, a fixed value for all particles involves that large particles, slowly moving at each iteration, require many iterations before encountering another

[1]An exact relationship between particle size and diffusivity was possible because particles were assumed perfectly spherical, and this is clearly a strong assumption especially for proteins with a highly irregular conformation. However, the intuition that large proteins diffuse slower than small proteins, that can be derived solely from theoretical speculations, is also corroborated by experimental evidence [45].

particle. This computational overhead can be particularly severe in dilute systems, but it can also be avoided if freely diffusing particles are allowed to perform large spatial displacements, i.e. the length of displacement is not directly determined by the fixed integration step. Large displacements must be performed accurately to avoid missing particle interactions, and, to facilitate that, domains are drawn around free particles so as to assume that particles do not interact while diffusing within the domain. The largest possible displacement within a domain is a displacement that brings the particle directly onto the domain borders, and this is verified at a time given by the first exit-time from the domain. First exit-times from regular domains can be computed with a probabilistic description of the particle position, which is provided by Einstein model to describe the motion of a mesoscopic particle.

2.3 Einstein equation

Diffusion of mesoscopic particles can be modeled using an approach developed by Einstein before the formalization of the Langevin equation. The Brownian motion was firstly observed in the modern era [2] by Robert Brown, a botanist that was observing pollen grains to understand the mechanism by which grains move in the fertilization process. A physical explanation of this motion was provided by Einstein in 1905 [46], and his solution was based on the intuition that pollen grains were mesoscopic particles surrounded by many microscopic particles. The motion of pollen grains is caused by the random collisions with

[2]An atomistic interpretation of the dust motion was already given by Lucretius in "De Rerum Natura" in 60 BC. The random motion of dust in the air was used by Lucretius to justify the existence of atoms. Dust motion was indeed the result of underlying movements of atoms that are not directly visible. This interpretation brings us back to the microscopic and mesoscopic distinction of particles, and largely anticipates Brown’s observations.

microscopic particles, and it could be explained only probabilistically because of the large number of collisions. Each observation of the mesoscopic particle position is performed after a time interval τ that is assumed large enough to contain many random collisions. After this time interval, the particle experiences a position displacement $\vec{\Delta}$ due to all microscopic collisions. Considering their stochastic nature, a displacement $\vec{\Delta}$ in a time interval τ is described by the probability distribution $\phi_\tau(\vec{\Delta})$

$$\int_{-\infty}^{\infty} \phi_\tau(\vec{\Delta})\, d\vec{\Delta} = 1, \qquad \phi_\tau(\vec{\Delta}) > 0, \quad \forall\vec{\Delta}. \tag{2.11}$$

Probabilities are assumed symmetric with respect to the displacement direction

$$\phi_\tau(\vec{\Delta}) = \phi_\tau(-\vec{\Delta}). \tag{2.12}$$

Within this stochastic model, particle position is naturally described by a probability distribution $p(\vec{x}, t)$. Particle position after a time interval τ is found by integrating over all possible paths starting from the previous configuration

$$p(\vec{x}, t+\tau) = \int_{-\infty}^{\infty} p(\vec{x}+\vec{\Delta}, t)\phi_\tau(\vec{\Delta})d\vec{\Delta}. \tag{2.13}$$

If we assume that the time interval τ is small, the probability distribution of the position is approximated with a first-order Taylor expansion in time

$$p(\vec{x}, t+\tau) = p(\vec{x}, t) + \tau\,\frac{\partial p(\vec{x}, t)}{\partial t}. \tag{2.14}$$

The displacement $\vec{\Delta}$ is also assumed small, and a second-order Tailor expansion in position approximates the particle probability distribution

$$p(\vec{x}+\vec{\Delta},t) = p(\vec{x},t) + \vec{\Delta}\frac{\partial p(\vec{x},t)}{\partial \vec{x}} + \frac{\vec{\Delta}^2}{2!}\frac{\partial^2 p(\vec{x},t)}{\partial \vec{x}^2}. \tag{2.15}$$

The Taylor expansions are inserted into the integral in Eq. (2.13)

$$\begin{aligned} p(\vec{x},t) + \frac{\partial p(\vec{x},t)}{\partial t}\tau =& p(\vec{x},t)\int_{\infty}^{\infty}\phi_\tau(\vec{\Delta})d\vec{\Delta}+ \\ &+\frac{\partial p(\vec{x},t)}{\partial \vec{x}}\int_{\infty}^{\infty}\vec{\Delta}\,\phi_\tau(\vec{\Delta})d\vec{\Delta}+ \\ &+\frac{\partial^2 p(\vec{x},t)}{\partial \vec{x}^2}\int_{\infty}^{\infty}\frac{\vec{\Delta}^2}{2}\phi_\tau(\vec{\Delta})d\vec{\Delta}. \end{aligned} \tag{2.16}$$

The first-order term in the right side $\int_{\infty}^{\infty}\vec{\Delta}\,\phi_\tau(\vec{\Delta})d\vec{\Delta}$ of the equation cancels for symmetry (see Eq. (2.12)), and, considering the normalization property $\left(\int_{-\infty}^{\infty}\phi_\tau(\vec{\Delta})\,d\vec{\Delta}=1\right)$, $p(\vec{x},t)$ cancels in both sides of the equation. The final result is the diffusion equation

$$\frac{\partial p(\vec{x},t)}{\partial t} = D\frac{\partial^2 p(\vec{x},t)}{\partial x^2}, \tag{2.17}$$

where D is the diffusion coefficient

$$D = \frac{1}{\tau}\int_{-\infty}^{\infty}\frac{\vec{\Delta}^2}{2}\phi_\tau(\vec{\Delta})d\vec{\Delta}. \tag{2.18}$$

The probability distribution of a mesoscopic particle freely diffusing, then satisfies the diffusion equation. However, in this mathematical model forces have not been defined, and motion of the mesoscopic particle is only due to the effect of microscopic collisions. Differently, Langevin's equation, that according to Einstein is also "infinitely more simple"[47], describes particle dynamics from Newton's laws of motion.

2.4 Combining Langevin integration and Einstein diffusion

The two mathematical models used to describe Brownian motions are different, in that one fully describes particle position in probabilistic terms, the other adds stochasticity as an additional term in the second Newton's equation. The two derivations are identical if the mesoscopic particle is freely diffusing, i.e. if there is no external potential in the overdamped Langevin equation. Because of the highly non-linear terms in the particle interaction, the Langevin equation is solved numerically. Einstein's equation instead allows for studying analytically the particle position as a flux of probability over longer time intervals. As already mentioned, time discretization of the Langevin equation is not efficient in dilute and multi-scale systems, where most of the computational power is used to simulate the diffusion of non-interacting particles. On the other hand, the stochastic motion of freely diffusing particles can be treated analytically with Einstein's probabilistic description. These two approaches are then combined to conceive a multi-scale scheme that treats particles interaction exactly with the Langevin equation, and efficiently solves free particle dynamics with Einstein's equation. This combined scheme is Molecular Dynamics - Green's Function Reaction Dynamics (MD-GFRD) algorithm, and it is outlined in the next chapter.

Chapter 3

MD-GFRD

Text and figures in this chapter have been previously published in:
L. Sbailò, and F. Noé. (2017) J. Chem. Phys. **147**, 184106.

Molecular Dynamics - Green's Function Reaction Dynamics (MD-GFRD) algorithm is a multi-scale scheme developed to efficiently simulate the dynamics of Brownian particles. The crux idea of this algorithm is that particle dynamics can be treated analytically in case of free diffusion, while a brute-force integration is required when particles are interacting. In MD-GFRD, interacting particles, i.e., particles that are close in space, are simulated via short time steps, whereas isolated particles are propagated via an event-based scheme on longer time scales. Spherical domains are drawn around isolated particles such that domains are not overlapping and contain only one particle each, the underlying stochastic dynamics are then analytically treatable. First exit-times are sampled in each domain and inserted in a time ordered event list.

The event list is the core of the asynchronous event-based algorithm: at every step, the system jumps to the next event and the list gets updated with a new event. Domain escape events must also be combined with a time-driven integration of the Langevin equation to define a multi-scale scheme. In this chapter, it is outlined how these two particle propagation methods are combined into a unique multi-scale algorithm as MD-GFRD. Domain escapes and particle updates within the domain are sampled from distributions that are derived analytically in the next chapter.

3.1 Event-based propagation of freely diffusing particles

The volume of the system is decomposed into many disjoint sub-volumes each containing one particle, and sub-volumes represent protective domains within which a particle can freely diffuse. For simplicity, domains Ω are spheres of radius b, and the domain size is chosen such that domains contain only one particle and the whole sphere volume is not subject to any external potentials, i.e. the interaction of other particles, membranes, etc. The free diffusion of Brownian particles is described by a probability distribution that satisfies the Einstein equation

$$\frac{\partial f(\vec{r},t)}{\partial t} = D\,\Delta f(\vec{r},t), \tag{3.1}$$

where $f(\vec{r},t)$ is the probability distribution of a Brownian particle with diffusion coefficient D, $\vec{r} = (r,\, \theta,\, \phi)^{\top}$ is the position of the particle and Δ is the Laplace operator in spherical coordinates. Given the spherical symmetry of this problem, the evolution of the probability distribution can be described by the radial function $p(r,t)$, which represents the probability of being on any point on the surface of a sphere of radius r. The radial probability of being at a radius $r < b$, without having previously hit the domain border b, is computed by imposing absorbing boundary conditions on the domain borders $\partial\Omega$,

$p_\Omega(b,t) = 0$ [48]. By imposing this boundary condition and the initial condition $p(r_0,t_0) = \delta(r_0)$ on Eq. (3.1), we obtain:

$$p_\Omega(r,t|r_0=0,t_0) = \frac{1}{S(t)} \sum_{m=1}^{\infty} \exp\left\{ - m^2 \frac{\pi^2 D}{b^2}(t-t_0)\right\} \frac{2\pi r}{b^2}\, m \,\sin\left(\frac{m\pi r}{b}\right), \qquad r < b. \tag{3.2}$$

$S_\Omega(t)$ is the survival probability

$$S_\Omega(t|r_0=0,t_0) = -2\sum_{n=1}^{\infty}(-1)^n \exp\left\{ - n^2\frac{\pi^2 D}{b^2}(t-t_0)\right\}, \tag{3.3}$$

which represents the probability that the particle is inside the domain Ω at t, without having previously hit the borders. The first exit-time probability $q(t)$ is defined *via* the survival probability $S(t)$

$$\begin{aligned} q_\Omega(\tau|r_0=0,t_0) &= -\frac{dS_\Omega(\tau)}{d\tau} = \\ &= -2\sum_{n=1}^{\infty}(-1)^n \exp\left\{ - n^2\frac{\pi^2 D}{b^2}(\tau-t_0)\right\}\frac{n^2\pi^2}{b^2}D, \end{aligned} \tag{3.4}$$

and it gives the probability that the particle escapes its domain for the first time at τ.

In this derivation, it has been assumed that no other particle enters the domain, and the particle inside the domain is not subject to any external potentials or forces (e.g. exerted by particles near the domain). However, in a multi-particle simulation this assumption is not always valid. Let us assume that at t_0 a protective domain has been constructed around an isolated particle, and this particle has sampled a first exit time $t_0 + \tau$ from its domain. In this situation, it is possible that an external particle, whose motion is brute-force integrated, is in proximity to the first domain at a time, $t_1 < t_0 + \tau$, *i.e.* before the escape time. The first exit time τ has been sampled assuming that no other particle interacts with the domain, hence the intrusion of another particle before that time would make the sampling of the particle escape time invalid. Consequently, to ensure that particles in protective domains are freely diffusing, a burst radius is defined for each pair of particles to be at least the interaction length between the intruding particle and the particle in the domain. Whenever a particle approaches a protective domain to a distance below the burst radius the domain is burst, i.e. destroyed. In that event, the particle position is updated inside the domain $t = t_1$, and the clock of the two particles is synchronized to t_1. The particle position inside the domain is sampled directly from eq. (3.2). This is possible because the bursting-time and the previously sampled exit time are assumed to be uncorrelated.

3.2 Algorithm outline

In MD-GFRD, the particle propagation is performed alternatively via direct time-step integrations or event-based first exit-time samplings. The choice of the propagation method depends on the system configuration and, in particular, whether the particle is freely diffusing or interacting with other particles. At each iteration of the algorithm, one particle is selected from a time-ordered event-list. If this particle is not interacting with other particles, the construction of a protective domain is attempted. Domain constructions are not always accepted because constructing a domain and sampling an event in it is computationally more demanding than performing few brute-force Brownian motion steps. Therefore, one typically avoids the construction of very small

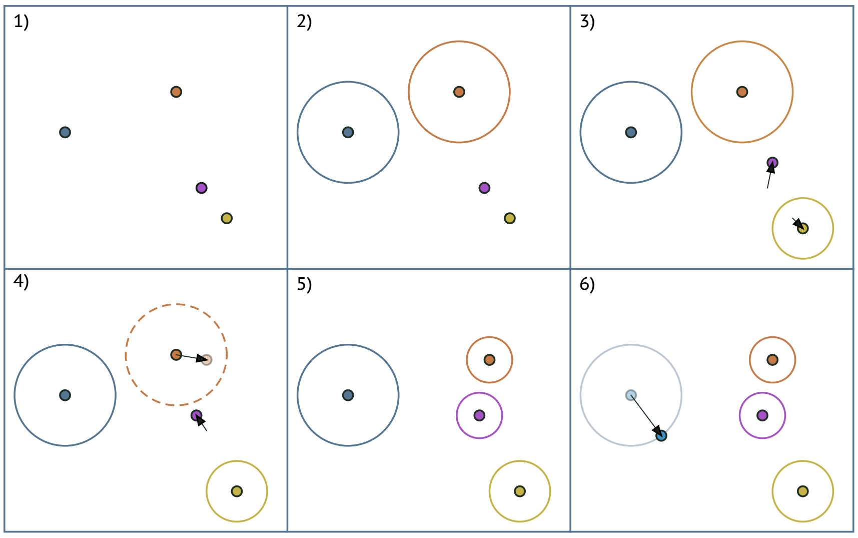


Figure 3.1: Outline of the multi-scale MD-GFRD algorithm. 1) Particles are placed in their initial configuration. 2) A protective domain is drawn on all those particles that are not directly interacting. Domains are effectively constructed only if their size is larger than the particle's minimal domain size (blue and orange particles). 3) Particles which don't have a protective domain are integrated with direct time steps (purple and yellow particles), and as soon as a particle becomes sufficiently distant from all others, a protective domain is generated around it (yellow particle). 4) When a particle gets too close to a domain (purple particle), this is burst and the inside particle samples a new position (orange particle). 5) After a domain burst, the new particle position can be sufficiently far from the intruding particle to allow both particles to construct a protective domain (orange and purple particles). 6) The global time advances to the next time from the scheduled exit-times, and the exiting particle position is sampled randomly from all points on the previous protective domain (blue particle).

domains and instead uses direct time step integration when the size of a newly constructed domain is below the minimal domain size. The construction is then accepted only if the domain radius is larger than the minimal domain size, whenever the construction is rejected the particle motion is instead brute-force integrated. Particles are then distinguished between GFRD particles, i.e. particles within a domain, and MD particles, i.e. particles undergoing a brute-force integration.

In this scheme, particle interactions are always evaluated on discrete times $\{t_n\}$, where $t_n = n\,dt$, n is an integer, and dt is the MD integration step. Therefore, a GFRD particle, that leaves a protective domain and thus becomes an MD particle is mapped to the next discrete time via a small Brownian motion step. MD particles that are evaluated at the same time point t can be updated simultaneously and collectively as in usual MD implementations. In the following pseudocode, however, it is simpler to explain the algorithm as if all particles are treated by an asynchronous event list.

Each particle possesses a *current* and a *scheduled* position and time. Each particle is also associated with an event, that takes the particle from its current position and time to its scheduled position and time, if it is successfully executed. Events include MD integration steps and scheduled exits from a protective domain, but they may be modified due to events such as domain bursting. In the beginning of the simulation, the domain making algorithm creates a protective domain for each particle that is not involved in a direct interaction. Domains larger than the minimal domain size ρ are constructed, and first exit times are sampled via Eq. (3.4). These exit events are then stored in a list ordered by increasing scheduled-time. All particles that could not construct a protective domain are placed on top of the event-list, forces between them are computed and their scheduled positions are computed and stored. Based on this initial list, the following asynchronous algorithm propagates the system state in time:

1. Pick the first particle i in the event-list:
   (a) If the particle was in a protective domain: place it on a position sampled uniformly at random on the domain boundary. Then, propagate it to the next discrete time t_i via a free Brownian motion sampling.
   (b) Else: update the particle position and time to the stored scheduled position and time.
2. Compute the distances $\{r_{ij}\}_{j=1}^{N}$ from the N neighboring particles. The distances are between the centers of mass and are computed between synchronous positions when particles are not located in a protective domain; otherwise, the distance between the center of mass of the particle i and the center of the protective domain of the particle j is computed.
3. For all $j = 1, ..., N$: if the particle j is in a protective domain and the $i-j$ distance is below the burst radius ($r_{ij} - r_j < R^{i}_{burst}$, where r_j is the domain size of the particle j):
   (a) Burst the j-domain.
   (b) Synchronize the scheduled time of particle j to t_i and update the scheduled position of particle j by sampling from Eq. (3.2).
   (c) Place particle j on top of the event-list.
   (d) Update the r_{ij} distance.
4. Use the distances $\{\widetilde{r}_{ij}\}_{j=1}^{N}$, where $\widetilde{r}_{ij} = r_{ij} - R^{ij}_{int}$ and R^{ij}_{int} is the interaction length, in a domain making algorithm to create a domain with radius r_i:
   (a) If the proposed radius is larger than the minimum domain size, $r_i > \rho_i$: accept the domain, sample the first exit time

τ_i from Eq. (3.4), and increase the particle event time by τ_i.

(b) Else: Update the scheduled position and scheduled time via direct time-step propagation (this step might also involve interactions and reactions).

5. Place the particle i in the event-list according to increasing event time.

Note that if particles i and j construct domains that are in contact and if following step 1a these particles have identical scheduled discrete exit times, it is possible that the particle i, upon escape, bursts the j-domain at a later time than the scheduled exit time of particle j. This apparent inconsistency is due to the fact that in this serial algorithm particle j has not executed the step 1a yet. Clearly, in this occasion the position of particle j is updated by executing the step 1a rather than sampling from Eq. (3.2).

In Fig. 3.1, a graphical representation of a possible outcome of this algorithm is shown; there is not a match between the numbering in the algorithm and the numbering in the figure.

Chapter 4

Efficiency in MD-GFRD

Text and figures in this chapter have been previously published in:
L. Sbailò, and F. Noé. (2017) J. Chem. Phys. **147**, 184106.

MD-GFRD has been shown to be several order of magnitudes faster than brute-force integration of Brownian dynamics [31, 33]. The efficiency improvement is particularly evident in dilute systems, where particles spend most of their time freely diffusing in the system before encountering each other, which renders an event-based algorithm, that directly samples encountering times, dramatically faster. However, this efficiency is lost at high densities, while the efficiency of direct time-step integration is only mildly dependent on the particle density (e.g. through the number of neighbor interactions that need to be evaluated in each time step). Indeed, constructing a domain and sampling an event in it is computationally more demanding than performing few brute-force Brownian motion steps. Therefore, one typically avoids the construction of very small domains that would burst rapidly, and instead uses direct time-step integration when the size of a newly constructed domain is below the minimal domain size [31, 33, 49]. Still, as the system becomes more dense, the efficiency of this scheme decreases, as the fraction of particles that are described by direct time-step integration increases, and domains, which are required to be non-overlapping, tend to be smaller and thus more prone to a premature burst. In this context, determining the optimal size of the minimal domain and avoiding unnecessary, premature bursts can be critical to ensure computational performance.

4.1 Domain making scheme and minimal domain size

The basic idea of domain making schemes is that larger domains correlate with more efficient computation, as the particle doesn't participate in direct time-step integration during the correspondingly longer exit-times (see Eq. (3.4)). However, choosing domain sizes in a greedy manner does not necessarily lead to optimal performance. For instance, when a large domain is next to a much smaller one, or to a domain close to its escape time, the latter domain is likely to experience a particle exit very soon, which might in turn burst the large domain, thereby annihilating the advantage of the long exit-time from that domain. Domain bursting is not convenient, since it involves sampling the particle position within the domain. Moreover, it represents an unscheduled event that is difficult to treat efficiently in a parallel implementation.

The minimal domain size determines whether the domain construction is accepted or not. Instead of sampling the first exit-time from a small domain, it might be more convenient to simulate the same particle propagation via direct time-step integrations. Indeed, solving a first exit-time problem generally has a higher computational cost than simulating a number of direct time-step integrations. Thus, in MD-GFRD algorithms, the dimension of the smallest domain whose

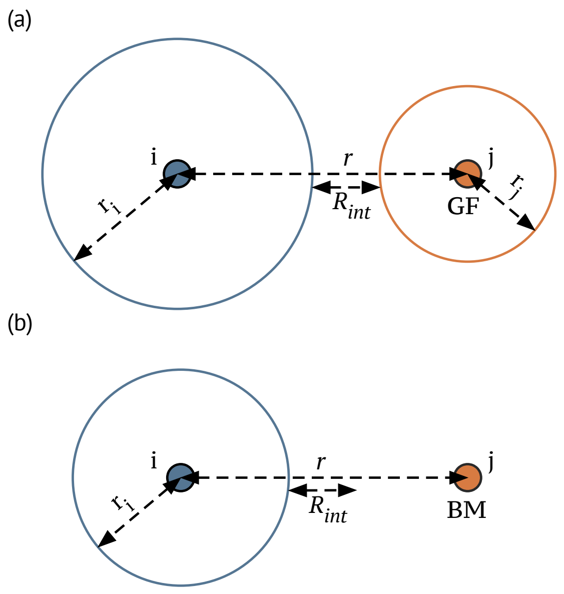


Figure 4.1: MD-GFRD domain making scheme suggested in [31, 33]. The domain size choice is made according to the status of the neighboring particle: a) the prime neighbor is a GF particle, then the shell takes all available space; b) the prime neighbor is a BM particle, then only half of the available space is used.

construction is allowed must be determined: whenever the construction of a domain of smaller size is attempted, this trial is rejected and the particle is instead brute-force integrated.

4.1.1 MD-GFRD

The MD-GFRD domain making schemes employ the largest shell principle to draw protective domains. Particles are distinguished between Green's function (GF) particles which are located in a protective domain and Brownian motion (BM) particles that are undergoing a direct time-step integration. The domain making routine firstly computes the center-center distance r_{ij} between the particle i of interest from all neighboring particles j, subtracting the interaction length R^{ij}_{int} of the particle pair. The resulting distance $\widetilde{r}_{ij} = r_{ij} - R_{int}^{ij}$ is then divided by 2 if the particle j is a BM particle. If the particle j is a GF particle, the distance is reduced by the j-domain size r_j (Fig. 4.1). In the case of a BM particle only half of the total distance is used to let the other particle construct a domain of equal size in the subsequent step. This routine is iterated over all neighboring particles and the lowest value obtained is finally selected. This domain creation makes domains as large as possible while avoiding direct particle interaction.

In previous studies, the minimal domain size in MD-GFRD algorithms has been set proportional to the particle radius [31, 33, 49], where the sum of the particles radii gives the particles pairwise interaction. In particular, the minimal domain size has been suggested to be always larger or equal than the particle radius [49]. In the implementation of Ref. [33], the minimal domain size ρ is chosen to be equal to the particle radius. In the implementation of Ref. [31], ρ can have different values depending on whether the particle is undergoing a direct time-step integration (ρ_{GFRD}) or has just escaped a protective domain (ρ_{BD}). The minimal domain value assumes a larger value when the particle is under direct time-step integration ($\rho_{GFRD} > \rho_{BD}$). This technique has been used to prevent particles from rapidly switching between the GF and BM mode. Indeed, when the particle motion is subject to direct time-step integration, it is likely to be located in a crowded region of the system, where a domain is more likely to be burst. Diminishing the number of domains constructed in this region correlates with a lowering of the total number of bursts. This scheme has been used to simulate particles interacting via a Lennard-Jones potential, and the minimal domain values $\rho_{GFRD} = 5\sigma$ and $\rho_{BD} = 3\sigma$ were used, where σ is the Van-der-Waals radius.

Finally, the bursting radius should be chosen equal to or larger than the interaction length of the two particles. However, it cannot be larger than the minimal domain size of any other particle to prevent the algorithm from entering into an infinite mutual bursting loop, where a pair of isolated particles alternatively construct a domain which is burst by the other particle in the subsequent step. In MD-GFRD, the bursting radius is set equal to the interaction length plus the minimal domain size of the particle, because whenever a particle is close to another domain, that domain must be burst in order to allow creating two new domains of significant size.

4.1.2 New domain-making scheme

The aim of the new scheme is to improve the algorithm's computational performance and to decrease the number of domain bursting events. In order to keep the number of bursting events small, domains are sized such that they have the same average first exit-time as the domains that will be constructed in their proximity. The key idea is that when domains are constructed, not only the first exit-time of the particle is sampled, but also its exit-position. This information is used by neighboring particles to propose an optimized domain size such that it has the same average first exit-time as the domains that will be later constructed on the memorized exit-positions (Fig. 4.2 1). In Ref. [50] the importance of constructing optimized domains has already been discussed, and it is suggested that domains should be constructed to delay in time as far as possible the first event in the queue, which corresponds to constructing domains with equal mean first exit-times. However, this was achieved only when all domains are constructed simultaneously, which optimizes only over the first event in queue. By pre-sampling the exit-position of particles, it is instead possible to construct balanced domains over a long series of events.

In order to further reduce the number of bursting events, the domain size is then shrunk. Although the domains are not chosen to be of maximum size, this approach significantly reduces the overall number

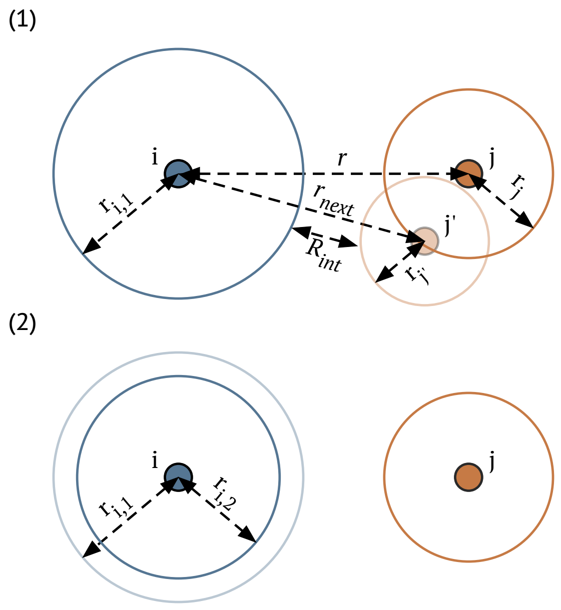


Figure 4.2: New domain making scheme for the case of an isolated pair of particles. At time t_i, particle i is attempting the construction of the i-domain close to particle j that is already enclosed in a domain. The escape time $t_{j'} > t_i$ and particle j's escape position were sampled when the domain was constructed. 1) Particle i constructs a domain whose size $r_{i,1}$ is such that its average first exit-time is the same as the average first exit-time of particle j from the j'-domain that might be constructed after the exit from its current j–domain. 2) The domain size in the previous step obtained is further reduced to finally obtain $r_{i,2}$.

of bursts compared to the scheme described in Sec. 4.1.1. The choice for the size reduction in the second step (Fig. 4.2 2) is performed to obtain a balance between a low number of bursts and long domain exit-times. Clearly, the specific setting of these parameters depends on implementation details such as serial or parallel execution, and can be adapted to the local setting. This algorithm is illustrated in the simplest case of an isolated pair of particles in Fig. 4.2. In the new scheme, the bursting radius is also chosen to be equal to the interaction length plus the minimal domain size.

In practice, if the domain is created close to a GF particle (Fig. 4.2 1) the first domain $r_{i,1}$ is obtained by solving a system of two equations:

$$\frac{r_{j'}^2}{6D_j} + \Delta t = \frac{r_{i,1}^2}{6D_i}, \tag{4.1}$$

$$\tilde{r}_{\text{next}} = r_{i,1} + r_{j'}, \tag{4.2}$$

where $\tilde{r}_{\text{next}} = r_{\text{next}} - R_{\text{int}}$ is the available space, r_{next} is the distance between the center of particle i and the exit-position of particle j, $\Delta t = t_{j'} - t_i$ is the time difference between the scheduled exit-time of particle j and the current time, *i.e.* the time in which particle i is attempting to construct a domain. The first equation imposes that the average exit-time from the i-domain is the same as from the j'-domain, where the expected exit-time $\langle\tau\rangle$ of a Brownian particle with diffusion coefficient D from a sphere of radius b is:

$$\langle\tau\rangle = \frac{b^2}{6D}. \tag{4.3}$$

The second equation enforces the domains to be adjacent by taking all available space, according to the largest shell principle. In contrast to MD-GFRD, the largest domain principle is applied between the i-domain and the j'-domain that is possibly constructed subsequently.

The solution to Eq. (4.1) has the following two roots:

$$r_{i,1} = \widetilde{r}_{next}\,\frac{1 \pm \sqrt{1 - (1 - \frac{D_j}{D_i})(1 + \frac{6\,\Delta t\, D_j}{\widetilde{r}_{next}^2})}}{1 - \frac{D_j}{D_i}}. \tag{4.4}$$

Assuming that the condition $\Delta t < \tilde{r}_{\text{next}}^2/6D_i$ is satisfied, the argument of the square root is nonnegative, resulting in two real-valued solutions. In the following derivations, two different cases are studied depending on D_i and D_j.

Firstly, $D_i > D_j$ is considered, which leads to $1-\frac{D_j}{D_i} > 0$. In case the discriminant is added, the factor that multiplies $\tilde{r}_{next}$ is clearly higher than one, since diffusion coefficients are always positive, then we would obtain $r_{i,1} > \tilde{r}_{next}$, an unphysical solution. The discriminant must thus be subtracted. Furthermore, imposing the condition $\frac{\tilde{r}^2_{next}}{6D_i} > \Delta\tau$, or equivalently $\frac{\Delta\tau}{\tilde{r}^2_{next}} < \frac{1}{6D_i}$, it can be verified that if the discriminant is subtracted:

$$r_{i,1} < \tilde{r}_{next}\,\frac{1-\sqrt{1-(1-\frac{D_j}{D_i})(1+\frac{D_j}{D_i})}}{1-\frac{D_j}{D_i}} = \tilde{r}_{next}. \tag{4.5}$$

The condition $r_{i,1} < \tilde{r}_{next}$ is satisfied if the discriminant is subtracted.

In case $D_j > D_i$, then $1-\frac{D_j}{D_i} < 0$:

$$r_{i,1} = \tilde{r}_{next}\,\frac{1 \mp \sqrt{1+ \mid 1-\frac{D_j}{D_i} \mid \,(1+\frac{6\,\Delta t\,D_j}{\tilde{r}^2_{next}})}}{\mid 1-\frac{D_j}{D_i} \mid}. \tag{4.6}$$

In order to satisfy the condition $r_{i,1} > 0$, the discriminant must have a positive sign. However, the sign of the discriminant has been inverted by the modulus in the denominator, since it comes from the subtraction of the discriminant.

To sum up, only the root obtained by subtracting the discriminant satisfies the condition $0 < r_{i,1} < \tilde{r}_{next}$:

$$r_{i,1} = \tilde{r}_{next}\,\frac{1-\sqrt{1-(1-\frac{D_j}{D_i})(1+\frac{6\,\Delta t\,D_j}{\tilde{r}^2_{next}})}}{1-\frac{D_j}{D_i}}. \tag{4.7}$$

If the average first exit-time of particle i from the available space $r_{i,1} = \tilde{r}_{\text{next}}$ is less than Δt, the time interval to the scheduled exit-time of particle j, the solution of the system in Eq. (4.1) has no real values, which means that the i-domain and the j'-domain cannot have the same average first exit-time. As the j-particle is not expected to burst the i-domain in this case, we use all available space for the i-domain, i.e. $r_{i,1} = \widetilde{r}_{next}$. Consistently, inserting $\Delta t = \widetilde{r}^2_{next}/6D_i$ in Eq. (4.1) results in the solution $r_{i,1} = \widetilde{r}_{next}$.

The system in Eq. (4.1) is then solved only when $\Delta t < \tilde{r}^2_{\text{next}}/6D_i$. The optimal domain size is then given by:

$$r_{i,1} = \begin{cases} \widetilde{r}_{\text{next}}, & \Delta t \geq \frac{\widetilde{r}^2_{\text{next}}}{6D_i}. \\ \widetilde{r}_{\text{next}} \, \frac{1-\sqrt{1-(1-\frac{D_j}{D_i})(1+\frac{6\,\Delta t\, D_j}{\widetilde{r}^2_{next}})}}{1-\frac{D_j}{D_i}}, & \text{otherwise.} \end{cases} \tag{4.8}$$

The square root argument in Eq. (4.8) is always positive if $\Delta t < \widetilde{r}^2_{next}/6D_i$, therefore the solution is always real-valued.

If the two particles have identical diffusion coefficients D, the solution simplifies to:

$$r_{i,1} = \frac{\widetilde{r}_{\text{next}}}{2} + \frac{3\,D\,\Delta t}{\widetilde{r}_{\text{next}}}. \tag{4.9}$$

The value $r_{i,1}$ obtained is a function of the distance $\widetilde{r}_{\text{next}}$. Hence, $r_{i,1}$ does not take the volume of the existing j-domain into account and thus does not ensure to avoid overlap of the i and j domains. To avoid such an overlap, the i-domain must be accordingly resized to the largest possible value: $r_{i,1} = \tilde{r} - r_j$, where $\tilde{r} = r - R_{\text{int}}$ and r is the center-center distance between particles i and j.

A similar approach is used if particle j is a BM particle. In this case, the i-domain is created so as to leave enough space for particle j to construct a domain whose first exit-time is equal to the i-domain:

$$r_{i,1} = \frac{\widetilde{r}}{1 + \sqrt{\frac{D_j}{D_i}}}. \tag{4.10}$$

Finally, the domain radius is further reduced as :

$$r_{i,2} = r_{i,1} - n_{\text{red}}\sqrt{2D_j\, dt},. \tag{4.11}$$

where n_{red} is a parameter (Fig. 4.2 2). The domain reduction is set proportional to the average displacement that the particle j performs in one integration step. This reduction is performed to reduce the probability that the particle j bursts the i-domain in cases where the sampled escape time of the particle i is larger than the expected value. Note that if $\Delta t > r_{i,1}^2/6D_i$ the particle j is expected to escape its domain after the particle i, in this case there is no need to reduce the size of the i-domain and thus the step in Eq. (4.11) is omitted. When this scheme is applied to multi-particle systems, the previously outlined approach is applied to all nearest-neighbor particle pairs, and the lowest value of $r_{i,2}$ is chosen.

4.1.3 New scheme for minimal domain size

In contrast to previous works, the minimal domain size is proposed here to be proportional to the square root of the particle diffusivity, rather than the particle size. The minimal domain size defines the particle distance below which direct time-step integration is assumed to be more efficient than sampling first exit-times. It is assumed that the CPU time required to sample the probability density of the first exit-time is approximately independent of domain size and diffusion coefficient. In contrast, the CPU time spent to simulate first exit-times via brute-force integrations depends on the domain size, on the particle diffusion coefficient and on the time-step length.

Given the average first exit-time $\langle\tau\rangle$ of a particle with diffusion coefficient D from a sphere of radius b, Eq. (4.3), the average number of steps $\langle n\rangle$ to simulate the first exit-time is:

$$\langle n\rangle = \frac{b^2}{6\,D\,dt}, \tag{4.12}$$

where dt is the time step. The average CPU time, $\langle T_{BF}(b)\rangle$, spent to compute escape times via brute-force integrations is proportional to

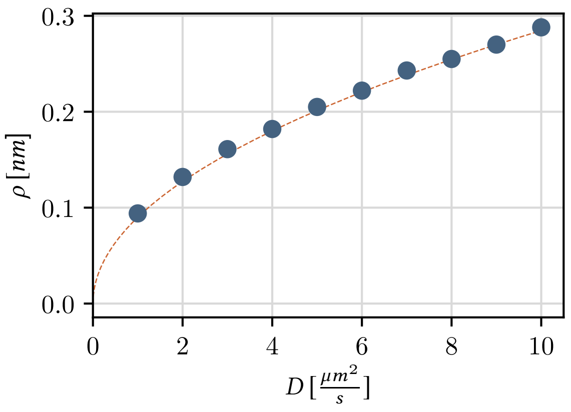


Figure 4.3: Minimal domain radius ρ as a function of D using the time step $\mathrm{d}t = 0.1\,ns$. The dots represent the radius of the minimal protective domain where first exit-time sampling and direct time-step integration have equal CPU costs. Simulations to compute the first exit-time from the domain with size ρ were conducted for different diffusion coefficients and domain sizes, using either direct time-step integration or first exit-time sampling. For small domain sizes, the direct time-step integration is always more efficient. The dashed red line shows $\rho = \alpha\sqrt{D\,dt}$, as described in Eq. (4.16), with the implementation-specific value $\alpha = 8.4$ that has been found empirically.

the number of integration steps, and thus:

$$\langle T_{BF}(b)\rangle \propto \frac{b^2}{D\,dt}. \tag{4.13}$$

It is assumed that the average CPU time, $\langle T_{GF}(b)\rangle$, spent to sample a first exit-time is approximately constant.

$$\langle T_{GF}(b)\rangle = Const. \tag{4.14}$$

Let ρ be the domain size at which the CPU times are equal, $\langle T_{BF}(\rho)\rangle = \langle T_{GF}(\rho)\rangle$, then:

$$\rho^2 \propto D\,dt. \tag{4.15}$$

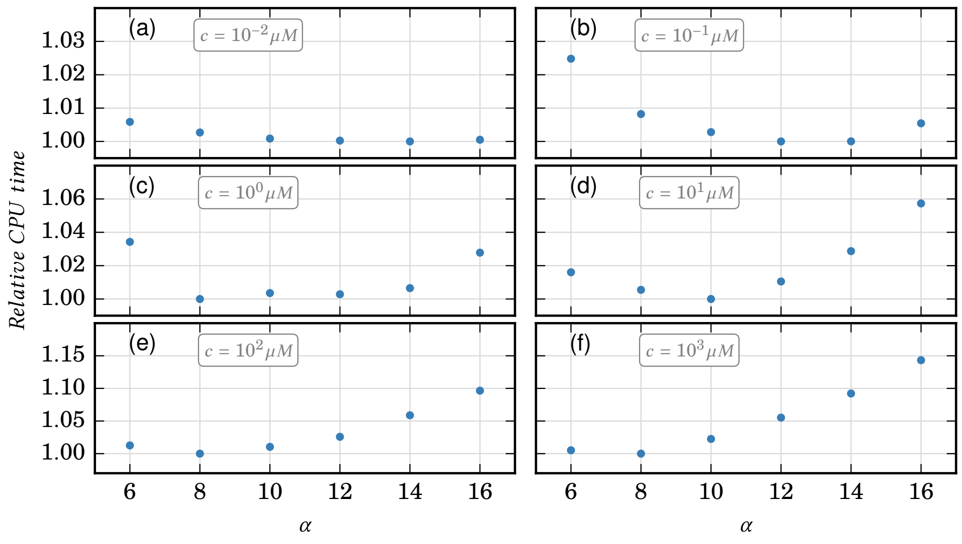


Figure 4.4: Relative CPU times required to perform the same simulation as described in Fig. 4.5 a for different α values. In each plot, the CPU times are relative to the minimum. The value $\alpha = 6$ permits the construction of very small domains, even when direct time-step integration would be preferable. The optimal value $\alpha = 8.4$ found in Fig. 4.3 would represent the optimal value in case the constructed domains do not burst. As α is increased from its optimal value $\alpha \approx 9$ the algorithm's performance decreases.

Hence, the minimal domain radius $\rho(D, dt)$ is defined as the threshold that determines whether the domain construction is accepted or not.

$$\rho(D, dt) = \alpha\sqrt{D\,dt}. \tag{4.16}$$

Simulations indicate that this function correctly describes the point where a direct time-step integration becomes more efficient than a first exit-time sampling (Fig. 4.3).

4.1.4 α-values in minimal domain size

The minimal domain size is given by Eq. (4.16), where α is a parameter that is determined in the beginning of the simulation. An optimal value $\alpha = 8.4$ has already been suggested in Fig. 4.3. However, that value was selected by taking only the first exit-time sampler and the direct time-step integrator into account. In general, it might seem appropriate to insert a penalty for the possibility of a burst and then to slightly rise the α value, where the penalty would be higher when a higher number of bursts is expected.

Fig. 4.4 shows that the optimal value of α lies in the range $8 < \alpha < 12$, in agreement with Fig. 4.3, and $\alpha = 9$ is chosen in the following benchmark simulations.

4.2 Results

Here we compare the performance of the multi-scale MD-GFRD scheme implemented in Refs. [31] and [33], the new scheme, and a direct time-step integration scheme using Brownian dynamics. Two versions of the new scheme are simulated, one with $n_{red} = 5$ in Eq. (4.11) (new scheme 1), and one which does not use domain size reduction ($n_{red} = 0$, new scheme 2), thus tending to size domains more greedily. In addition, a hybrid scheme is tested, which implements the minimal domain size as described in Sec. 4.1.3 but employs the same domain making scheme as proposed in Refs. [31] and [33]. For simplicity particles in a periodic box and interacting with a harmonic repulsion are simulated:

$$V(r) = \frac{1}{2} k \left(R_{int} - r\right)^2, \qquad r < R_{int}, \tag{4.17}$$

where r is the inter-particle distance between the centers of mass, $k = 100$ is the spring constant, and the interaction length R_{int} is equal to the sum of particle radii. Reactions, more complex particle-particle

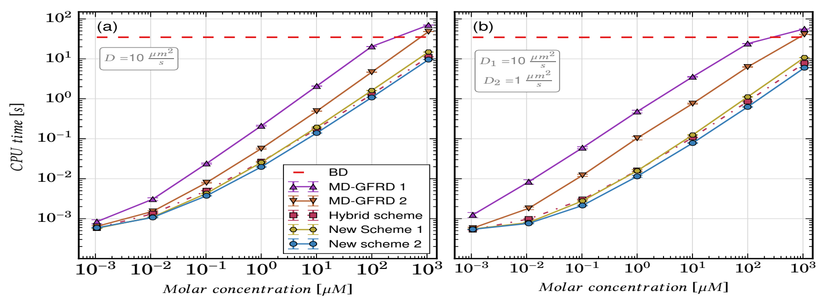


Figure 4.5: CPU time required to simulate $1\,ms$ of real time, using a brute-force integration step of $dt = 0.1\,ns$. The number of particles is kept fixed to $N = 10$, while the system volume is adapted to the selected molar concentration. Simulations are performed in a cubic-shaped box with periodic boundary conditions. Particles are spherical-shaped with radius $R = 2.5\,nm$ and diffusion coefficient $D = 10\,\mu m^2/s$ in (a) and radii $R_1 = 1.5\,nm$ and $R_2 = 3.5\,nm$ and diffusion coefficients $D_1 = 10\,\mu m^2/s$ and $D_2 = 1\,\mu m^2/s$ in (b). A binary interaction length is defined as the sum of particles' radii, when particles are within this distance repulse according to a harmonic potential as in Eq. (4.17), where $k = 100$. The minimal domain of the new schemes and of the hybrid scheme uses the pre-factor $\alpha = 9$ as defined in Eq. (4.16). In new scheme 1, $n_{red} = 5$; in new scheme 2, $n_{red} = 0$, see Eq. (4.11). In MD-GFRD 1, the minimal domain size is equal to the particle radius[33]. In MF-GFRD 2, the minimal domain sizes $\rho_{GFRD} = 2.5\,R$ and $\rho_{BD} = 1.5\,R$[31] were used, where the pre-factors 1.5 and 2.5 have been chosen to adapt to a different simulation than the pre-factors used in Ref. [31], while preserving their same relative proportions. At low concentrations, MD-GFRD schemes are several order of magnitude faster than BD. The new schemes and the hybrid scheme are faster than BD up to concentrations of $10^3\,\mu M$, while MD-GFRD schemes are preferable over BD up to $10^2\,\mu M$.

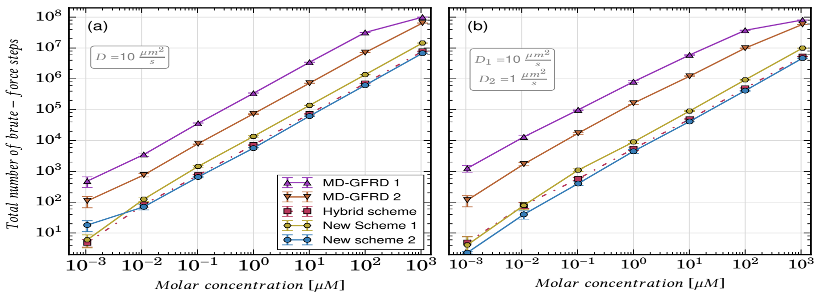


Figure 4.6: Total number of direct time-steps in the multi-scale MD-GFRD simulations described in Fig. 4.5. As the particle density increases, interactions between particles become more frequent, and more simulation time is spent in conducting direct time-step integration. The behavior of these curves is similar to that in Fig. 4.5, indicating that the number of brute-force Brownian motion steps represent the bottle-neck in the present simulations. The largest value in each plot, 10^8, represents the condition where each of the 10 particles have performed 10^7 direct time-steps, which means that no particle propagation was made using first exit-time sampling.

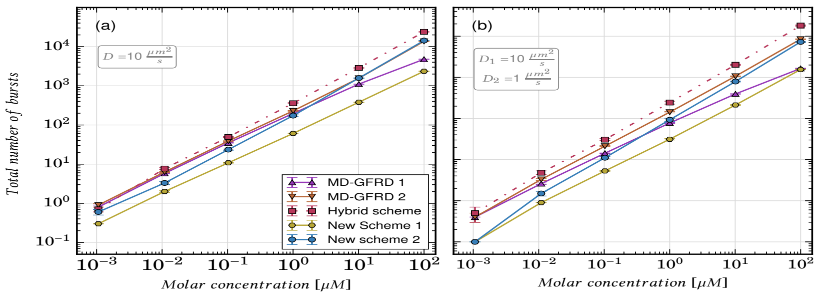


Figure 4.7: Average total number of protective domain bursts in the multi-scale MD-GFRD simulations described in Fig. 4.5. As the molar concentration is increased, domains tend to be smaller and to be constructed more often, which goes along with an increase of the number of bursts. The average number of bursts in the new scheme 1 is lower than in the previous MD-GFRD implementations at any density. Keeping the total number of bursts low can be important for efficient parallelization, e.g. using Graphical Processing Units (GPUs).

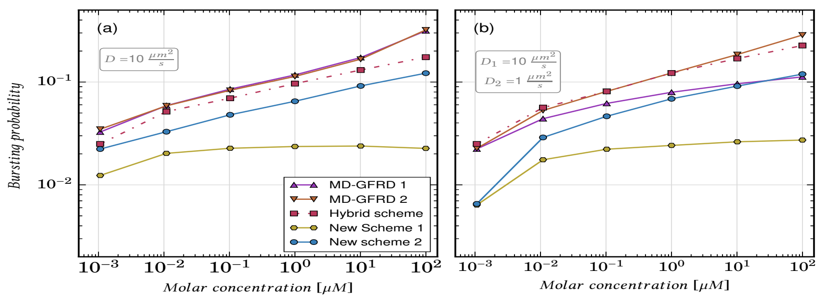


Figure 4.8: Domain bursting probability, i.e. ratio of the total number of domains burst over the total number of domains constructed in the MD-GFRD simulations described in Fig. 4.5. The domain construction schemes proposed here are clearly more efficient than previous schemes and results in domains that are more likely to survive until the particles contained therein make successful exits. The bursting probability is always lower than 3% in the new scheme 1, while in the implementations MD-GFRD 1,2 this value is roughly an order of magnitude larger at the higher concentrations.

potentials, or other near-space interactions can be straightforwardly integrated in the direct time-step integration regime that is used to simulate interacting particles.

Two simulations have been performed using different diffusion coefficients and particle radii:

1. 10 spherical particles with radius $R = 2.5\,nm$ and diffusion coefficient $D = 10\,\mu m^2/s$.

2. 5 faster and smaller particles with radius $R_1 = 1.5\,nm$ and diffusion coefficient $D_1 = 10\,\mu m^2/\mathrm{s}$ and 5 slower and larger particles with radius $R_2 = 3.5\,nm$ and diffusion coefficient $D_2 = 1\,\mu m^2/\mathrm{s}$.

4.2.1 Efficiency comparisons of different MD-GFRD schemes and direct Brownian dynamics

To obtain clean benchmarks, most calculations are run with ten particles and direct evaluation of all pairwise particle distances, while the particle density is adjusted by choosing the box size. For a more complex test, Sec. 4.2.3 simulates larger particle numbers with a neighbor list implementation.

The efficiency of MD-GFRD strongly depends on the particle concentration, since in case of dilute systems particles are allowed for constructing large domains and performing large time steps. Hence, MD-GFRD algorithms are dramatically faster than BD schemes at low concentrations. As the particle concentration is increased, MD-GFRD becomes less efficient, while the BD efficiency remains constant. Consequently, there is a concentration threshold where BD starts being more efficient than MD-GFRD. In Fig. 4.5, the performance is compared between the new schemes, the hybrid scheme, the previous MD-GFRD schemes and direct BD simulation. It is evident that all MD-GFRD schemes are several order of magnitude faster than BD at low densities. Moreover, the new schemes are faster than the previous MD-GFRD schemes at all densities, but performances are similar at low densities. In particular, for both diffusion coefficients, the new schemes and the hybrid scheme are preferable over BD for concentrations up to 10^3 μM, whereas previous MD-GFRD schemes were preferable over BD only up to molar concentrations of 10^2 μM. The schemes which implement the new minimal domain size all show similar performance, and among them the new scheme 2 is the fastest. These numbers may be different in different implementations (codes and machines), and comparison is therefore only meaningful within the same implementation.

The total number of direct integration time-steps performed in each multi-scale MD-GFRD simulation increases with increasing particle concentration (Fig. 4.6). This growth is remarkably similar to the growth in CPU time, indicating that the reason of the improved performance of MD-GFRD schemes is essentially due to a reduction of the direct time-integration steps that represent the computational bottleneck. In the new schemes and in the hybrid scheme, the minimal domain size is smaller than in previous MD-GFRD schemes, which enables more protective domains to be constructed, which in turn reduces the fraction of time spent in direct time-step integrations. Although having equal minimal domain size, the new scheme 2 shows a slightly lower number of direct integration time-steps with respect to the hybrid scheme. This is essentially the result of the construction of more balanced domains which allow for an optimization of the available space. On the other hand, the new scheme 1 spends a larger fraction of time under direct time-step integration, because after the reduction step more domains are not sufficiently large for construction.

4.2.2 Minimization of the domain burst frequency

Despite the fact that domain sizes are small on average, Fig. 4.7 shows that the total number of bursts is the lowest in new scheme 1, i.e. when the domain reduction is included. The hybrid scheme involved the highest number of bursts, since the construction of small domains is allowed, but their sizes are not chosen optimally. The incorporation of particle exit-positions into domain construction, and the choice of domain sizes so as to balance the exit-times allows to reduce the number of bursts to one third (new scheme 2); if a reduction step is also added (new scheme 1), the number of bursts is further reduced by approximately one order of magnitude. This improved efficiency on the domain construction is evident in Fig. 4.8, which shows the probability that a protective domain is burst prematurely by intrusion of another particle rather than being annihilated by a regular exit of the particle contained therein. This quantity is computed as the ratio of the total number of domain bursts over the total number of constructed domains. At low concentrations the bursting probability is small, but it increases with increasing particle density. The new domain-making scheme clearly results in more efficient domains that are much less probably to be burst prematurely compared to the previous MD-GFRD scheme, especially at higher concentrations.

The full implementation of the new scheme (version 1) is to be preferred to previous MD-GFRD schemes in both cases: when the serial computational performance is most relevant and when the number of total bursts is required to be low. The MD-GFRD implemented in Ref. [33] is faster than the implementation in Ref. [31], while the latter scheme has a lower number of domain bursts. The new scheme 1 is instead superior in both computational performance and number of domain bursts. More specifically, the implementation as in new scheme 1 is optimal to drastically lower the number of bursts while preserving efficiency. The new scheme 2 instead has a slightly higher CPU performance in our implementation, but does not keep the number of bursts small. The improvements result in up to an order of magnitude of gain in the CPU performance and an order of magnitude of gain in the total number of bursts.

4.2.3 Large particle numbers

The general trends observed in the benchmarks shown in the previous sections are also expected to hold for systems with many particles. However, in systems with many particles n, it is necessary to implement a neighbor list to avoid that each time step scales with n^2 as a result of the pairwise distance calculations.

In order to validate that the new MD-GFRD scheme can still be efficiently implemented with many particles, new scheme 1 is imple-

Table 4.1: Computational time to simulate $1\,ms$ of real time, using a brute-force integration step of $dt = 0.1\,ns$. In these simulations the new scheme is more efficient than BD up to a molar concentration of $10^3 \mu M$.

Molar concentration	**Particles number**	**CPU time new scheme**	**CPU time BD**
$10^2\,\mu M$	10^3	$271\,s$	$14.5 \cdot 10^3\,s$
$10^3\,\mu M$	10^4	$230 \cdot 10^3 s$	$260 \cdot 10^3 s$

mented with $n_{red} = 5$ using a neighbor list. Particles are interacting with harmonic repulsion with radius $R = 2.5\,nm$, $k = 100$, and periodic boundary conditions are applied as described in the previous section. The system volume is kept fixed to $17.576\,10^6\,nm^3$, while the number of particles is adapted to achieve the desired molar concentration. All particles have diffusion coefficient $D = 10\,\mu m^2/s$.

In order to efficiently implement a neighbor list, a discretization of the simulation box in cells of length $L_{cell} = 5\,nm$ is used for the brute-force BD simulations, and of $L_{cell} = 10\,nm$ is used for the MD-GFRD simulations. Each particle checks the cell it is located in and the 26 neighboring cells for possible neighbors. In such a cell discretization, the smallest distance at which two particles can loose track of each other is the cell length, and thus the maximum protective domain size must be limited to at most half the cell length minus the interaction length, which is the gap to be left between contiguous domains. Here, the maximum domain size is limited to $R_{max} = 2.5\,nm$.

The simulation results in Tab. 4.1 show that the new scheme remains to be faster than a brute-force integration up to a molar concentration of $10^3 \mu M$.

Chapter 5

First passage schemes

Text and figures in this chapter have been previously published in:
L. Sbailò, and L. Delle Site. (2019) J. Chem. Phys. **150**, 134106.

MD-GFRD algorithm can be inserted into a broader class of first passage algorithms that deploy first exit-time sampling in event-based schemes. Applications of first passage algorithms, across different subjects and disciplines, span from the determination of the ground state of bosonic particles [51] to the clustering of the oncoprotein Ras [52] (further representative examples can be found in Refs.[13, 53, 50, 54, 55, 56, 57, 8, 58, 59, 33]), and their first use can be dated back to the 1970s.

The use of first passage propagators to simulate particle diffusion was originally conceived by Kalos, Levesque and Verlet (KLV) in Ref. [51] as a synchronous scheme: (i) protective domains are constructed around each particle and the corresponding exit-time is sampled; (ii) the shortest exit-time is determined and particles are propagated to this time; (iii) finally, new protective domains are constructed for all particles. The main disadvantage of the KLV scheme is that local particle updates affect the total system. This algorithm is then inefficient in simulating dishomogeneous multi-scale systems. Instead, different time scales can be maintained through an asynchronous scheme [60], as in the first passage kinetic Monte Carlo (FPKMC) [13]. In asynchronous event-based schemes, the position of the particle involved in the next exit-time event is updated locally, while next events of all other particles are maintained in a time ordered event-list, which is updated every time a new event is sampled. Finally, MD-GFRD pairs the event-driven first exit-time sampling to a time-driven integration of the particles motion, and is the most recent development of first passage schemes.

In first passage schemes, the event-based particle propagation is performed either by placing the particle randomly on the domain borders when the pre-sampled exit-time is reached, or by prematurely updating the particle position inside the domain before exit-time. These values are sampled from probability distributions that are derived and studied in this chapter.

5.1 Derivation of the first exit-time probability distribution

The motion of a Brownian particle is described stochastically by Einstein's diffusion equation

$$\frac{\partial}{\partial t} f(\vec{r}, t) = D \Delta f(\vec{r}, t), \tag{5.1}$$

where $f(\vec{r}, t)$ is the probability of finding the particle in position $\vec{r}$ at time t, and D is the diffusion coefficient. We assume that the particle position is known at time t_0, and we are interested in the first exit-time of the particle from a domain Ω that is constructed around the particle. The domain Ω is supposed to be spherical and centered on the particle position. Given the spherical symmetry of the system, the angular coordinates can be averaged out from Eq. (5.1)

$$f(\vec{r},t) = const\, p(r,t). \tag{5.2}$$

The first exit-time from a protective domain Ω is derived by imposing absorbing boundary conditions on the domain boundary $\partial\Omega$[48]. The domain Ω is here assumed to be a sphere of radius b

$$p_\Omega(r=b,t) = 0; \quad \forall t. \tag{5.3}$$

This condition ensures that once the particle hits the domain borders, it is removed from the system. Firstly, the Green's function for the diffusion equation is found, then exit-time boundary conditions are applied to the solution. The diffusion equation is only solved for the radial component

$$\frac{\partial p(r,t)}{\partial t} = D\frac{1}{r^2}\frac{\partial}{\partial r}\left(r^2\frac{\partial p(r,t)}{\partial r}\right). \tag{5.4}$$

The solution is found by assuming that it can be separated in the variables t and r, $p(r,t) = g(r)h(t)$,

$$\frac{1}{D\,h(t)}\frac{\partial h(t)}{\partial t} = \frac{1}{g(r)r^2}\frac{\partial}{\partial r}\left(r^2\frac{\partial g(r,t)}{\partial t}\right). \tag{5.5}$$

Each term of the above equation has only one variable, in this case both terms can be equated to a constant value resulting in two different equations

$$\frac{1}{D\,h(t)}\frac{\partial h(t)}{\partial t} = \frac{1}{g(r)r^2}\frac{\partial}{\partial r}\left(r^2\frac{\partial g(r,t)}{\partial t}\right) = -\lambda^2. \tag{5.6}$$

The solution of the first equation is an exponential function

$$h(t) = Ae^{-\lambda^2 Dt}. \tag{5.7}$$

The second equation can be expressed as a harmonic oscillator differential equation by substituting $\tilde{g}(r) = r\, g(r)$. The solution is then straightforward

$$g(r) = \frac{1}{r}\left(B\sin(\lambda r) + C\cos(\lambda r)\right). \tag{5.8}$$

The capital letters A, B, C used are normalization factors. Absorbing boundary conditions $p_\Omega(b,t) = 0$ are imposed on the equation $p(r,t) = g(r)h(t)$ obtained from Eqs. (5.7) and (5.8)

$$p_\Omega(r,t) = \frac{1}{r}\sum_m A_m e^{-\frac{m^2\pi^2}{b^2}Dt}\sin\left(\frac{m\pi}{b}r\right). \tag{5.9}$$

The particle is known to be in r_0 at t_0

$$p_\Omega(r,t|r_0,t_0) = \frac{1}{2\pi b}\frac{1}{r\, r_0}\sum_m e^{-\frac{m^2\pi^2}{b^2}D(t-t_0)}\sin\left(\frac{m\pi}{b}r\right)\sin\left(\frac{m\pi}{b}r_0\right). \tag{5.10}$$

The survival probability is obtained by integrating the above equation

$$\begin{aligned} S_\Omega(t|r_0,t_0) &= \int_{\vec{r}\in\Omega} p_\Omega(\vec{r},t|r_0,t_0)\, d\vec{r} \\ &= -\frac{2b}{\pi r_0}\sum_m \frac{(-1)^m}{m} e^{-\frac{m^2\pi^2}{b^2}D(t-t_0)}\sin\left(\frac{m\pi}{b}r_0\right). \end{aligned} \tag{5.11}$$

The first exit-time is then derived from the survival probability

$$\begin{aligned} q_\Omega(\tau|r_0,t_0) &= \frac{dS_\Omega(\tau|r_0,t_0)}{d\tau} = \\ &= -\frac{2\pi D}{b r_0}\sum_m m(-1)^m e^{-\frac{m^2\pi^2}{b^2}D(t-t_0)}\sin\left(\frac{m\pi}{b}r_0\right). \end{aligned} \tag{5.12}$$

First-exit times from protective domains are sampled from $q_\Omega(\tau|0,0)$ upon domain construction, then this time is inserted in the event list.

The event-based particle propagation is performed either by placing the particle randomly on the domain borders when the sampled exit-time is reached, or by prematurely updating the particle position inside the domain when this is burst before the exit-time. The position update within domain is an unscheduled event that occurs when the free potential assumption is broken, because this invalidates the exit-time that has been sampled upon domain construction. The particle position is updated before the scheduled domain escape, and this event might affect the probability distribution of the particle position before the sampled exit-time. Hence, a discussion about the correlations between the sampled exit-time and particle position before escape is required.

5.2 Sampling particle position before domain escape

In first passage schemes, the probability distribution used to update the particle position is the Green's function in Eq. (5.10) renormalized by the survival probability

$$g_\Omega(\vec{r}, t|0,0) = \frac{p_\Omega(\vec{r}, t|0,0)}{S_\Omega(t|0,0)}. \tag{5.13}$$

However, the position sampling is carried out after the extraction of the first exit-time from the domain, but in Eq. (5.13) the information of the exact moment when the particle hits the boundary is missing as a condition for the above probability distribution, despite this being a requirement [61]. According to the Bayes theorem, the probability distribution is conditional with respect to the sampled exit event

$$g_\Omega(\vec{r}, t|0,0;\tau) = \frac{q_\Omega(\tau|\vec{r}, t) p_\Omega(\vec{r}, t|0,0)}{q_\Omega(\tau|0,0)}. \tag{5.14}$$

Indeed, when an observable is sampled at a time between the initial time and the time of a determined event, the probability distribution should be conditional until realization of the determined event. For instance, if the particle position is sampled immediately before the exit-time, the particle is expected to be in proximity to the domain borders. Let us assume that the sampling-time t is equal to the exit-time τ minus an infinitesimal time $t = \tau - dt$, if the particle position $\vec{r}$ is sampled at a finite distance $\Delta\vec{R}$ from the domain border $\vec{\Omega}$ then $\vec{r} = \vec{\Omega} - \Delta\vec{R}$, which leads to the evidence that the particle travels a finite distance $\Delta\vec{R}$ in an infinitesimal time dt. This is a contradiction, which might occur when the condition on the exit-time is missing. Despite the conceptual contradiction underlined above, numerical applications of first passage schemes have always been shown to correctly reproduce the expected results. The discussion above naturally leads to the need of a formal proof of correctness of the algorithm which can clarify the issue about the reported conceptual contradiction.

The distinction between the conditional and unconditional probability becomes relevant when a systematic bursting of the domains is performed, and this might be the case of MD-GFRD. In MD-GFRD, the particles clock must be synchronized during their interaction, and this synchronization can be obtained by projecting the exit-time onto a discrete temporal grid. The projection can be performed after the exit-event with a fractional Brownian motion propagation or before the exit-event by sampling the particle position in the last discrete time before the exit-time. For the latter case, the choice of the probability distribution is crucial, for the reason that a systematic use of an unconditional probability distribution under these circumstances would bring the simulation to inaccurate results.

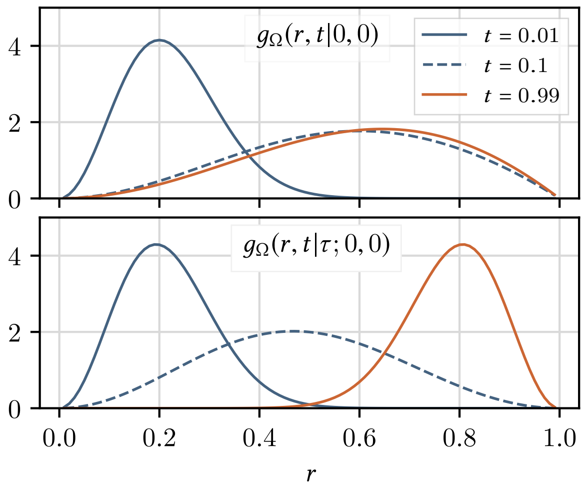


Figure 5.1: Radial probability distributions as in Eq. (5.14) and Eq. (5.13). The domain Ω is a sphere centered in the origin, and, given the spherical symmetry of the problem, the probability distributions depend only on the radial distance r from the origin. All variables are adimensional. The first exit-time has been given a unitary value $\tau = 1$; plots have been performed for different values of the bursting-time t. Immediately after the initial time the conditional and unconditional probability distributions result identical. The particle is located in proximity of the origin, and the information of the exit-time is essentially irrelevant at this time. As time increases, the distributions diverge, becoming clearly different when the bursting-time is close to the extracted exit-time.

5.3 Exit-time conditions on the position distribution probability

Eq. (5.14) and Eq. (5.13) are clearly different (see Fig. 5.1) and the assumption that these two distributions can be used interchangeably might be surprising. In particular, in asynchronous schemes as FPKMC and MD-GFRD, the correctness of the use of the unconditional probability is not evident, because one may ask how the position probability of the particle inside the domain is characterized when the position of other particles is updated and known. This can be justified by considering the probability distribution conditional with respect to the exit-time. Indeed, when the exit-time is extracted, it is ensured that the particle is inside the domain and that the particle position is probabilistically described by Eq. (5.14) until realization of the exit-event.

It is possible to sample from Eq. (5.13), only when the sampling time is independent of the exit-time extraction. The problem can also be reformulated by introducing $f(\vec{r}, t|0, 0)$, which represents the unconditional probability that the particle is in $\overrightarrow{r}$ at t. The particle is supposed to be initially placed in a protective domain Ω. According to the Bayes theorem, the probability distribution $f(\vec{r}, t|0, 0)$ can be decomposed into two conditional probabilities with respect to the event that the particle never crossed the domain borders $\partial\Omega$ until t

$$\begin{aligned} f(\vec{r}, t|0, 0) &= g(\vec{r}, t|0, 0; \vec{r} \in \Omega\, \forall t' \leq t) h(\vec{r} \in \Omega\, \forall t' \leq t) \\ &+ (1 - h(\vec{r} \in \Omega\, \forall t' \leq t)) \mathcal{P}(\vec{r}, t|\partial\Omega), \end{aligned} \tag{5.15}$$

where $h(\vec{r} \in \Omega\, \forall t' \leq t)$ is the probability that the particle has never escaped the domain, and $\mathcal{P}(\vec{r}, t|\partial\Omega)$ is the probability that the particle is in $\vec{r}$ after crossing the domain borders $\partial\Omega$. The particle position at t can be sampled from the above equation as follows:

1. Sample the first exit-time τ.
2. *if* $t < \tau$: *then* the particle position is sampled from $g(\vec{r}, t|0, 0; \vec{r} \in \Omega\, \forall t' \leq t)$.
3. *else if* $t > \tau$: *then* the particle position is the result of a Wiener process of time length $t - \tau$, starting in a random position on the domain borders.

For a given observation time t the first exit-time τ is extracted, the comparison between these times, i.e. the *if* statements in the list, in effect samples $h(\vec{r} \in \Omega\, \forall t' \leq t)$. The only relevant assumption to sample $h(\vec{r} \in \Omega\, \forall t' \leq t)$ is that t and τ are uncorrelated.

The probability distribution $g(\vec{r}, t|0, 0; \vec{r} \in \Omega\, \forall t' \leq t)$ requires that the particle is known to be inside the domain at time t and that it has never left it before, conditions satisfied by both Eq. (5.14) and Eq. (5.13). The connection between these probability distributions is that Eq. (5.13) is given by the integral of Eq. (5.14) over all possible exit-times $\tau > t$ weighted with their likelihood

$$g_\Omega(\vec{r}, t|0, 0) = \int_t^\infty d\tau\, g_\Omega(\vec{r}, t|\tau; 0, 0)\, \frac{q_\Omega(\tau|0, 0)}{S_\Omega(t|0, 0)}, \tag{5.16}$$

where $q_\Omega(\tau|0, 0)/S_\Omega(t|0, 0)$ is the probability that the particle exits at τ, conditional with respect to the information that at t is still inside the domain. From Eq. (5.16) it is possible to infer that, given the information that the particle at t is still inside the domain, the position sampled from Eq. (5.13) is on average the same as if sampled from Eq. (5.14), when the exit-time is extracted from Eq. (5.12). An evident assumption that has been made is that t and τ are uncorrelated, otherwise the integral in Eq. (5.16) could not be defined. This assumption is the essential reason that explains why sampling from Eq. (5.13) and from Eq. (5.14) are statistically equivalent, see Eqs. (5.15) and (5.16).

In Fig. 5.2, simulations have been performed using the two discussed probability distributions to update the particle position when domains are burst. Since in these simulations the domain bursts occur only in occasion of the random collision of an external particle with the protective domain of another particle, it is reasonable to assume that the bursting-time and the first exit-time are uncorrelated, hence sampling from Eq. (5.13) is statistically equivalent to sampling from Eq. (5.14). The mean squared displacement in a simple diffusive

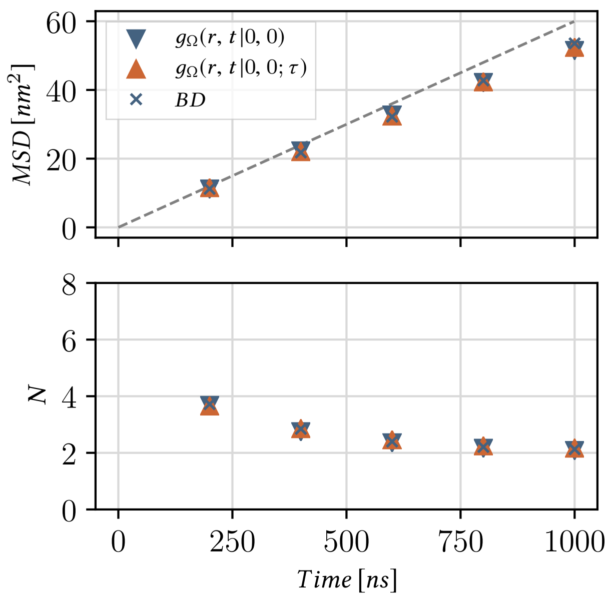


Figure 5.2: Top: Mean squared displacement of Brownian particles simulated with the hybrid MD-GFRD scheme as described in Ch. 4 and with a brute-force integration of the Brownian dynamics (BD). The hybrid scheme was chosen to maximize the number of bursts during simulations. The diffusion coeficient is $D = 10\,\mu m/s$ and the system volume is a box of length $14\,nm$ with periodic boundary conditions. Particles interact with a harmonic repulsive potential when within a distance of $5\,nm$. The dashed line is the expected value for a free diffusion, BD lies below this slope for crowding effects. Bottom: Kinetics of a two species annihilation reaction-diffusion system. The same simulation settings as above have been used. Particles are initially equally assigned to two different species A, B, and they annihilate when they collide with another particle of the same species, $A + A \to 0$ and $B + B \to 0$. In both plots, the MD-GFRD simulations have been performed using conditional ($g_\Omega(r, t|\tau; 0, 0)$, see Eq. (5.14)) and unconditional ($g_\Omega(r, t|0, 0)$, see Eq. (5.13)) distribution probabilities to sample the particle position at bursts, both showing a good agreement with the BD integration.

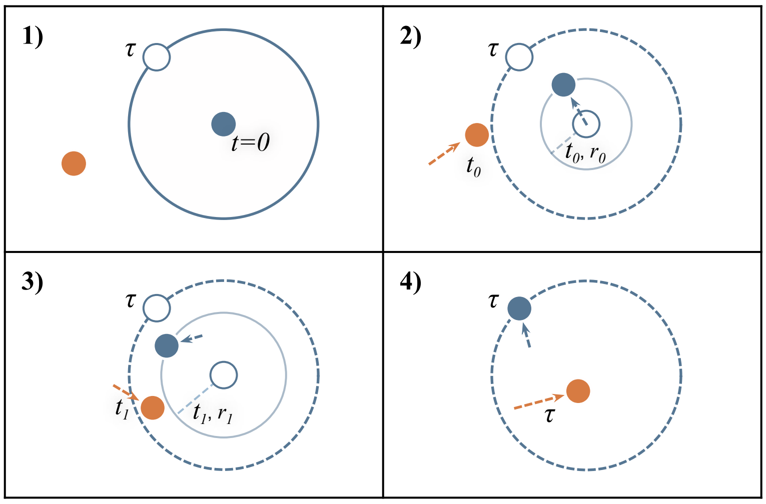


Figure 5.3: Graphical representation of the motion of a particle after the bursting of its domain. (1) The blue particle constructs a domain at time $t = 0$, and it samples τ as first exit-time from the domain. (2) The orange particle bursts the domain at time t_0. The blue particle samples from Eq. (5.14) its position r_0 at t_0 and all successive free displacements that will take place until the extracted exit-time τ. The relative displacements are recorded and successively used for simulating the free diffusion of the particle until τ. (3) The blue particle motion is integrated synchronously with the other particles. Given the linearity of the Langevin equation, it is possible to integrate the free diffusion component and the interaction with other particles separately. At each discrete temporal step the free diffusion displacement is taken from the relative displacements already sampled at the bursting-time, while the displacements due to interactions are iteratively given by the local configuration of the system. (4) In case of no previous interactions the particle is exactly located on the domain borders at τ.

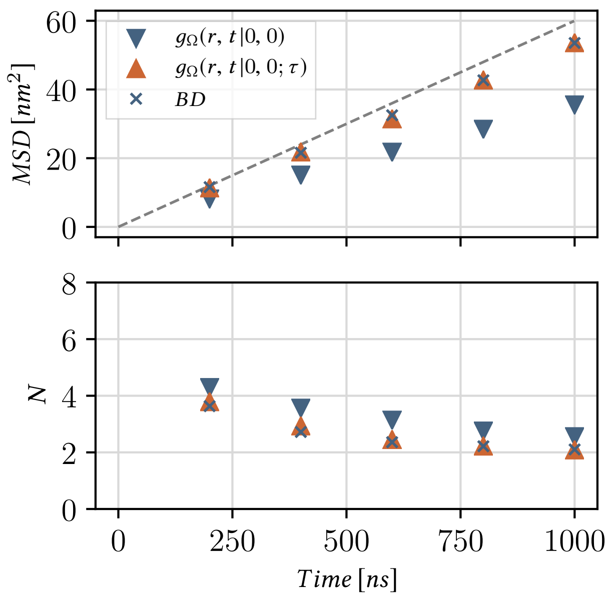


Figure 5.4: Mean squared displacement of Brownian particles and annihilation kinetics with the same properties and in the same system as described in Fig. 5.2. In contrast to the schemes outlined in Ch. 4, the clock synchronization is performed by sampling the particle position at the last discrete step before the exit-time. This procedure involves a systematic domain bursting, where the bursting-time is correlated to the exit-time, making the sampling from Eq. (5.13) invalid.

process and the kinetics of a two species annihilation reaction-diffusion system have been observed, and, as expected, the use of the conditional and unconditional distributions led to indistinguishable results.

In asynchronous schemes such as FPKMC and MD-GFRD the particle position is updated serially while other particles lie in protective domains, but it is still an open question how the motion of particles in such protective domains is characterized before their exit-time. Sampling the particles position from Eq. (5.13) simply gives an average behavior that is justifiable when the bursting-time is given randomly. More rigorously, the particle position should be sampled from Eq.

(5.14) until realization of the exit-time. Let us suppose that a domain is burst at a time t and the particle that was inside the domain interacts with another particle on the time grid $\{t_i\}_{i=0}^{n}$, where $t_{i+1} = t_i + dt$ and dt is the integration step. The particle position should be sampled from Eq. (5.14) at all time steps until the pre-sampled exit time, $t > \tau$. However, in Eq. (5.14) the domain has been assumed to be interaction-free until τ, but the particle is also assumed to be interacting. It is still possible to use the conditional sampling because the Langevin equation is linear, and the contribution of the free displacement can be separated from the interaction term. When the particle is burst at time t all particle displacements until τ are sampled and recorded. These will give the free displacements of the particle that will then be summed to the interaction term (see Fig. 5.3). The sampling procedure just enlightened involves many integration steps between the bursting time and the first-exit time, and in each of these steps the particle position is sampled from Eq. (5.14). This is clearly computationally more expensive than sampling once from Eq. (5.13), that can be done efficiently using the rejection method as described in Refs. [53, 50]

In the above derivations t and τ have been assumed uncorrelated, therefore it is possible to sample from Eq. (5.13). We have assumed the uncorrelation assumption reasonable because the bursting-time is given by the random collisions between different particles in a multi-particle simulation. However, a systematic bursting at a time related to the exit-time could be used in a multi-scale simulation as MD-GFRD. If the particle synchronization, performed in this multi-scale algorithm, is obtained by time-projection onto the last discrete time before the exit-time of the particle, this systematic bursting would clearly insert a correlation between t and τ. Eq. (5.16) would not be valid because it is not possible to integrate τ independently from t and in Eq. (5.15) $h(\vec{r} \in \Omega\, \forall t' \leq t)$ is not sampled.

In Fig. 5.4, the mean squared displacements of particles simulated as illustrated in Fig. 5.3 and particles that use the position sampling from Eq. (5.13) are shown and compared to a brute-force integration that is taken as reference. The synchronization step was performed before the exit-time during simulations, and it is clear from the figure that the use of Eq. (5.13) makes the particles diffuse more slowly. Indeed, immediately before the sampled exit-time the particle is expected to be in proximity of the domain borders, but this is not ensured without the condition on the exit-time (see Fig. 5.1). The correct slope of the mean square displacement is instead reproduced when the Eq. (5.14) is sampled. In the simulation, particles are interacting and their mean squared displacements lie below the expected free diffusion value due to crowding effects.

Chapter 6

Efficient sampling in first passage schemes

First passage schemes have been thoroughly discussed in previous chapters. In these asynchronous algorithms, Brownian particles perform large displacements in position and time, and each of these large displacements can correspond to many small time-driven displacements in a brute-force algorithm. The advantage of using an asynchronous first passage algorithm over a brute-force discretization lies in the lower total number of particle displacements in the simulation. In first passage schemes, in fact, particle updates are delayed to the next interaction event, but event-based sampling in first passage schemes is also computationally more demanding than in time-driven integrations. The efficiency of first passage schemes clearly depends also on the computational time necessary for event-based sampling, and how this compares to a brute force discretization of the Langevin equation. Particle displacements in a brute-force discretization of the Langevin equation is realized through a straightforward Gaussian extraction, an operation that can be implemented efficiently. Asynchronous moves must then also be optimized, otherwise, if the time needed to sample one asynchronous move is several orders of magnitude larger than the time needed to sample Gaussians, using first passage schemes is advantageous only for extremely dilute systems. In Chapter 4, however, it has been shown that a first passage scheme like MD-GFRD is more efficient than brute-force integration up to a concentration of $10^3 \mu M$. This result relied on the fact that large asynchronous displacements were sampled efficiently, and the sampling method used is described in detail in this chapter.

6.1 Sampling probability distributions with a root finding algorithm

In first passage schemes, particle displacements are carried out through a time extraction from $q_b(\tau)$ or a position extraction from $g_b(r,t)$. Target probability distributions $q_b(\tau), g_b(r,t)$ have already been derived and studied in previous chapters, and are written here again for the convenience of the reader :

$$q_\Omega(\tau) = -2 \sum_{n=1}^{\infty} (-1)^n \exp\left\{ -n^2 \frac{\pi^2 D\tau}{b^2} \right\} \frac{n^2\pi^2}{b^2} D, \tag{6.1}$$

$$g_\Omega(r,t) = \frac{1}{S_\Omega(t)} \sum_{m=1}^{\infty} \exp\left\{ -m^2 \frac{\pi^2 Dt}{b^2} \right\} \frac{2\pi r}{b^2} m \sin\left(\frac{m\pi r}{b}\right), \qquad r < b, \tag{6.2}$$

where $S_\Omega(t)$ is the survival probability. It is possible to sample the above distributions using samples generated from a uniform distribution on the unitary interval $w(x)$

$$w(x) = \begin{cases} 1, & 0 \le x \le 1. \\ 0, & otherwise. \end{cases} \tag{6.3}$$

Let us assume that we are interested in sampling a generic distribution probability $p(x)$ defined in the interval $[a, b]$, whose cumulative function $P(X)$ is defined as:

$$P(X) = \int_a^X p(x)\,dx, \qquad X \in [a, b]. \tag{6.4}$$

It can be noted that $P(X)$ is a monotonically not decreasing function, $p(x)$ never being negative. Two different probability distributions can be equated with an appropriate variable transformation $y = h(x)$

$$p(x)dx = \tilde{p}(y)dy, \tag{6.5}$$

where $\tilde{p}(y)$ is a not negative function defined in $y \in [h(a), h(b)]$ and $\int_{h(a)}^{h(b)} \tilde{p}(y)dy = 1$. The two cumulative functions are shown to be identical by integrating Eq. (6.4)

$$P(X) = \int_a^X p(x)\,dx = \int_{h(a)}^{h(X)} \tilde{p}(y)\,dy = \int_{h(a)}^{Y} \tilde{p}(y)dy = \tilde{P}(Y), \tag{6.6}$$

where $Y = h(X)$. If the cumulative of a uniform distribution is taken

$$W(\xi) = \int_0^\xi w(x)dx = \xi; \qquad \xi \in [0, 1], \tag{6.7}$$

which inserted in Eq. (6.6) results in

$$P(X) = W(\xi) = \xi, \tag{6.8}$$

where ξ is a number extracted from a uniform distribution in the unitary interval. Uniform distributions can be sampled with pseudo random number generators. The inversion of the equation above allows the extraction of a value X sampled from $P(X)$

$$X = P^{-1}(\xi). \tag{6.9}$$

The cumulative $S_\Omega(t)$ of the time probability distribution (see Eq. (6.1)),

$$S_\Omega(t) = -2\sum_{n=1}^{\infty}(-1)^n \exp\Big\{-n^2\frac{\pi^2 Dt}{b^2}\Big\}, \tag{6.10}$$

and the cumulative of the position distribution (see Eq. (6.2))

$$G_\Omega(r,t) = \frac{2}{b\pi S_\Omega(t)}\sum_{m=1}^{\infty}\exp\Big\{-m^2\frac{\pi^2 Dt}{b^2}\Big\}\left(\frac{b}{m}\sin\left(\frac{m\pi r}{b}\right) - r\pi\cos\left(\frac{m\pi r}{b}\right)\right) \tag{6.11}$$

are not analytically invertible, which makes the direct sampling as in Eq. (6.9) not applicable. However, $p(x)$ can be sampled from Eq. (6.8) also by finding numerically the root of the equation

$$P(X) - \xi = 0. \tag{6.12}$$

The root above can be approximated numerically by different algorithms, as the Newton-Raphson method. This method has the advantage of converging quadratically, but convergence to the root is not guaranteed and is quite sensitive to the starting point choice. However, if the root is searched in a function containing only one inflection point, and the inflection point is chosen as starting point, convergence is guaranteed. The target probability distributions $q_b(\tau), g_b(r,t)$ are unimodal, which involves their cumulative functions having only one inflection point, and this inflection point is used as starting point. The exact position of the inflection points in the cumulative functions $S_\Omega(\tau)$ and $G_\Omega(r,t)$ can be approximated accurately through a numerical study of the target distributions.

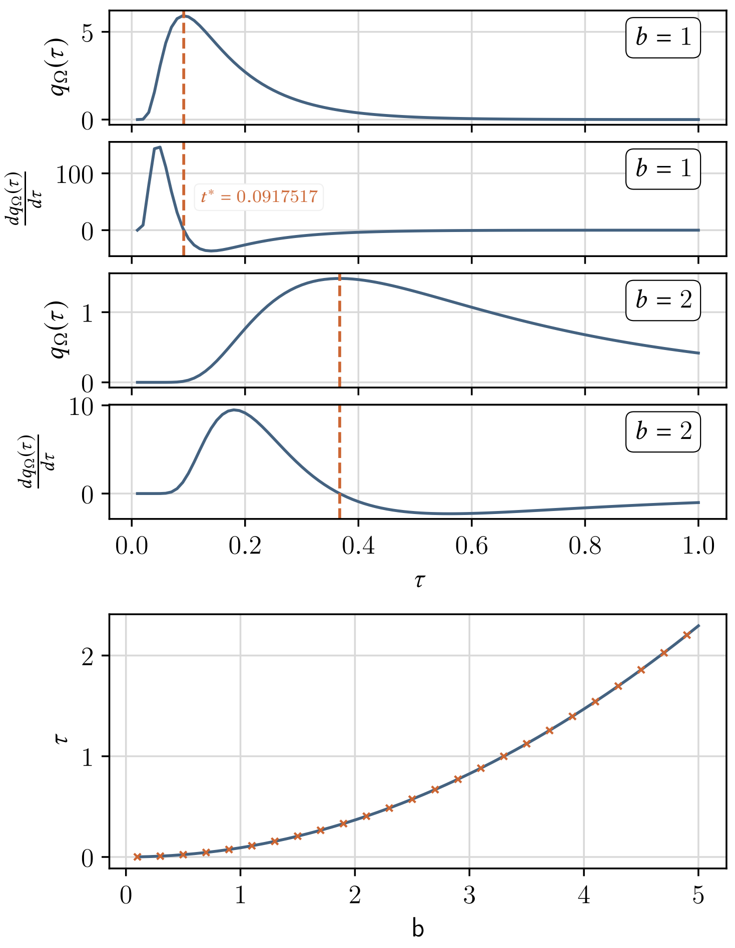


Figure 6.1: Location of the inflection point in the time cumulative function $S_\Omega(\tau)$. Top) First ($q_\Omega(\tau)$) and second ($\frac{dq_\Omega(\tau)}{d\tau}$) derivative of the cumulative function for different domain size $b = 1, b = 2$. The diffusion coefficient is fixed $D = 1$. The orange dashed bar is placed at the time of the inflection point. The inflection point is found numerically for $b = 1, D = 1$ at $\tau^* = 0.0917517$. Bottom) blue line - expected location of the inflection point following Eq. (6.15) for different domain size and fixed diffusion coefficient $D = 1$; orange cross - inflection point time found numerically. It is possible to see that the time of the inflection point scales quadratically with the domain size, as expected from Eq. (6.15).

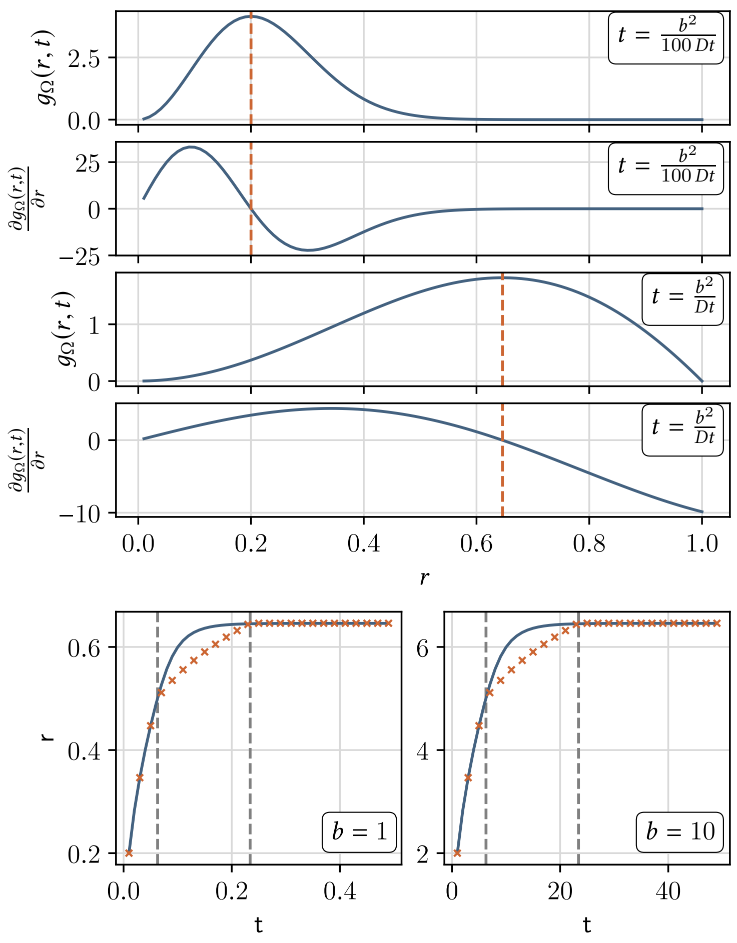


Figure 6.2: Location of the inflection point in the position cumulative function $P_\Omega(r,t)$. Top) First ($g_\Omega(r,t)$) and second ($\frac{dg_\Omega(r,t)}{dr}$)) derivative of the cumulative function at a small ($t = b^2/100\,Dt$) and large ($t = b^2/\,Dt$) time. The diffusion coefficient and the domain size are fixed $D = 1, b = 1$. The orange dashed bar is placed at the position of the inflection point. Bottom) blue line - expected location of the inflection point according to Eq. (6.21) for different domain size and fixed diffusion coefficient $D = 1$; orange cross - inflection point time found numerically. The two vertical dashed bars delimit short and large times ($t_{small} = 0.0646\frac{b^2}{D}$, $\quad t_{large} = 0.234\frac{b^2}{D}$).

6.2 Inflection point in the time cumulative function

The objective is to find a function of the parameters b, D aimed at accurately approximating the time position of the inflection point in the cumulative function $S_\Omega(\tau)$. After deriving twice the cumulative function, the inflection point is at the time τ^* that satisfies the following equation:

$$f_{b,D}(\tau^*) = \sum_{n=1}^{\infty}(-1)^n n^2 \exp\left\{ - n^2 \frac{\pi^2 D\tau^*}{b^2}\right\} = 0. \tag{6.13}$$

In the above equation, the time τ^* is multiplied by the scaling factor D/b^2. Hence, once the inflection point $\tau^{*'}$ is known for some specific parameters $b^{'}, D^{'}$, the inflection point can be inferred for other parameters' values

$$\tau^* \frac{D}{b^2} = \tau^{*'} \frac{D^{'}}{b'^2}. \tag{6.14}$$

The inflection point is found numerically at $\tau^{*'} = 0.0917517$ with parameters $b^{'} = 1$, $D^{'} = 1$. Then the inflection point can be inferred for other values according to the following rule:

$$\tau^* = 0.0917517 \frac{b^2}{D}. \tag{6.15}$$

This approximation is tested in Fig. 6.1, where a good agreement with values found numerically is evident.

6.3 Inflection point in the position cumulative function

The cumulative function $G_\Omega(r,t)$ depends on the position variable r in a different, more complex, manner compared to the dependence of the time cumulative function $S_\Omega(\tau)$ on τ, and finding a simple scaling rule as above is not possible.

Moreover, $G_\Omega(r,t)$ also has an additional external parameter, the observation time t, other than the domain size b and the diffusion coefficient D. However, based on the observation time t, it is possible to approximate the cumulative function $G_\Omega(r,t)$ to simpler functions where the position of the inflection point is found analytically. The observation time t is considered small or large considering how it compares to the coefficient b^2/D, which is a multiplicative factor of the observation time in Eq. (6.11). Based on this consideration, different approximations of the cumulative function can be used to find the inflection point.

In case of small time, it can reasonably be assumed that the probability of the particle having escaped the domain is small. The target probability distribution $g_\Omega(r,t)$ is obtained by solving the diffusion equation and imposing boundary conditions on the domain borders. Boundary conditions are applied on the probability flux once it reaches the domain borders, but if the time is small enough this probability is small, and the effect of the boundary conditions is small as well. Hence, at small times boundary conditions can be ignored, and $g_\Omega(r,t)$ approximated with a free diffusion probability $p_{free}(r,t)$

$$p_{free}(r,t) = \frac{r^2}{\sqrt{4\pi(Dt)^3}} e^{-\frac{r^2}{4Dt}}. \tag{6.16}$$

The maximum position, or inflection point of the cumulative, in the above equation r^* is found analytically

$$r^* = \sqrt{2Dt}. \tag{6.17}$$

The model that has been used at small times was based on a physical assumption, but as time increases this assumption is not verified; then, at large times, a different model is used, which is built upon different mathematical considerations. When time is large, the absolute value of exponents in the sum in Eq. (6.2), which scales with $\frac{Dt}{b^2}$, is also large. Considering that these exponents grow quadratically with the index element m in the series, all terms for $m > 1$ are negligible compared to $m = 1$ if the time is large enough. The equation then results in

$$g_b(r,t) = \frac{1}{S_\Omega(t)} \exp\left\{\frac{\pi^2 Dt}{b^2}\right\} \frac{2\pi r}{b^2} \, \sin\left(\frac{\pi r}{b}\right). \tag{6.18}$$

The position of the maximum of the equation above scales with the variable b, and the exact proportionality coefficient is found numerically

$$r^* \approx 0.646\, b. \tag{6.19}$$

Two different approximation models of the position cumulative function have been outlined. These models are alternatively valid at small or large times, and they allow us to find analytically the position of the inflection point. However, these two models are very different, and are not capable of approximating accurately the cumulative function at intermediate times. The small time model gives the inflection point as a function of the diffusion coefficient D and the observation time t, without considering the domain size b. On the contrary, in the large time model the maximum position scales with the domain size b, while the other parameters are ignored. In spite of these very different behaviors, considering that the particle is diffusing, we assume that the inflection point position is described over time by a monotonically not decreasing function. Given the position of the inflection point at a small and a large time, at intermediate times the position is found with an interpolation between the two known values. Thresholds that delimit small t_{small} and large t_{large} times are here defined after numerical investigations:

$$t_{small} = 0.0646\frac{b^2}{D}, \qquad t_{large} = 0.234\frac{b^2}{D}. \tag{6.20}$$

An exponential interpolation is then drawn between these two values. The inflection point of the cumulative r^* is approximated with different functions depending on the observation time t:

$$\begin{cases} t \leq t_{small} \Rightarrow r^* = 2\sqrt{Dt}, \\ t_{small} < t < t_{large} \Rightarrow r^* = \beta\left(1 - \exp\left(\frac{Dt}{b^2}\right)\right) + \gamma, \\ t \geq t_{large} \Rightarrow r^* = 0.646\, b, \end{cases} \tag{6.21}$$

where the parameters β and γ have been fixed to ensure continuity of the function

$$\begin{cases} \beta = \frac{(r^*_{small} - r^*_{large})\exp\left(\sqrt{t_{small}} + \sqrt{t_{large}}\right)}{\exp\left(\sqrt{t_{small}}\right) - \exp\left(\sqrt{t_{large}}\right)}, \\ \gamma = -\frac{r^*_{small}\left(\exp\left(\sqrt{t_{small}}\right) - 1\right)\exp\left(\sqrt{t_{large}}\right) - r^*_{large}\left(\exp\left(\sqrt{t_{large}}\right) - 1\right)\exp\left(\sqrt{t_{small}}\right)}{\exp\left(\sqrt{t_{small}}\right) - \exp\left(\sqrt{t_{large}}\right)}, \\ r^*_{small} = 2\sqrt{Dt_{small}}, \\ r^*_{large} = 0.0646\, b. \end{cases} \tag{6.22}$$

In Fig 6.2, it is possible to see that the location of the inflection points can be well approximated using the piecewise function introduced above at small and large times. Although the interpolation at intermediate times is not as accurate, the accuracy reached is still enough for a fast and robust convergence of the Newton-Raphson

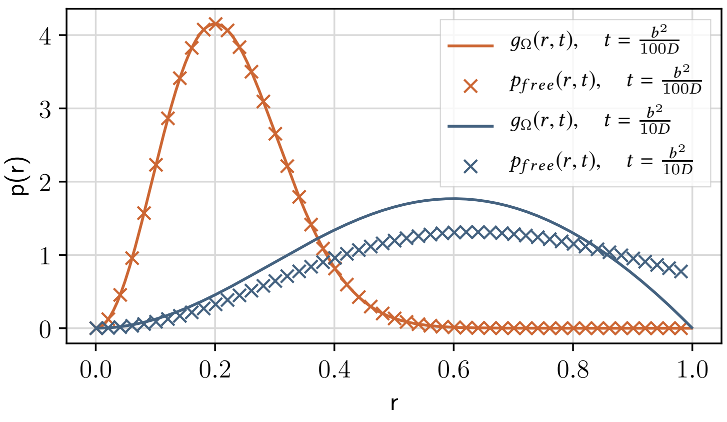


Figure 6.3: Comparison of the probability distributions in Eq. (6.2) and Eq. (6.16) at different times. In $g_\Omega(r,t)$ (Eq. (6.2)) absorbing boundary conditions on the domain Ω have been applied, while boundary conditions have not been applied in $p_{free}(r,t)$ (Eq. (6.16)). In the figure, it is possible to see that at small times $t < \frac{b^2}{100\,D}$ the two distributions are indistinguishable. The reason is that after a short time it is unlikely that the particle has escaped the domain, and the effect of the boundary conditions is negligible. As time increases, the probability of escaping the domain becomes significant, and the two distributions differ.

method, which will be discussed in the next section.

6.4 Numerical convergence

Newton-Raphson method has the main advantage of converging fast, but at each iteration it requires the computation of the function, whose root is searched, and its derivative. In the specific application of this method to cumulative functions as $S_\Omega(\tau)$ and $G_\Omega(r,t)$, it should also be ensured that convergence of sums in cumulative functions and their derivatives is fast enough to make the method in effect convenient (see Eq. (6.1), Eq. (6.2), Eq. (6.10), and Eq. (6.11)). Especially in the case of the radial distribution probabilities $g_\Omega(r,t)$ and $G_\Omega(r,t)$, converge can be extremely slow as the observation time becomes very small $t << b^2/D$. Application of the Newton-Raphson method in this situation can be avoided because sampling $g_\Omega(r,t)$ is not necessary at this time scale. As already mentioned, when observation time is very small, the particle distribution can be modeled without imposing absorbing boundary conditions, and position can be sampled from Eq. (6.16). In Fig. 6.3, it is possible to see that $g_\Omega(r,t)$ and $p_{free}(r,t)$ are indistinguishable as the observation time is set to $t = b^2/(100\,D)$; this value is then taken as threshold: when the observation time is smaller, position is sampled directly from Eq. (6.16).

Convergence of sums in Eq. (6.1), Eq. (6.2), Eq. (6.10), and Eq. (6.11) is verified because the succession in the series reaches zero asymptotically, which is ensured by the exponential decay in the succession. The series must be truncated in order to be evaluated numerically, and this can be done within a controllable truncation error. In Fig. 6.5, the number of evaluated elements in the succession to reach converge within a fixed error is shown. At small times, convergence happens more slowly, but it rarely needs more than 20 iterations to reach convergence. Overall, the number of iterations is low, mostly below 10.

Given the fast convergence of sums in target distributions, Newton-Raphson method can be efficiently applied, and this method ensures fast convergence. In Fig. 6.4, the Newton-Raphson method is applied to find roots of equations as Eq. (6.12) for different values of ξ and for the cumulative functions $S_\Omega(\tau)$ and $G_\Omega(r,t)$. Initial conditions are set as previously explained, and the method reaches always convergence.

Finally, the method is tested for a large number of random extractions, and in Fig. 6.6 it is possible to see that the target distributions are correctly reconstructed.

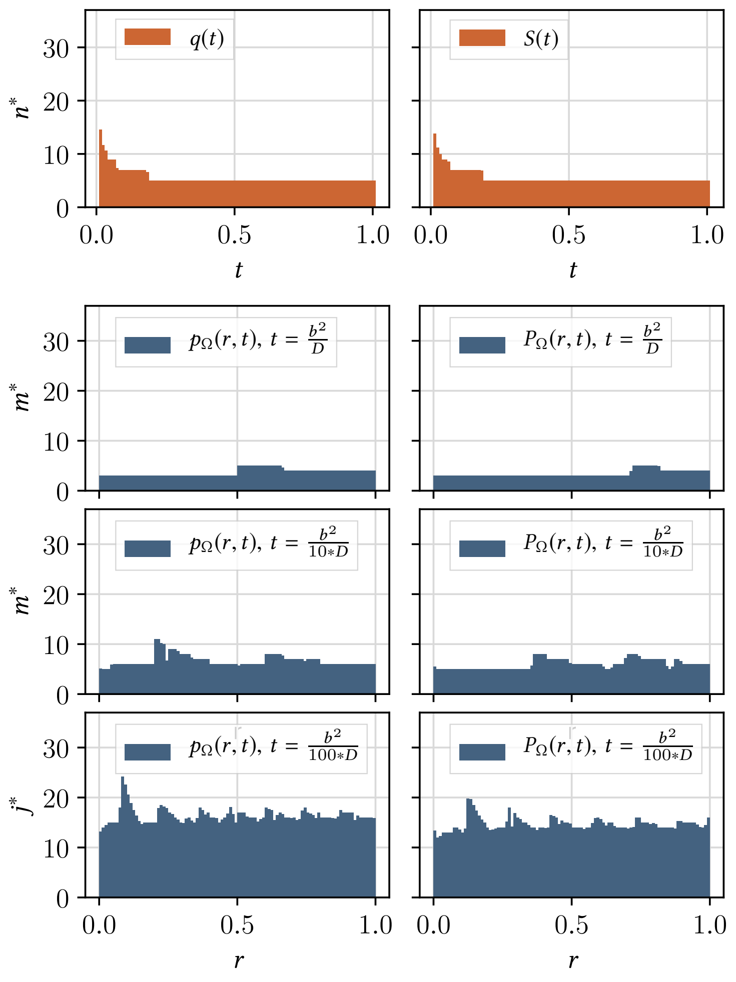


Figure 6.4: Iterations necessary to reach convergence in sums in Eq. (6.1), Eq. (6.2), Eq. (6.10), and Eq. (6.11). Parameters are fixed $D = 1, b = 1$, and radial distributions are studied for different observation time t. Convergence of the sum to a finite value is ensured because the succession decreases exponentially. Sums have alternating sign, and, when convergence is evaluated, the sum of two consecutive elements of opposite sign is considered as one term in the succession. A convergence threshold is fixed to 10^{-5}, and convergence is reached when the sum of two elements in the succession is smaller than the convergence threshold. The truncation error is then controllable, and it is smaller than the convergence threshold.

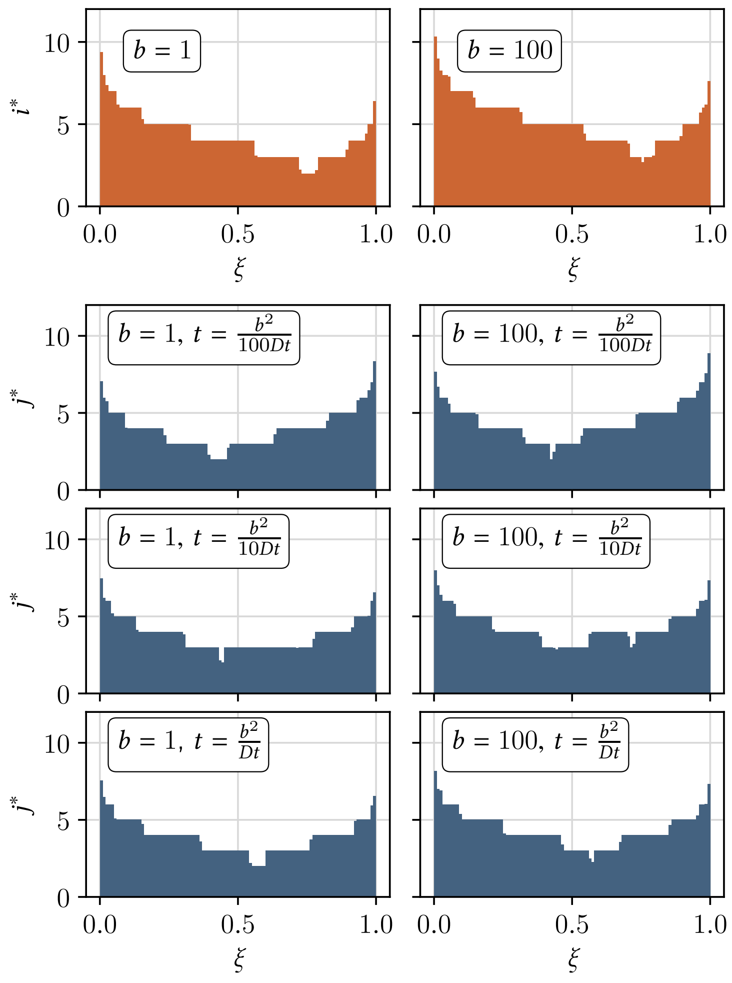


Figure 6.5: Number of iterations in the Newton-Raphson method to find the roots of $S_\Omega(\tau) - \xi$ (see Eq. 6.10) and $G_\Omega(r,t) - \xi$ (see Eq. 6.10). Diffusion coefficient is fixed $D = 1$, while the domain size assumes two different values $b = 1, 100$. Starting point of the method is chosen in proximity of the inflection point, and convergence is ensured for all different values of ξ. It is possible to see in the figure that convergence is always fast, at most about 10 iterations.

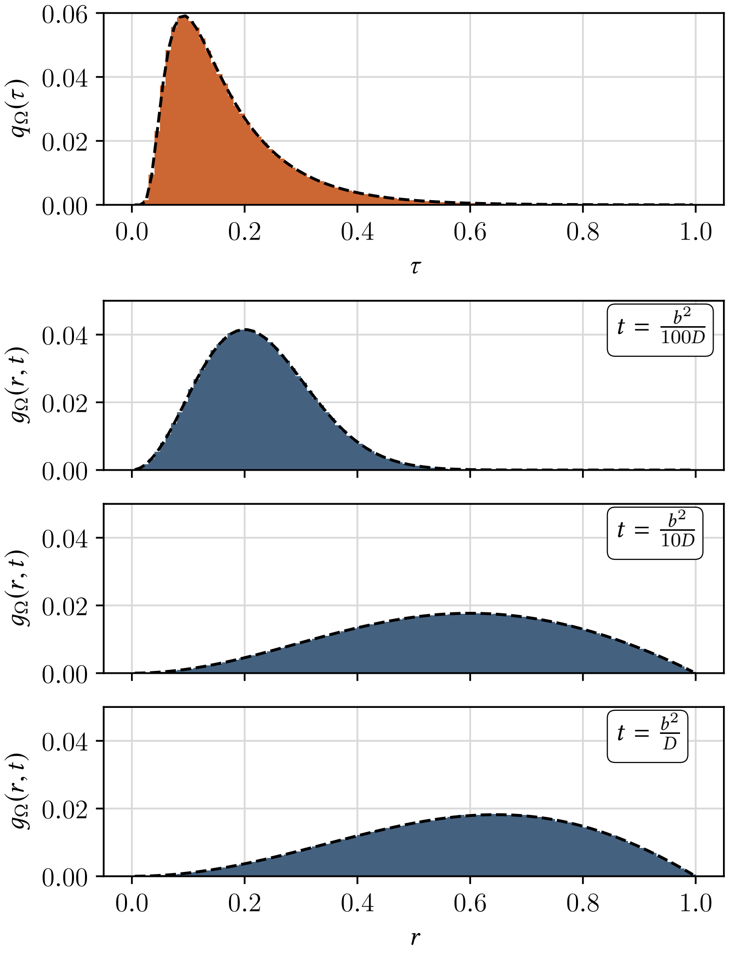


Figure 6.6: Target distributions $q_\Omega(\tau)$ and $g_\Omega(r,t)$ are reconstructed using values sampled from a uniform distribution in the unitary interval. Parameters are fixed $b = 1, D = 1$, and $g_\Omega(r,t)$ is evaluated at different times. From random uniform samples, values of τ or r are obtained using cumulative functions and the Newton-Raphson method to find the root. These values are then grouped into different bin intervals and plotted as a histograms. Black dashed curves are numerical approximations of the sums in the target distributions in Eq. (6.1) and Eq. (6.2). In the figure, it is shown that the outlined sampling method accurately reconstructs the target distributions.

Part II

Atomistic Scale

Chapter 7

Markov chain Monte Carlo

Since the beginning of their invention, computers have being used for simulations involving pseudo-random number extractions, or also called Monte Carlo simulations. Markov Chain Monte Carlo (MCMC) methods have been introduced by Metropolis et al. in 1953 to compute "the properties of any substance which may be considered as composed of interacting individual molecules" [62]. In the Metropolis algorithm, molecules follow a random walk in the phase space, but individual moves are randomly accepted with an acceptance rate determined by the energy difference between initial and final configuration. At each iteration, a new configuration is generated with a trial move, then, if the proposed state is accepted, the system jumps into that configuration, otherwise it remains in the previous configuration. The ensemble of states generated with this procedure forms a Markov chain that asymptotically samples the equilibrium distribution, and is used to compute averages of macroscopic observables.

The Metropolis algorithm was generalized by Hastings in 1970 [63]. In this more general formalization it is suggested that molecules must not necessarily move following a random walk, but trial configurations are drawn from selection probabilities that can be arbitrarily designed for the specific system that is investigated. The probability used to accept trial moves then depends not only on the energy difference between final and initial configuration, but also on the degree of reversibility of the selection move. If the probability to propose a state is the same as the probability to select back the original state from the proposed one, moves are perfectly reversible and the reversibility factor does not appear in the acceptance rate. For example, when moves are generated with a random walk in the phase space as in the original Metropolis scheme, moves are reversible and the acceptance rate is only determined by the energy difference. On the other hand, if trial moves are generated with an arbitrary algorithm, reversibility in the acceptance probability must be evaluated at each iteration. The specific scheme used to propose moves plays a key factor in the convergence rate of the Markov chain, and in his work Hastings already warns against "high rejection rates as indicative of a poor choice of transition matrix", where transition matrix indicates the ensemble of selection probabilities defined in the system. Since then, the development of efficient proposal moves to accelerate the convergence rate in MCMC has been an active research field.

In this chapter, the Metropolis-Hastings algorithm is outlined. It is demonstrated that the Markov chain generated with this algorithm asymptotically samples the equilibrium distribution of condensed-matter systems at thermodynamic equilibrium. However, if moves are proposed with a local random walk in the configuration space, this algorithm produces highly correlated samples, and the evaluation of macroscopic observables becomes computationally challenging for multi-dimensional systems. The choice of the algorithm for proposing trial configurations is crucial for fast convergence, and in the next chapter it is shown how this convergence rate can be dramatically increased with the aid of deep-learning in neural mode jump Monte Carlo.

7.1 Sampling equilibrium distributions

We study a stochastic system characterized by an equilibrium distribution $\pi(\vec{x})$, and our goal is to compute the expected value $< A >$ of an observable quantity $A(\vec{x})$:

$$\langle A \rangle = \int d\vec{x}\, A(\vec{x})\pi(\vec{x}). \tag{7.1}$$

Computational time required to numerically solve this integral scales with the phase space volume where it is defined, hence numerical integration is not a reasonable choice for highly dimensional systems. On the other hand, assuming that we are able to generate independent samples from the equilibrium distribution, an estimate of the average value A_M over M samples is obtained from the following sum:

$$\tilde{A}_M = \frac{1}{M}\sum_{i=0}^{M} A(\vec{x}^i). \tag{7.2}$$

The central limit theorem then ensures that this sum converges to the expected value

$$\langle A \rangle = \lim_{M\to\infty} \tilde{A}_M. \tag{7.3}$$

Many systems of interest in biology are at thermal equilibrium with a reservoir at fixed temperature T. If the investigated system is closed, i.e. the number of molecules is fixed, it is statistically described by the canonical ensemble and the microscopic probability to be in the state $\vec{x}$ is given by the Boltzmann distribution

$$\pi(\vec{x}) = \frac{e^{-E(\vec{x})/K_B T}}{Z}, \tag{7.4}$$

where K_B is the Boltzmann constant, and Z is the partition function

$$Z = \int e^{-E(\vec{x})/K_B T}\, d\vec{x}. \tag{7.5}$$

The next step is then to understand how a probability distribution as in Eq. (7.4) can be sampled efficiently.

7.2 Brute-force sampling of multidimensional probability distributions

An arbitrary mono-dimensional probability distribution can be sampled using samples from the uniform distributions as described in Ch. 6. Assuming that the system has N dimensions $\pi(\vec{x}) = \pi(x_1, \dots, x_N)$, we then would like to obtain N separate mono-dimensional probability distributions, and individually sample from each of them. In the trivial case of independent variables the joint distribution is directly factorizable

$$\pi(x_1, \dots, x_N) = \pi(x_1), \dots, \pi(x_N). \tag{7.6}$$

However, usually there are correlations between variables, and a factorization into conditional probabilities is instead required

$$\pi(x_1, \dots, x_N) = \pi(x_1), \pi(x_2|x_1), \dots, p(x_N|x_1, \dots, x_{N-1}). \tag{7.7}$$

A brute-force routine to sample the above conditional distribution is defined below. Firstly, we sample from the mono-dimensional distribution $\pi(x_1)$, and this distribution is obtained straightforwardly with a marginalization of the joint probability

$$\pi(x_1) = \int \pi(x_1, \dots, x_N)\, dx_2 \dots dx_N. \tag{7.8}$$

The value x_1^* is then directly sampled from $\pi(x_1)$, and this specific value conditions successive extractions through correlations. Hence, to obtain the conditional probability distribution $\pi(x_2|x_1)$, the joint probability $\pi(x_1, x_2)$ is required, which is found by marginalizing the joint distribution over the other $N-2$ variables

$$\pi(x_1, x_2) = \int \pi(x_1, \ldots, x_N)\, dx_3 \ldots dx_N. \tag{7.9}$$

Then, the Bayes theorem is applied

$$\pi(x_2|x_1) = \frac{\pi(x_1, x_2)}{\pi(x_1)}. \tag{7.10}$$

The value x_1^* is inserted into $p(x_2|x_1)$, and x_2^* is extracted from the conditional distribution. In turn, the same routine is repeated to sample the third dimension, and this is iterated for all N dimensions. The collection of N values $x_1^*, \ldots, x_N^*$ represents one sample of $\pi(\vec{x})$.

The just outlined brute-force approach has the advantage that each sample $\vec{x}^i$ is uncorrelated with respect to the previous one $\vec{x}^{i-1}$, consequently convergence of Eq. (7.2) is attained after a relatively small number of samples M. However, for each dimension two integrals must be solved: one to marginalize the joint probability, and one for the cumulative function (see ch. 6). Excluding trivial cases, these integrals do not have an analytical solution, and integrations are computationally demanding, especially when the marginalization is performed over a large number of dimensions. Interesting biological systems usually are highly multi-dimensional, and a brute-force sampling is computationally unfeasible.

7.3 Metropolis-Hastings algorithm

The generation of equilibrium samples for complex many-body systems is arduous if this involves computations over the whole configuration space. In fact, the canonical ensemble is characterized by a peaked probability distribution that can be largely approximated to zero. The fraction of configuration space that is relevant in integrals as in Eq.

(7.1) is vanishingly small for many dimensions, and the iteration of numerical operations over this irrelevant region is unnecessary. Rather, once the relevant region has been found in the configuration space, it would be convenient to employ this knowledge in the successive sampling and explore the surrounding area. Given the system in a certain configuration, the probability distribution to sample the next configuration should be conditional with respect to the current one, but memory of previous configurations is not relevant. This generates a memoryless stochastic process, also known as Markov chain.

The Metropolis-Hastings algorithm generates a Markov chain that aims to sample a specific target distribution, in our case the Boltzmann distribution. At each iteration, given the system in the state $\vec{x}_{t_n}$, a transition probability $p(\vec{x}_{t_{n+1}}|\vec{x}_{t_n})$ is defined, and the next state $\vec{x}_{t_{n+1}}$ is drawn from this probability. The ensemble of states $\vec{x}_{t_0}, \ldots, \vec{x}_{t_n}, \vec{x}_{t_{n+1}}, \ldots$ visited by the system represents the Markov chain. To facilitate the discussion, the system is now assumed to be discrete, results can then be generalized to continuum. In a discrete system there is a finite number of transition probabilities $p_{ij}^{(1)}$ to move with one transition from a state $\vec{x}_i$ to a state $\vec{x}_j$, and the probability $p_{ij}^{(2)}$ to move from state i to state j with two transitions is decomposable into two consecutive one-transition moves

$$p_{ik}^{(2)} = \sum_{j=1}^{Q} p_{ij}^{(1)} p_{jk}^{(1)}, \tag{7.11}$$

where Q is the total number of discrete states. In turn, the same reasoning is applicable to the n-th transition

$$p_{ik}^{(n)} = \sum_{j=1}^{Q} p_{ij}^{(n-1)} p_{jk}^{(1)}. \tag{7.12}$$

Assuming that the states i and j belong to the same ergodic class, it is possible to show [64] that the limit

$$\lim_{n\to\infty} p_{ij}^{(n)} = \pi_j, \qquad j = 1, 2 \ldots, Q \tag{7.13}$$

exists for each j, and is independent of i. Initial conditions are then irrelevant, because after a certain number of transitions the system loses memory, and the probability π_j to be in a state j does not depend on the initial state i. Inserting the above equation into Eq. (7.12)

$$\pi_k = \sum_{j=1}^{Q} \pi_j p_{jk}. \tag{7.14}$$

This sum represents an eigenvector equation relative to the eigenvalue 1, that can be solved to find the eigenvector elements $\{\pi_k\}$ from the transition probability $\{p_{ik}\}$. In our case, however, the problem is the opposite, because the target distribution $\{\pi_k\}$ is known, i.e. the Boltzmann distribution, but the transition probabilities $\{p_{jk}\}$ are to be determined. The above equation can be rewritten as:

$$\sum_{j=1}^{Q} \pi_k p_{kj} = \sum_{j=1}^{Q} \pi_j p_{jk}, \tag{7.15}$$

where it has been considered that $\sum_{j=1}^{Q} p_{kj} = 1$, i.e. starting from j the system will certainly move into one of the Q states. One possible solution of Eq. (7.15) is given by the condition of detailed balance

$$\pi_k p_{kj} = \pi_j p_{jk}. \tag{7.16}$$

It is now possible to decide on the target distribution $\{\pi_k\}$ that is asymptotically sampled from the transition probabilities $\{p_{jk}\}$. A discrete system has been used for simplicity in this derivation, but the discussion can be generalized to a continuous target distribution $\pi(\vec{x})$ and a continuous transition probability $p(\vec{x} \to \vec{y})$.

Metropolis choice is to divide the transition probability into two different logical steps: firstly, a new state is selected from a selection probability $p_{prop}(\vec{x} \to \vec{y})$; then, the proposed state is accepted with an acceptance probability $p_{acc}(\vec{x} \to \vec{y})$. If the new state $\vec{y}$ is accepted, this is added to the Markov chain and the next proposal starts from $\vec{y}$; otherwise, the starting state $\vec{x}$ is added to the Markov chain and the next proposal starts again from $\vec{x}$. According to this decomposition, the condition of detailed balance becomes:

$$\pi(\vec{x})p_{prop}(\vec{x} \to \vec{y})p_{acc}(\vec{x} \to \vec{y}) = \pi(\vec{y})p_{prop}(\vec{y} \to \vec{x})p_{acc}(\vec{y} \to \vec{x}). \tag{7.17}$$

If trial configurations are proposed with a random displacement in the configuration space, as in the original Metropolis work, proposal probabilities cancel out in the condition of detailed balance

$$e^{-\beta E(\vec{x})}p_{acc}(\vec{x} \to \vec{y}) = e^{-\beta E(\vec{y})}p_{acc}(\vec{y} \to \vec{x}), \tag{7.18}$$

where the Boltzmann distribution (Eq. (7.4)) has been used as target distribution and $\beta = K_B T$. The relative value of acceptance probabilities is determined by the energy difference between initial and final state:

$$\frac{p_{acc}(\vec{x} \to \vec{y})}{p_{acc}(\vec{y} \to \vec{x})} = e^{-\beta(E(\vec{y})-E(\vec{x}))}. \tag{7.19}$$

In the above equation the absolute value of the acceptance probabilities is not given, and there is still a degree of freedom that can be used to maximize the efficiency of the Markov chain. An indicator of efficiency is given by the capability of the Markov chain to generate independent samples, and generation of independent samples is clearly accelerated if trial moves are frequently accepted. The acceptance probability is then to be maximized with the constraint of having to fulfill the detailed balance, and, obviously, it must be lower than or equal to 1 because it is a probability. This can be achieved with the following choice for the acceptance probability:

$$p_{acc}(\vec{x} \to \vec{y}) = \begin{cases} e^{-\beta(E(\vec{y})-E(\vec{x}))}, & \text{if } E(\vec{y}) > E(\vec{x}). \\ 1, & \text{otherwise.} \end{cases} \tag{7.20}$$

To sum up, the Metropolis-Hastings scheme is an algorithm to generate a Markov chain. This has been proven to asymptotically sample a target distribution, in our case the Boltzmann distribution. After an equilibration time, the algorithm generates states located in a likely region of the system, and the probability of visiting a certain state is proportional to the Boltzmann probability. Hence, states generated with the Metropolis-Hastings algorithm can be used to compute average quantities as in Eq. (7.2).

After proving that the Metropolis-Hastings algorithm samples the Boltzmann distribution, we are concerned about the efficiency of the Markov chain, i.e. the rate at which independent samples are generated. Proposing states with a random walk in the phase space allows us to ignore proposal probabilities in Eq. (7.19), because moves are fully reversible, and this simplifies the acceptance probability. However, random displacements must be very small to avoid the acceptance probability dropping to zero. The reason of this is clear from Eq. (7.19), where the acceptance probability decays exponentially with the energy difference, and in rugged energy landscapes random selections of new states often involve large energy differences. Choosing small displacements in a local MCMC allows us to obtain a high acceptance rate, but samples generated are also highly correlated. More specifically, local MCMC allows for an accurate local exploration of the phase space, but crossing energy barriers remains difficult. However, most biologically relevant systems are composed of many metastable states that are separated by high energy barriers, and it is required to cross these energy barriers several times during simulations to converge to the equilibrium distribution. It is practically impossible to design manually long range moves that are capable of making a direct transition into another metastable state, but this can be achieved with the aid of deep-learning. This is the core idea of Neural Mode Jump Monte Carlo, which is outlined in the next chapter.

Chapter 8

Neural Mode Jump Monte Carlo

Text in this chapter has been previously published in:
L. Sbailò, M. Dibak, and F. Noé. (2019) Arxiv:1912.05216.

Local Markov chain Monte Carlo (MCMC) is not suited to sample the equilibrium distribution of systems composed of many metastable states that are separated by high energy barriers. In fact, transitions between different metastable states might require an enormous computational effort, and it is often impossible to cross the energy barriers many times with unique simulations. The impossibility of visiting all relevant regions in the configuration space leads to a problem known as broken ergodicity, or quasiergodicity, which implicates that simulations do not converge properly. In the last decades, many different methods have been developed to circumvent the problem of broken ergodicity. One of the most widely recognized methods to enhance MCMC sampling in quasiergodic systems is simulated tempering [65]. Simulations based on simulated tempering randomly change the temperature of the sampler while remaining at equilibrium in an augmented configuration space. The strength of this method relies on the fact that trajectories are more likely to cross energy barriers as temperature is increased, and raising the temperature allows for a faster exploration of the configuration space. However, a temperature change is likely to be accepted only if there is a significant overlap in the energetic distribution sampled at the two temperatures, and this can be a problem if the energy barriers are particularly high and a wide range of different temperatures is required to cross them. The effectiveness of this method is, in fact, still susceptible to the height of the energy barriers. Another approach recently developed [66, 67, 68] constructs reversible moves between equilbrium states as a collection of short out-of-equilbrium trajectories. However, this approach also depends on the path connecting the equilibrium states, and a system specific protocol to generate the candidate state must be designed. Differently, Smart Darting Monte Carlo [69] is a promising method that alternates local moves and long range moves from one region of the configuration space to another that is arbitrarily far, outperforming parallel tempering in a number of relevant potential energy surfaces [70]. Small spheres of radius ϵ are drawn around local minima, and, when the system is within one of those ϵ-spheres, long range moves into another minimum can be attempted. The system is projected in proximity of the new minimum, and projection is performed so as to maintain the relative distance to the closest minimum. The rule used to generate long range displacements is quite straightforward, and it does not take into account the local energetic conformation of the different minima. However, the energy difference proposed in trial moves must be minimal according to Metropolis criterion [62], thus the volumes of ϵ-spheres are forced to be small. Local explorations can be long before entering ϵ-spheres, which makes this method difficult to apply in highly dimensional systems where the fraction of ϵ-spheres volume becomes vanishingly small. This problem is circumvented in ConfJumps [71] by projecting the system into the closest minimum, and there long range moves are attempted. In doing so, it is not required that small ϵ-spheres are randomly found through local displacements, but this approach still requires to search the configuration space before running MCMC simulations to find all local minima.

The generation of long range moves is challenging when the energy landscape is rough, since the potential energy surface in the region surrounding local minima can drastically change between the different minima. In this case, using trivial projections as long range moves would mostly cause large energy differences, and trial moves are likely rejected. Instead, a specific bijective function pairing points so as to keep the energy difference small should be employed, but constructing such bijection analytically is practically impossible in multi-dimensional systems, which substantially limits applications of this method. On the other hand, recent advances in the field of machine learning have permitted to deal with problems that were not solvable with a sole human understanding, and, more specifically, deep neural networks (DNNs) are an ideal tool to facilitate the construction of the bijective function we are concerned about. DNNs can approximate a target function with arbitrary precision, and, in fact, they have already been applied to enhance statistical sampling [72]. DNNs have also been employed to construct MCMC moves, but current methods either optimize a self-learning process over a known path in a discrete configuration space [73] or are applied to bi-dimensional systems [74]. An agnostic method, not system specific, capable to connect through DNNs regions that are distant in the configuration space, and applicable to multi-dimensional systems in continuum, is still missing.

Neural mode jump Monte Carlo (neural MJMC) is a novel method to sample efficiently the equilibrium distribution of complex many-body systems with unbiased Markov chains. In this, scheme neural networks are trained to propose "neural" moves that directly connect different metastable states. Local displacements and neural moves are alternated in a combined scheme to accelerate the convergence rate of Markov chains. A general formalism is proposed, where local moves and neural moves are generated with specific kernels, and kernels are randomly selected according to their selection probability. The combined scheme satisfies detailed balance which requires that kernels must be reversible, and, to facilitate that, neural moves are produced with bijective functions that are constructed with reversible DNNs. Equilibrium configurations are collected in the different metastable states and are used to train the networks, which are then optimized to produce high acceptance probability moves from one metastable state to another. Local exploration ensures ergodicity of the scheme, while neural moves accelerate convergence to equilibrium, realizing an accurate and deep exploration of the configuration space.

8.1 Theory

Markov chains asymptotically sample the equilibrium distribution if ergodic and if the condition of detailed balance is satisfied. Given the system in a configuration $\vec{x}$ a new state $\vec{y}$ is added to the chain with a transition probability $p(\vec{x} \to \vec{y})$. The transition probability is defined to satisfy the condition of detailed balance

$$\pi(\vec{x})\, p(\vec{x} \to \vec{y}) = \pi(\vec{y})\, p(\vec{y} \to \vec{x}), \tag{8.1}$$

where $\pi(\vec{x})$ is the equilibrium distribution. According to Metropolis-Hastings algorithm [62, 63], the transition probability is decomposed into two logical steps: firstly, a new configuration $\vec{y}$ is selected with a proposal probability $p_{prop}(\vec{x} \to \vec{y})$; then, the new state is accepted with an acceptance probability $p_{acc}(\vec{x} \to \vec{y})$. If the transition is accepted, the new state $\vec{y}$ is added to the Markov chain, otherwise the previous state $\vec{x}$ is added to the Markov chain.

In neural MJMC, the proposal probability is in turn decomposed into two logical steps: firstly, a selection-kernel K^i is extracted with a probability $p^i_K(\vec{x})$ from a pre-defined list of kernels $\{K^i\}_{i=1}^{N_K}$, where N_K is the total number of kernels in the system; then, a new state $\vec{y}$ is drawn from the proposal probability $p^i_{prop}(\vec{x} \to \vec{y})$ that is associated to the extracted selection-kernel K^i. The resulting transition probability is thus decomposed into three different steps, and the condition of detailed balance is generalized as follows:

$$\frac{p_{acc}(\vec{x} \to \vec{y}) \sum_i^{N_K} p^i_K(\vec{x})\, p^i_{prop}(\vec{x} \to \vec{y})}{p_{acc}(\vec{y} \to \vec{x}) \sum_j^{N_K} p^j_K(\vec{y})\, p^j_{prop}(\vec{y} \to \vec{x})} = \frac{\pi(\vec{y})}{\pi(\vec{x})}. \tag{8.2}$$

Kernels are distinguished between local kernel and neural kernels, where the local kernel generates local moves through Gaussian extractions, as already proposed in Metropolis scheme, and neural kernels connect different metastable states. Let us assume that the configuration space Ω is decomposed into a number of non overlapping subsets called cores, each representing a different metastable state. We assume that each neural kernel is accessible within only one specific core. Given the neural kernel K^i, this is accessible only within the core Ω_α, and the probability to select the kernel K^i is constant within the whole core, $p^i(\vec{x}) = p^i$ if $\vec{x} \in \Omega_\alpha$, otherwise $p^i(\vec{x}) = 0$. The formalism indicating the list of probabilities to select a kernel $\{p^i_K(\vec{x})\}_{i=0}^{N_K}$ is then simplified with the use of characteristic functions $\{\chi_i(\vec{x})\, p^i_K\}_{i=0}^{N_K}$, where $\chi_i(\vec{x}) = 1$ if $\vec{x} \in \Omega_\alpha$, otherwise $\chi_i(\vec{x}) = 0$. The local kernel is instead defined over the whole configuration space, thus enforcing ergodicity. The characteristic function associated to the local kernel always has unitary value. The acceptance probability is then defined according to

Metropolis choice

$$p_{acc}(\vec{x} \to \vec{y}) = \min\left\{1, \frac{\pi(\vec{y}) \sum_i^{N_K} \chi_i(\vec{y})\, p_K^i\, p_{prop}^i(\vec{y} \to \vec{x})}{\pi(\vec{x}) \sum_j^{N_K} \chi_j(\vec{x})\, p_K^j\, p_{prop}^j(\vec{x} \to \vec{y})}\right\}. \tag{8.3}$$

Let us take two different states in the configuration space $\vec{x} \in \Omega_\alpha$, $\vec{y} \in \Omega_\beta$. If the states are defined in different cores ($\alpha \neq \beta$), we assume that there is only one neural kernel K^j whose probability $p^j(\vec{x} \to \vec{y})$ to generate $\vec{y}$ from $\vec{x}$ is larger than zero, and, to attain reversibility, there is only one neural kernel K^i whose probability $p^i(\vec{y} \to \vec{x})$ is larger than zero. If instead the states are defined in the same core ($\alpha = \beta$), we assume that the transition between the two states can be generated only through the local kernel. These assumptions allow to simplify the acceptance probability

$$p_{acc}(\vec{x} \to \vec{y}) = \min\left\{1, \frac{\pi(\vec{y})\, p_K^i\, p_{prop}^i(\vec{y} \to \vec{x})}{\pi(\vec{x})\, p_K^j\, p_{prop}^j(\vec{x} \to \vec{y})}\right\}. \tag{8.4}$$

In case $\vec{x}$ and $\vec{y}$ belong to the same core the transition between the two states is possible only with the local kernel, that is trivially reversible, and proposal probabilities simplify in the acceptance probability. Still, we desire to simplify proposal probabilities also when $\vec{x}$ and $\vec{y}$ belong to different cores. Given the neural kernels K^i and K^j connecting the cores Ω_α and Ω_β, we define a bijective function $\mu_{\alpha\beta}(\cdot)$ pairing the states defined in the two cores, i.e $\vec{y} = \mu_{\alpha\beta}(\vec{x})$, $\mu_{\alpha\beta}^{-1}(\vec{y}) = \vec{x}$, $\forall \vec{x} \in \Omega_\alpha$, where $\vec{x} \in \Omega_\alpha$, $\vec{y} \in \Omega_\beta$. The bijective function $\mu_{\alpha\beta}(\cdot)$ and its inverse $\mu_{\alpha\beta}^{-1}(\cdot)$ are then used as neural kernels K^i and K^j, which involves that for each pair of different cores $\Omega_\alpha, \Omega_\beta$ a bijective function $\mu_{\alpha\beta}(\cdot)$ is defined. The selection probability of neural moves, which are generated with deterministic displacements, is then approximated with

Dirac delta functions

$$p_{acc}(\vec{x} \to \vec{y}) = \min\left\{1, \frac{\pi(\vec{y})\, p_K^i\, \delta(\vec{x} - \mu_{\alpha\beta}^{-1}(\vec{y}))}{\pi(\vec{x})\, p_K^j\, \delta(\vec{y} - \mu_{\alpha\beta}(\vec{x}))}\right\}. \tag{8.5}$$

According to the change of variable formula in the Dirac distribution $\delta(\vec{x} - \mu_{\alpha\beta}^{-1}(\vec{y})) = |\det J(\mu_{\alpha\beta}(\vec{x}))|\, \delta(\vec{y} - \mu_{\alpha\beta}(\vec{x}))$, then the acceptance probability can be simplified as follows:

$$p_{acc}(\vec{x} \to \vec{y}) = \min\left\{1, \frac{\pi(\vec{y})\, p_K^i}{\pi(\vec{x})\, p_K^j} |\det J(\mu_{\alpha\beta}(\vec{x}))|\right\}. \tag{8.6}$$

The bijective function $\mu_{\alpha\beta}(\cdot)$ is constructed with specific reversible neural networks as described in the following section.

8.2 Constructing bijective functions with deep neural networks

Reversible neural networks can be constructed using real-valued non-volume preserving (real NVP) transformations [75]. It has been proven that feed forward deep neural networks are capable of approximating any continuous function [76], but, in general, it is not possible to invert these functions. However, reversibility can still be attained if the architecture of the network is made of a sequence of trivially invertible operations as multiplication and addition. The core idea of real NVP networks is to divide the variables into two disjoint sets of coordinates $\vec{x} = (\vec{x}_1, \vec{x}_2)$, then only simple operations are performed on the defined sets. The variable transformation is thus defined as follows:

$$\begin{cases} \vec{z}_1 = \vec{x}_1, \\ \vec{z}_2 = x_2 \exp(S(\vec{x}_1; \theta)) + T(\vec{x}_1; \theta), \end{cases} \tag{8.7}$$

where S and T are feed forward neural networks, and θ are the

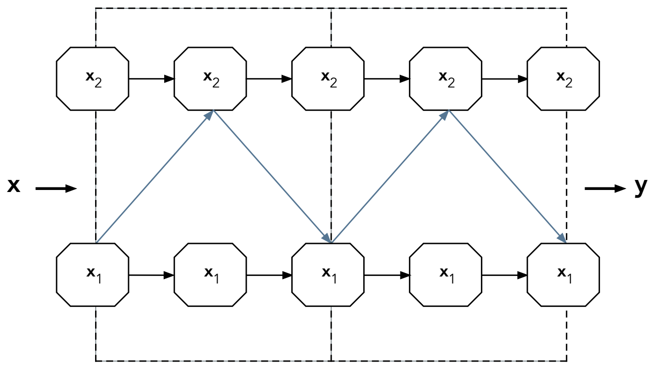


Figure 8.1: Real NVP networks are invertible transformations, where non linear operations are performed only on subsets of the system coordinates. The input vector $\vec{x}$ is separated into two disjoint sets of coordinates $\vec{x}_1, \vec{x}_2$, and at each iteration one subset is the input of feed forward networks (blue line in the figure); the output of the networks then modifies the other subset. The network is then invertible, that allows us to reconstruct the input $\vec{x}$ from the output $\vec{y}$. We define one real NVP block as the concatenation of two consecutive transformation on the two subsets. The network in the figure is constituted by two real NVP blocks, and the different blocks are evidenced with dashed lines.

trainable parameters of the network. It is then possible to reconstruct the input of the network from the output

$$\begin{cases} \vec{x}_1 = \vec{z}_1, \\ \vec{x}_2 = (\vec{z}_2 - T(\vec{z}_1;\theta)) \exp(-S(\vec{z}_1;\theta)). \end{cases} \tag{8.8}$$

Nonlinearities are inserted into the transformation through the neural networks S and T, but we note that these networks are defined only on one set of the variables. To insert correlations between the two sets of variables it is then necessary to serially concatenate many variable transformations, iteratively alternating the subsets where the feed forward networks are defined (see Fig. 8.1). The following concatenation of variable transformations is then defined as one real NVP block:

$$\begin{cases} \vec{z}_1 = \vec{x}_1, \\ \vec{z}_2 = \vec{x}_2 \exp(-S^{(m)}(\vec{x}_1;\theta)) + T^{(m)}(\vec{x}_1;\theta), \end{cases}$$
$$\begin{cases} \vec{y}_1 = \vec{z}_1 \exp(-S^{(n)}(\vec{z}_2;\theta)) + T^{(n)}(\vec{z}_2;\theta), \\ \vec{y}_2 = \vec{z}_2, \end{cases} \tag{8.9}$$

the block is also invertible

$$\begin{cases} \vec{z}_1 = (\vec{y}_1 - T^{(n)}(\vec{y}_2;\theta)) \exp(S^{(n)}(\vec{y}_2;\theta)), \\ \vec{z}_2 = \vec{y}_2, \end{cases}$$
$$\begin{cases} \vec{x}_1 = \vec{z}_1, \\ \vec{x}_2 = (\vec{z}_2 - T^{(m)}(\vec{z}_1;\theta)) \exp(S^{(m)}(\vec{z}_1;\theta)). \end{cases} \tag{8.10}$$

We notice that within one block non linear transformations are defined on both variable sets, but a concatenation of many of these blocks is actually required to effectively pass information between the two variable sets. Another interesting property of real NVP transformations is that the determinant of the Jacobian J associated to the variable transformation $\mu_{\alpha\beta}(\vec{x})$ is trivial to compute

$$|\det J(\mu_{\alpha\beta}(\vec{x}))| = \exp\left(\sum_{m,j} S^{(m)}(x_j)\right), \tag{8.11}$$

where the sum is iterated over all dimensions per each transformation. The determinant of the Jacobian is then interpretable as the change in entropy caused by the transformation. The entropy within the core Ω_α is defined as follows:

$$S_{\Omega_\alpha} = -K_B \int_{\Omega_\alpha} \pi(\vec{x}) \log\left(\pi(\vec{x})\right)\, d\vec{x}. \tag{8.12}$$

Considering the change of variable $\vec{y} = \mu(\vec{x})$:

$$\begin{aligned} S_{\Omega_\alpha} &= -K_B \int_{\Omega_\beta} \pi\left(\mu^{-1}(\vec{y})\right) \log\left(\pi(\mu^{-1}(\vec{y}))\, |\det J(\mu_{\alpha\beta}(\vec{x}))|\right)\, d\vec{y} = \\ &= S_{\Omega_\beta} - K_B \int_{\Omega_\beta} \log\left(|\det J(\mu_{\alpha\beta}(\vec{x}))|\right)\, d\vec{y} = \\ &= S_{\Omega_\beta} - E_{\Omega_\beta}(\log\left(|\det J(\mu_{\alpha\beta}(\vec{x}))|\right)). \end{aligned} \tag{8.13}$$

Hence, the entropy difference between the two cores is given by the expected value of the determinant of the Jacobian.

8.3 Deep neural networks training

Neural moves are generated with a bijective function that is constructed using real NVP transformations. Let us take the cores Ω_α, Ω_β, and we aim to construct the bijective function $\mu_{\alpha\beta}(\cdot)$ that is defined on the two cores. Networks are trained with equilibrium configurations of the system, and training sets $\{\vec{x}_n\}_{n=1}^{N_{set}}$, $\{\vec{y}_n\}_{n=1}^{N_{set}}$ are obtained running local MCMC simulations in the different cores $\vec{x}_n \in \Omega_\alpha$, $\vec{y}_n \in \Omega_\beta$, where N_{set} is the total number of configurations in the training set. Training of the network is performed so as to establish a bijection between the two cores $\vec{y} = \mu_{\alpha\beta}(\vec{x})$, $\vec{x} = \mu_{\alpha\beta}^{-1}(\vec{y})$, and this is achieved through a supervised training. Each minibatch randomly selects N_{mini} configurations from the two cores $\{\vec{x}_n\}_{n=1}^{N_{mini}}$, $\{\vec{y}_n\}_{n=1}^{N_{mini}}$, and in each minibatch the average configuration is computed $\langle\vec{x}\rangle = \sum_{n=1}^{N_{mini}} \vec{x}/N_{mini}$, $\langle\vec{y}\rangle = \sum_{n=1}^{N_{mini}} \vec{y}/N_{mini}$. Average configurations are then used as labels for the supervision. One term in the cost function is aimed to establish a bijection between the two cores α, β

$$C_{\alpha\beta} = \sum_{n=1}^{N_{mini}} \left\|\mu_{\alpha\beta}(\vec{x}_n) - \langle\vec{y}\rangle\right\|^2 .$$

Networks are trained to maximize the acceptance probability, that is achieved by maximizing the following term

$$\min\left\{1, \frac{\pi(\mu_{\alpha\beta}(\vec{x}))}{\pi(\vec{x})} \left|\det J(\mu_{\alpha\beta}(\vec{x}))\right|\right\}. \tag{8.14}$$

We then assume the system in the canonical ensemble

$$\min\left\{1, e^{-\beta\left(V(\mu_{\alpha\beta}(\vec{x}))-V(\vec{x})\right)} \left|\det J(\mu_{\alpha\beta}(\vec{x}))\right|\right\}, \tag{8.15}$$

being $V(\vec{x})$ the energy of the system and β the Boltzmann constant. The above term must be maximized for both directions, i.e. for states that belong to the core α and states that belong to the core β, that is achieved if the following equation is satisfied:

$$e^{-\beta\left(V(\mu_{\alpha\beta}(\vec{x}))-V(\vec{x})\right)} \left|\det J(\mu_{\alpha\beta}(\vec{x}))\right| = 1, \tag{8.16}$$

and taking the logarithm of both terms in the equation

$$-\beta\left(V(\mu_{\alpha\beta}(\vec{x})) - V(\vec{x})\right) + \log\left(\left|\det J(\mu_{\alpha\beta}(\vec{x}))\right|\right) = 0. \tag{8.17}$$

The expected value of the above equation is physically interpretable as the free energy difference caused by the transformation (see Eq. (8.13)) for a derivation of the entropic term)

$$\Delta V + T\,\Delta S = 0. \tag{8.18}$$

The cost function then results as it follows:

$$C = \gamma_0\, C_{\alpha\beta} + \gamma_1\, \| V(\mu_{\alpha\beta}(\vec{x})) - V(\vec{x}) \|^{\,2} + \gamma_2\, \| S(\mu_{\alpha\beta}(\vec{x})) - S(\vec{x}) \|^2 ,$$

where γ_0, γ_1, γ_2 are hyper-parameters. For simplicity, the cost function has been here defined only for states $\vec{x} \in C_\alpha$ and relative image $\mu_{\alpha\beta}(\vec{x})$. The same cost function is also applied to states $\vec{y} \in C_\beta$, whose image is $\mu^{-1}_{\alpha\beta}(\vec{y})$, and each minibatch is set to contain the same number of states from the two different cores. Finally, note that, to facilitate the establishment of the bijection between the two cores, training is initialized with the absolute value of the free energy terms in the cost function, rather than the square.

8.4 Results

Neural MJMC is tested in a crowded condensed matter system composed of strongly repelling particles confined in a two-dimensional box. A bistable particle dimer is immersed in a fluid, where the dimer is composed of a pair of particles interacting with a potential with minima in closed or open configurations, that are separated by a high energy barrier. Opening and closing of the dimer requires a concerted motion of the 36 solvent particles, that makes it difficult to sample the physical path connecting the two configurations (see Ref. [72] for a more detailed description of the system). An efficient MCMC sampler requires in this case the generation of neural moves between

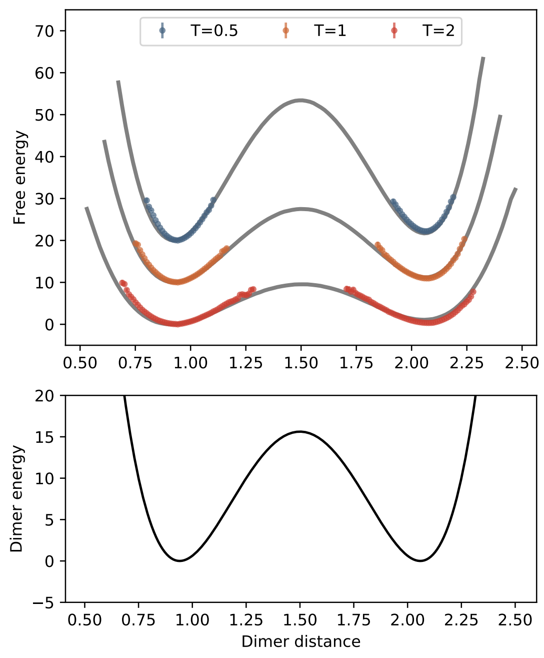


Figure 8.2: Free energy and dimer potential. The grey bands represent values sampled with umbrella samplings within uncertainty. The neural network has been trained at temperature $T = 1$, then simulations at different temperatures have been performed using neural MJMC. Simulations are run for 1000 steps, and error bars are produced using bootstrapping. The average acceptance probability of neural moves is about 21%. In this figure, we can see that after short simulations the right distribution is already reconstructed. On the other hand, the estimated time to cross the energy barrier with local MCMC moves is 10^{12} steps at temperature $T = 1$, which makes the observation of spontaneous crossing with local moves computationally unfeasible. We also emphasize that, in contrast to umbrella sampling that requires new iterations per each different temperature, neural MJMC is not bound to the temperature used during training, and the same network can be applied also to other temperatures.

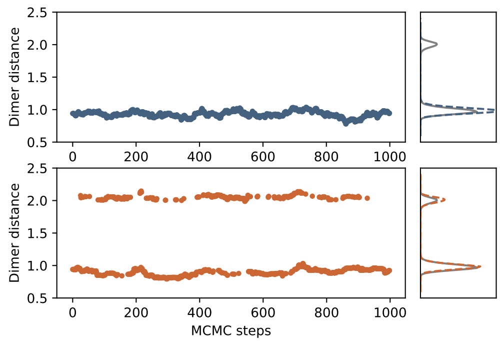


Figure 8.3: Dimer distance over a single realization using local MCMC (top), and using neural MJMC (bottom). Right) Histogram of the dimer distance averaged over many realizations. It is possible to see that using local moves it is unlikely to observe spontaneous transitions at this time scale, while using neural MJMC both metastable states are explored in each trajectory.

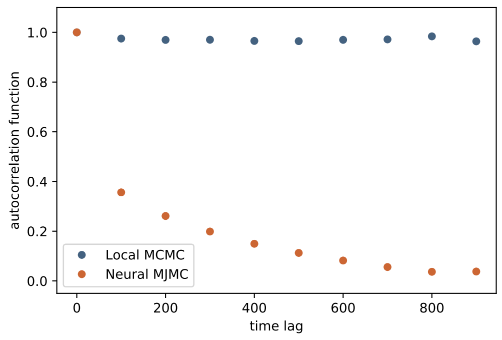


Figure 8.4: Autocorrelation of the dimer distance. Neural moves allow for a fast exploration of both metastable states, accelerating the production of uncorrelated samples. In this figure, it is evident that neural MJMC frequently generates uncorrelated samples, and short trajectories are sufficient to reconstruct the right distribution. In contrast, configurations generated with local MCMC are highly correlated.

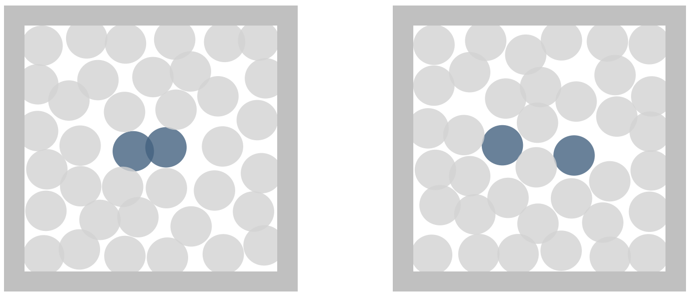


Figure 8.5: Example of states in the closed (left) and open (right) dimer configuration generated with neural moves in neural MJMC. The dimer particles are evidenced in blue, while the solvent particles are grey. The strongly repulsive potential does not allow for significant overlaps between particles at equilibrium. Designing manually moves that avoid particle overlaps when jumping between the open and closed configuration is practically impossible, but this is achievable using machine learning.

the two metastable states, but designing manually a bijective function connecting equilibrium states of the system in the open and closed configurations appear unfeasible due to the large number of possible arrangements of the solvent particles.

Neural displacements are generated with deep neural networks that are trained on equilibrium states of the system, which are collected in the open and closed configurations of the dimer. The system is invariant under permutation of the solvent particles, and permutation invariance must be incorporated to facilitate the training. In fact, if each possible particle exchange were considered a new configuration, the complexity of the problem would scale factorially with the number of particles. The identical permutations of the system are then removed by appropriately relabeling the particles so as to minimize the distance to a reference configuration, that is realized using the Hungarian algorithm [77].

DNNs training is performed with a real NVP network that is composed of 50 blocks. Each feed-forward network contains 38 parameters, which makes approximately 6×10^6 parameters in total. Neural MJMC is then used to generate trajectories, all starting with equal initial conditions in the closed dimer configuration. Neural moves cause direct transitions between the two metastable states and thus a rapid exploration of the configuration space, that facilitates converge to the right distribution, as shown in Fig. 8.2. We also notice that the network is transferable to different temperatures, and temperature is just a parameter that is adjusted to the desired value after training. Crossing the energy barrier is unlikely at this time scale with local displacements (see Fig. 8.3), and exhausting simulations would be necessary before ergodicity is verified. Samples generated with local MCMC are indeed highly correlated, while the autocorrelation function decays quickly in neural MJMC simulations (see Fig. 8.4). In Fig. 8.5, we see equilibrium configurations generated with neural moves in neural MJMC. Particles strongly repulse with each other, and it is unlikely to observe significant particle overlaps at equilibrium. An analytic construction of a bijective function that pairs equilibrium configurations from both metastable states is impossible due to the difficulty of avoiding particle overlaps, which is instead possible with the aid of deep neural networks.

Chapter 9

Conclusion and Outlook

In this thesis, several numerical methods to efficiently simulate biological and condensed matter systems have been introduced. The first part of the thesis deals with reaction-diffusion algorithms that are used to simulate systems at the mesoscopic scale. These algorithms are based on the assumption that mesoscopic particles diffuse according to the Langevin equation, and when they are close to each other can react. However, particle encounters can happen rarely in dilute systems, that makes it difficult to study the kinetics of the system over long time scales. First passage algorithms circumvent this problem by avoiding full resolution of free diffusion, using an event-based scheme that directly calculates the time and position of particle encounters. MD-GFRD is a recent first passage algorithm that alternates first passage propagations within protective domains and particle interactions. This algorithm has been shown to be many orders of magnitude more efficient than brute-force approaches. We have seen that the scheme used to determine the size of protective domains is relevant for performance, and a novel scheme that enhances MD-GFRD providing an additional tenfold improvement in computational efficiency has been presented. Still, performance is susceptible to the time required to sample particle displacements within protective domains, and an efficient method to sample those displacements has been discussed. Despite their popularity and wide range of applications, a rigorous formalization of first passage schemes was missing. Following the original formalization, applications of these schemes might lead to conceptual inconsistencies and, in some specific conditions, to misleading results. In this thesis, first passage schemes have been reformalized so as to avoid conceptual contradictions, thus placing first passage algorithms on firm conceptual ground. MD-GFRD, being based on an event list, is intrinsically sequential, but it would be desirable to deploy the computational power provided by the most recent graphical processing units (GPUs), which is possible only if this algorithm is parallelized. A future improvement could then be the development of an efficient parallel implementation of this algorithm. Combining the efficiency of MD-GFRD and the computational capacity of GPUs would produce a scheme incredibly efficient, opening up the doors to simulations of enormous complexity, as, for instance, the simulation of a whole cell.

The second part of the thesis introduces a novel Markov chain Monte Carlo (MCMC) scheme that makes use of deep learning to accelerate convergence to equilibrium. MCMC methods sample the microscopic equilibrium distribution relative to a stochastic system, thus allowing to compute macroscopic observables as averages over microscopic configurations of the system. This method has been shown to asymptotically converge to the right distribution if the conditions of detailed balance and ergodicity are satisfied. The condition of detailed balance ensures that Markov chains remain within regions of the system that are likely at equilibrium, while ergodicity ensures that all likely regions in the system are visited. However, many interesting systems are composed of basins of equilibrium states that are mutually connected with out-of-equilibrium paths, which are arduous to sample. Many different methodologies have been developed during the last years, but they all either depend on out-of-equilibrium paths or are difficult to apply to multi-dimensional systems. However, we have seen that with the aid of neural networks it is possible to generate displacements directly connecting equilibrium basins. These "neural" moves are alternated with local random displacement in a combined scheme that we call neural mode jump Monte Carlo (neural MJMC). Neural MJMC is theoretically consistent and it is shown to satisfy detailed balance and ergodicity, which ensures that it asymptotically samples the equilibrium distribution. This method has been tested in a non-trivial multi-dimensional particle system, and it has been shown that convergence is rapidly reached. The neural networks employed in this work are a special class of reversible networks (real NVP), which are also used in the field of generative probabilistic modeling. Considering the great attention this field is lately facing, and the continuous technological advances in machine learning, it would not be surprising to witness dramatic improvements in the performance of such reversible networks. Efficiency and perspectives of neural MJMC profoundly rely on the specific network employed, and more sophisticated networks would allow us to deal with systems of increasing complexity. Neural MJMC is a generable and transferable method, its range of applications goes beyond biology, and we can expect it to be applied to a wealth of different systems in the next few years.

Acknowledgement

I am grateful to many people for the help and support that I received during the years of my PhD.

I would like to thank Frank for giving me the freedom to follow my intuition, providing me with exceptionally interesting topics, and guiding me through them. I thank Luigi, firstly, for assisting me in the theoretical development of one project, and also for the many interesting scientific discussions we had. We share many common interests, and I am glad that I have had the opportunity to work with him. I am grateful to Giovanni Ciccotti, because I believe that the attention to detail that I have learned from him has been very useful throughout my PhD. During these years, I have been collaborating and interacting with all members of the CMB group. In this group, I had the pleasure to constantly learn from many brilliant persons. I have also interacted with many other members of the Mathematics Department and Zuse Institute in a fantastic atmosphere, that was tangible, for example, during the Winterseminars. I would like to thank Christof Schütte for creating such an exciting atmosphere, and also Marcus Weber for many inspiring discussions. The support that I have received from Kirsten and Katja is unmeasurable, and I feel lucky to have met these exceptional persons.

I thank all my friends for the great time I had during these years. Firstly, the friends, not just colleagues, in the CMB group. It is hard to find a research group composed of so many people with outstanding scientific and social skills as in the CMB group. My friends from Rome and the ones spread over Europe supported me in many occasions, and I am grateful for that. Among them, I shared the beginning of this journey with Daniele. We share similar passions, and our long discussions have fueled the development of my scientific work. Similarly, I am grateful to Angelo, the very first friend I had with whom I shared this passion for science. I also thank Matteo for having visited me in many occasions in Berlin. Lastly, thank you, Katharina, for the unforgettable time together during most part of our PhDs.

I would like to thank the Max Planck Institute, the European Research Commission, the Zuse Institute Berlin, and the Deutsche Forschungsgemeinschaft for the financial support that I received.

Finally, I thank my father and my mother, and all my family. They brought me here, without their support this would have not been possible.

Bibliography

[1] Jie Xiao and Yves F. Dufrêne. Optical and force nanoscopy in microbiology. *Nature Microbiology*, 1:16186 EP –, 10 2016.

[2] Philip R Nicovich, Dylan M Owen, and Katharina Gaus. Turning single-molecule localization microscopy into a quantitative bioanalytical tool. *Nature Protocols*, 12:453 EP –, 02 2017.

[3] Steffen J. Sahl, Stefan W. Hell, and Stefan Jakobs. Fluorescence nanoscopy in cell biology. *Nature Reviews Molecular Cell Biology*, 18:685 EP –, 09 2017.

[4] Giuseppe Vicidomini, Paolo Bianchini, and Alberto Diaspro. Sted super-resolved microscopy. *Nature Methods*, 15:173 EP –, 01 2018.

[5] Vsevolod V. Gurevich and Eugenia V. Gurevich. How and why do gpcrs dimerize? *Trends in Pharmacological Sciences*, 29(5):234 – 240, 2008.

[6] M. Gunkel, J. Schöneberg, W. Alkhaldi, S. Irsen, F. Noé, U.B. Kaupp, and A. Al-Amoudi. Higher-order architecture of rhodopsin in intact photoreceptors and its implication for phototransduction kinetics. *Structure*, 23:628–638, 2015.

[7] Jun Xing, David D. Ginty, and Michael E. Greenberg. Coupling of the ras-mapk pathway to gene activation by rsk2, a growth factor-regulated creb kinase. *Science*, 273(5277):959–963, 1996.

[8] Koichi Takahashia, Sorin Tănase-Nicolad, and Peter Rein ten Wolde. Spatio-temporal correlations can drastically change the response of a mapk pathway. *Proceedings of the National Academy of Sciences*, 107(6):2473– 2478, 2009.

[9] B. Trzaskowski, D. Latek, S. Yuan, U. Ghoshdastider, A. Debinski, and S. Filipek. Action of molecular switches in gpcrs - theoretical and experimental studies. *Current Medicinal Chemistry*, 19(8):1090–1109, 2012.

[10] Vladimir J. Kefalov. Rod and cone visual pigments and phototransduction through pharmacological, genetic, and physiological approaches. *Journal of Biological Chemistry*, 287(3):1635–1641, 2012.

[11] Volker Haucke, Erwin Neher, and Stephan J. Sigrist. Protein scaffolds in the coupling of synaptic exocytosis and endocytosis. *Nature Reviews Neuroscience*, 12:127 EP –, 02 2011.

[12] Jeroen S. van Zon and Pieter Rein ten Wolde. Simulating biochemical networks at the particle level and in time and space: Green's function reaction dynamics. *Phys. Rev. Lett.*, 94:128103, Apr 2005.

[13] Tomas Opplestrup, Vasily V. Bulatov, George H, Gilmer, Malvin H. Kalos, and Babak Sadigh. First-passage monte carlo algorithm: Diffusion without all the hops. *Phys. Rev. Lett.*, 97:230602, 2006.

[14] J. Schöneberg and F. Noé. Readdy - a software for particle based reaction diffusion dynamics in crowded cellular environments. *PLoS ONE*, 8(e74261), 2013.

[15] D. T. Gillespie. A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. *J. Comput. Phys.*, 22:403, 1976.

[16] Stefanie Winkelmann and Christof Schütte. The spatiotemporal master equation: Approximation of reaction-diffusion dynamics via markov state modeling. *J. Chem. Phys.*, 145:214107, 2016.

[17] J. Schöneberg, M. Heck, K.-P. Hofmann, and F. Noé. Explicit spatio-temporal simulation of receptor-g protein coupling in rod cell disk membranes. *Biophys. J.*, 107:1042–1053, 2014.

[18] Radek Erban and S. Jonathan Chapman. Stochastic modelling of reaction-diffusion processes: algorithms for bimolecular reactions. *Physical Biology*, 6(4):046001, 2009.

[19] Jeroen S. van Zon and Pieter Rein ten Wolde. Green's-function reaction dynamics: A particle-based approach for simulating biochemical networks in time and space. *The Journal of Chemical Physics*, 123(23):234910, 2019/07/05 2005.

[20] Johannes Schöneberg, Martin Lehmann, Alexander Ullrich, York Posor, Wen-Ting Lo, Gregor Lichtner, Jan Schmoranzer, Volker Haucke, and Frank Noé. Lipid-mediated px-bar domain recruitment couples local membrane constriction to endocytic vesicle fission. *Nat. Commun.*, 8:15873, 2017.

[21] J. Schöneberg, A. Ullrich, and F. Noé. Simulation tools for particle-based reaction-diffusion dynamics in continuous space. *BMC Biophysics*, 7:11, 2014.

[22] Steven S. Andrews. *Particle-Based Stochastic Simulators*, pages 1–5. Springer New York, New York, NY, 2018.

[23] Moritz Hoffmann, Christoph Fröhner, and Frank Noé. Readdy 2: Fast and flexible software framework for interacting-particle reaction dynamics. *PLOS Computational Biology*, 15(2):1–26, 02 2019.

[24] Steven S Andrews. Smoldyn: particle-based simulation with rule-based modeling, improved molecular interaction and a library interface. *Bioinformatics*, 33(5):710–717, 12 2016.

[25] R. Kerr, T. Bartol, B. Kaminsky, M. Dittrich, J. Chang, S. Baden, T. Sejnowski, and J. Stiles. Fast monte carlo simulation methods for biological reaction-diffusion systems in solution and on surfaces. *SIAM Journal on Scientific Computing*, 30(6):3126–3149, 2008.

[26] Chris Sanford, Matthew L.K. Yip, Carl White, and John Parkinson. Cell++?simulating biochemical pathways. *Bioinformatics*, 22(23):2918–2925, 10 2006.

[27] Johan Hattne, David Fange, and Johan Elf. Stochastic reaction-diffusion simulation with MesoRD. *Bioinformatics*, 21(12):2923–2924, 04 2005.

[28] Satya Nanda Vel Arjunan and Masaru Tomita. A new multi-compartmental reaction-diffusion modeling method links transient membrane attachment of e. coli mine to e-ring formation. *Systems and Synthetic Biology*, 4(1):35–53, Mar 2010.

[29] Paul J. Michalski and Leslie M. Loew. Springsalad: A spatial, particle-based biochemical simulation platform with excluded volume. *Biophysical Journal*, 110(3):523 – 529, 2016.

[30] Gerd Gruenert, Bashar Ibrahim, Thorsten Lenser, Maiko Lohel, Thomas Hinze, and Peter Dittrich. Rule-based spatial modeling with diffusing, geometrically constrained molecules. *BMC Bioinformatics*, 11(1):307, 2010.

[31] A. Vijaykumar, P.G. Bolhuis, and P.R. ten Wolde. Combining molecular dynamics with mesoscopic green's function reaction dynamics simulations. *J. Chem. Phys.*, 143:214102, 2015.

[32] Masao Doi. Theory of diffusion-controlled reaction between non-simple molecules. i. *Chemical Physics*, 11(1):107 – 113, 1975.

[33] A. Vijaykumar, T.E. Ouldridge, P.R. ten Wolde, and P.G. Bolhuis. Multiscale simulations of anisotropic particles combining molecular dynamics and green's function reaction dynamics. *J. Chem. Phys.*, page 114106.

[34] Jakob Schluttig, Christian B. Korn, and Ulrich S. Schwarz. Role of anisotropy for protein-protein encounter. *Phys. Rev. E*, 81:030902, Mar 2010.

[35] Jan-Hendrik Prinz, Hao Wu, Marco Sarich, Bettina Keller, Martin Senne, Martin Held, John D. Chodera, Christof Schütte, and Frank Noé. Markov models of molecular kinetics: Generation and validation. *The Journal of Chemical Physics*, 134(17):174105, 2011.

[36] Gregory R. Bowman, Vijay Pande, and Frank Noé. *An Introduction to Markov State Models and Their Application to Long Timescale Molecular Simulation*, volume 797. 01 2014.

[37] Ch. Schütte and M. Sarich. *Metastability and Markov State Models in Molecular Dynamics: Modeling, Analysis, Algorithmic Approaches.* American Mathematical Society, 2013.

[38] Nuria Plattner, Stefan Doerr, Gianni De Fabritiis, and Frank Noé. Complete protein-protein association kinetics in atomic detail revealed by molecular dynamics simulations and markov modelling. *Nat. Chem.*, 9:1005–1011, 2017.

[39] Manuel Dibak, Mauricio J. del Razo, David De Sancho, Christof Schütte, and Frank Noé. Msm/rd: Coupling markov state models of molecular kinetics with reaction-diffusion simulations. *Journal of Chemical Physics*, 148(214107), 2018.

[40] Hao Wu, Antonia S. J. S. Mey, Edina Rosta, and Frank Noé. Statistically optimal analysis of state-discretized trajectory data from multiple thermodynamic states. *The Journal of Chemical Physics*, 141(21):214106, 2014.

[41] Hao Wu, Fabian Paul, Christoph Wehmeyer, and Frank Noé. Multiensemble markov models of molecular thermodynamics and kinetics. *Proceedings of the National Academy of Sciences*, 113(23):E3221–E3230, 2016.

[42] Jack Dongarra and Francis Sullivan. Top ten algorithms of the century. *Computing in Science and Engineering - C in SE*, 01 2000.

[43] Daan Frenkel and Berend Smit. Chapter 3 - monte carlo simulations. In Daan Frenkel and Berend Smit, editors, *Understanding Molecular Simulation (Second Edition)*, pages 23 – 61. Academic Press, San Diego, second edition edition, 2002.

[44] R Kubo. The fluctuation-dissipation theorem. *Reports on Progress in Physics*, 29(1):255–284, 1966.

[45] Anja Nenninger, Giulia Mastroianni, and Conrad W. Mullineaux. Size dependence of protein diffusion in the cytoplasm of escherichia coli. *Journal of Bacteriology*, 192(18):4535–4540, 2010.

[46] A. Einstein. *Annalen der Physik*, 322(8):549–560, 1905.

[47] C. W. Gardiner. *Handbook of stochastic methods for physics, chemistry and the natural sciences*, volume 13 of *Springer Series in Synergetics*. Springer-Verlag, Berlin, third edition, 2004.

[48] S. Redner. *A Guide to First-Passage processes*. Cambridge University Press, 2001.

[49] T.R. Sokolowski. *pp. 48-49.* PhD thesis, 2013.

[50] A. Donev, V. V. Bulatov, T. Oppelstrup, G. H. Gilmer, B. Sadigh, and M. H. Kalos. A first-passage kinetic monte carlo algorithm for complex diffusion-reaction systems. *J. Comp. Phys.*, 229:3214–3236, 2010.

[51] M. H. Kalos, D. Levesque, and L. Verlet. Helium at zero temperature with hard-sphere and other forces. *Phys. Rev. A*, 9:2178–2195, May 1974.

[52] Martijn Wehrens, Pieter Rein ten Wolde, and Andrew Mugler. Positive feedback can lead to dynamic nanometer-scale clustering on cell membranes. *J. Chem. Phys.*, 141(20):205102, 2014.

[53] T. Oppelstrup, V. V. Bulatov, A. Donev, M. H. Kalos, G. H. Gilmer, and B. Sadigh. First-passage kinetic monte carlo method. *Phys. Rev. E*, 80:066701, 2009.

[54] V. I. Tokar and H. Dreyssé. Accelerated kinetic monte carlo algorithm for diffusion-limited kinetics. *Phys. Rev. E*, 77:066705, Jun 2008.

[55] Ava J. Mauro, Jon Karl Sigurdsson, Justin Shrake, Paul J. Atzberger, and Samuel A. Isaacson. A first-passage kinetic monte carlo method for reaction–drift–diffusion processes. *J. Comp. Phys.*, 259(Supplement C):536 – 567, 2014.

[56] Andri Bezzola, Benjamin B. Bales, Richard C. Alkire, and Linda R. Petzold. An exact and efficient first passage time algorithm for reaction–diffusion processes on a 2d-lattice. *J. Comp. Phys.*, 256(Supplement C):183 – 197, 2014.

[57] Karsten Schwarz and Heiko Rieger. Efficient kinetic monte carlo method for reaction–diffusion problems with spatially varying

annihilation rates. *J. Comp. Phys.*, 237(Supplement C):396 – 410, 2013.

[58] Thomas R. Sokolowski, Joris Paijmans, Laurens Bossen, Martijn Wehrens, Thomas Miedema, Nils B. Becker, Kazunari Kaizu, Koichi Takahashi, Marlieen Dogterom, and Pieter Rein ten Wolde. egfrd in all dimensions. *arXiv:1708.09364*, 2017.

[59] Joris Paijmans and Pieter Rein ten Wolde. Lower bound on the precision of transcriptional regulation and why facilitated diffusion can reduce noise in gene expression. *Phys. Rev. E*, 90:032708, Sep 2014.

[60] Aleksandar Donev. Asynchronous event-driven particle algorithms. *SIMULATION*, 85(4):229–242, 2019/02/01 2009.

[61] Uttam Bhat and S Redner. Intermediate-level crossings of a first-passage path intermediate-level crossings of a first-passage path intermediate-level crossings of a first-passage pathintermediate-level crossings of a first-passage path intermediate-level crossing of a first passage path. *J. Stat. Mech. Theory Exp.*, 2015(6):P06035, 2015.

[62] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, and E. Teller. Equation of State Calculations by Fast Computing Machines. *The Journal of Chemical Physics*, 21:1087–1092, June 1953.

[63] W. K. Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.

[64] W. Feller. *An Introduction to Probability Theory and Its Applications*, volume 1. Wiley, January 1968.

[65] E Marinari and G Parisi. Simulated tempering: A new monte carlo scheme. *Europhysics Letters (EPL)*, 19(6):451–458, jul 1992.

[66] Harry A. Stern. Molecular simulation with variable protonation states at constant ph. *The Journal of Chemical Physics*, 126(16):164112, 2007.

[67] Jerome P. Nilmeier, Gavin E. Crooks, David D. L. Minh, and John D. Chodera. Nonequilibrium candidate monte carlo is an efficient tool for equilibrium simulation. *Proceedings of the National Academy of Sciences*, 108(45):E1009–E1018, 2011.

[68] Yunjie Chen and Benoît Roux. Constant-ph hybrid nonequilibrium molecular dynamics monte carlo simulation method. *Journal of Chemical Theory and Computation*, 11(8):3919–3931, 2015. PMID: 26300709.

[69] Ioan Andricioaei, John E. Straub, and Arthur F. Voter. Smart darting monte carlo. *The Journal of Chemical Physics*, 114(16):6994–7000, 2001.

[70] K. Roberts, R. Sebsebie, and E. Curotto. A rare event sampling method for diffusion monte carlo using smart darting. *The Journal of Chemical Physics*, 136(7):074104, 2012.

[71] Lionel Walter and Marcus Weber. Confjump : a fast biomolecular sampling method which drills tunnels through high mountains. Technical Report 06-26, ZIB, Takustr. 7, 14195 Berlin, 2006.

[72] Frank Noé, Simon Olsson, Jonas Köhler, and Hao Wu. Boltzmann generators: Sampling equilibrium states of many-body systems with deep learning. *Science*, 365(6457), 2019.

[73] Huitao Shen, Junwei Liu, and Liang Fu. Self-learning monte carlo with deep neural networks. *Phys. Rev. B*, 97:205140, May 2018.

[74] Daniel Levy, Matt D. Hoffman, and Jascha Sohl-Dickstein. Generalizing hamiltonian monte carlo with neural networks. In *International Conference on Learning Representations*, 2018.

[75] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. 2017.

[76] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2(4):303–314, Dec 1989.

[77] H. W. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(12):83–97, 1955.

Selbstständigkeitserklärung

Hiermit versichere ich, dass ich die vorliegende Arbeit selbstständig verfasst habe und alle dabei verwendeten Hilfsmittel und Hilfen angegeben habe. Die Arbeit ist nicht schon einmal in einem früheren Promotionsverfahren eingereicht worden.

Berlin, September 2019, ...