MESOSCALE SIMULATION OF LIPID BILAYERS: QUASI-2D HYDRODYNAMICS AND SHEAR VISCOSITY

by

Andrew Zgorski

A dissertation submitted to the Faculty of the University of Delaware in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Physics

Summer 2018

MESOSCALE SIMULATION OF LIPID BILAYERS: QUASI-2D HYDRODYNAMICS AND SHEAR VISCOSITY

by

Andrew Zgorski

Approved: ____
Edmund Nowak, Ph.D.
Chair of the Department of Physics and Astronomy

Approved: ____
George Watson, Ph.D.
Dean of the College of Arts & Sciences

Approved: ____
Douglas Doren, Ph.D.
Interim Vice Provost for Graduate and Professional Education

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed: ______________________________

Edward Lyman, Ph.D.
Professor in charge of dissertation

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed: ______________________________

Stephen Barr, Ph.D.
Member of dissertation committee

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed: ______________________________

Eric Furst, Ph.D.
Member of dissertation committee

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed: ______________________________

Sandeep Patel, Ph.D.
Member of dissertation committee

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed: ____

Krzysztof Szalewicz, Ph.D.
Member of dissertation committee

ACKNOWLEDGMENTS

Edward Lyman, Ph.D. for his indispensable guidance, thoughtful advice, and academic support over the last six years.

Richard Pastor, Ph.D. for his helpful collaboration, professional mentorship, and financial sponsorship through an NIH IRTA predoctoral fellowship.

Bernard Brooks, Ph.D. for thought-provoking discussions facilitating the development of the shearing protocol described in Chapter 7 and his insightful diagnosis of an integration bug in the original implementation of STRD Martini.

Members of the Lyman research group, past and present, whose conversations, feedback, and fellowship enabled the development of this work and cultivated a better understanding of membrane simulation and biology.

My family and friends, who have tirelessly supported and encouraged me on my academic journey.

This work was supported by a National Institutes of Health grant NIH P20GM104316. Computations were performed on the University of Delaware's community cluster *Mills*, the departmental *Greene* cluster, the *Naja* cluster (supported by the NIH IDeA program through NIH P20GM104316), and on *Stampede* supercomputer at the Texas

Advanced Computing Center of the University of Texas at Austin (supported through the XSEDE program of the NSF).

Certain figures and passages throughout the text are adapted from the journal article “Toward Hydrodynamics with Solvent Free Lipid Models: STRD Martini” published by the author in the Biophysical Journal [1].

Certain figures in this work were created using the free Visual Molecular Dynamics (VMD) software package [2].

TABLE OF CONTENTS

Appendix

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

The computational capabilities of molecular dynamics (MD) simulations have greatly advanced in recent years, allowing for the modeling of ever more complex systems. In the field of membrane simulation, this has facilitated studies of both large, heterogeneous systems and dynamics on millisecond time scales. Concurrently, innovations in experimental technique have allowed for probing dynamics on length and time scales approaching those in simulation. As these efforts continue to progress, future extensions will allow for direct comparison between experiment and simulation, enabling further refinement to both.

Membranes are quasi-2D viscous fluids which require accurate modeling of hydrodynamic transport to fully capture their dynamics. Relevant hydrodynamic theory predicts long-range coupling among proteins diffusing laterally in the membrane. In MD simulation, these long-distance interactions lead to self-interaction through the periodic image lattice and other finite size effects which may only be reduced by increasing the system size. Consequently, accurate modeling of bulk hydrodynamic transport using traditional MD (i.e. with explicit solvent particles) is not feasible. Calculating pairwise forces between the solvent particles demands an overwhelming majority of the available computational resources at the requisite system sizes. This predicament constitutes an unmet scientific need as novel algorithms and software implementations are required for accurate and efficient modeling of hydrodynamic interactions at scale.

We have met that need by supplementing an implicit-solvent lipid model called Dry Martini with an efficient mesoscopic hydrodynamics model called multi-particle collision (MPC) dynamics. Our hybrid model, called STRD Martini, is implemented in the popular open-source MD software package GROMACS v5.0.1, opening the way to further studies of membrane dynamics with proper accounting for hydrodynamic interactions. The selection of MPC dynamics for the mesoscopic solvent model was motivated by its particle-based nature, which cleanly interfaces with existing GROMACS code. As such, GROMACS may treat MPC particles just as any other particle for the purposes of integration, parallelization, trajectory writing, analysis, and force calculation (when desired). When combined with domain decomposition, STRD Martini scales to thousands of processors, providing accurate hydrodynamics while running at least an order of magnitude faster than equivalent explicit-solvent simulations.

The theory for membrane hydrodynamics in periodic geometries, called periodic Saffman-Delbrück theory, requires three parameters, two of which may be measured independently and a third which is a true fit parameter of the model. The independent parameters characterize the membrane surface viscosity and coefficient of friction between membrane leaflets. These parameters are not commonly calculated from simulation and remain uncharacterized for most popular membrane force fields. Following the blueprint of an earlier work, we further develop a protocol for conducting nonequilibrium shearing simulations to measure these parameters and apply the protocol to both coarse-grain and all-atom membranes.

Chapter 1

INTRODUCTION

1.1 Introduction to Lipid Membranes

The study of biology and soft matter is replete with examples of complex molecular systems exhibiting interesting physics. One fascinating example are plasma membranes; the semi-permeable, self-assembling fluid sheets which enclose cells and yet allow them to interact with their environment [3]. The plasma membrane hosts thousands of different proteins, the molecular machinery responsible for cellular activity. About one-third of the proteins in the human proteome are integral membrane proteins [4], and well over half of pharmaceutical drugs target these proteins or the membrane itself to trigger or disrupt important cellular functions [5]. The organization and dynamic behavior of these proteins, and hence their response to treatment, is governed by the physical environment provided by the plasma membrane. In addition, viral infection requires crossing the membrane via the process of endocytosis; following replication, viral exit requires passing back through the membrane via exocytosis [3]. Hence, a nuanced understanding of the structure and dynamics of cell membranes and the transport of proteins embedded within them is of great interest to medical science independent of the interesting questions they pose for soft matter physicists.

Plasma membranes are composed of amphipathic molecules called lipids, which join one or more hydrophobic fatty acid tails with hydrophilic, polar heads [6]. In the presence of water, this difference in hydrophobicity causes these molecules to spontaneously form double sheets with the heads on the exterior and the tails shielded within. Each sheet of the membrane is called a leaflet; together they form the phospholipid bilayer. These membranes are roughly 4 nm thick but extend over micrometer length scales. The lipids in each leaflet are free to move about, creating a two-dimensional viscous fluid. Momentum transport in the membrane is closely coupled to hydrodynamic flows in the surrounding water, creating a hybrid system which exhibits characteristics of both two- and three-dimensional hydrodynamics at different length scales [7].

Over sufficiently long length scales, membranes can be accurately modeled as continuous elastic sheets with elastic moduli describing the energetic penalty to stretch or to bend them [8]. The required bending energy is low enough for thermal fluctuations to spontaneously excite undulations of the membrane surface. Cells (and invading virions) are able to exploit this flexibility to remodel the membrane in a variety of ways, creating pits or folds in the surface or budding off a region of the membrane and its contents entirely. The plasma membrane is also connected at various points to an internal fiber network called the cytoskeleton, which provides rigidity and structure to the cell. In addition to the plasma membrane, cells use other lipid membranes to provide internal structure and to compartmentalize important cellular functions into organelles [3].

Many different proteins associate with membranes. Some only attach to their membrane on a temporary basis, while others are permanently anchored. The latter are called integral membrane proteins and are “transmembrane,” i.e., they span both leaflets and often include domains located inside (intracellular) and outside the membrane (extracellular). These proteins serve a variety of important cellular functions; including proteins which induce or facilitate membrane curvature, channels that allow ions or other small molecules to pass through the membrane, and receptors which react to the presence of particular molecules outside the cell. Over half of all FDA-approved drugs target these transmembrane proteins, with the most common class being a group of receptors coupled to intracellular “G-proteins” [9].

There are also a wide variety of different lipids found in the membranes of living cells — ca. 800 different combinations of acyl chain, backbone, and headgroup chemistry [10], but about one-third by mole of the lipid component in eukaryotic membranes is cholesterol. Depending on their mutual interactions, these complex mixtures of lipids and proteins might form localized regions of distinct composition in the cell membrane. The notion that these domains may serve a functional role as platforms for signaling is known as the “Raft Hypothesis” [11]. When divorced from the complex, nonequilibrium environment of the cell, model membrane mixtures can form coexisting fluid phases, called “liquid ordered” and “liquid disordered” [12]. Both are fluid, but they differ in composition, and as a result, in the extent to which the acyl chains order. Such model systems have long served as stand-ins for real cell membranes in both experiment and simulation.

1.2 Lipid Types

Four lipids are frequently referenced throughout this document using their abbreviated names: DPPC, DOPC, POPC, and PSM. Full chemical names for these lipids are given in **Table 1.1** and their chemical structures are illustrated in **Fig. 1.1**. The first three differ only in their tails; they share the same phosphatidylcholine (PC) headgroups connected to the glycerol backbone. Their tails are all either palmitoyl or oleyl, with the latter containing a single unsaturation about halfway along the acyl chain and two additional carbons. DP- has two palmitoyl, DO- has two oleyl, and PO- has one of each. As is often the case in chemistry, this minor difference has a profound impact on their collective dynamics. The unsaturation causes a “kink” in its chain, which causes oleyl tails to be more disordered, leading higher membrane viscosity, slower diffusion, and lower melting temperature. In contrast, the orderly palmitoyl tails tend to line up more readily, leading to higher melting temperature.

Table 1.1: Lipids referenced throughout this document (melting temperatures given by DPPC [13], DOPC [14], POPC [15], PSM [16], CHOL [17]).

Abbreviation	Full Chemical Name	Melting Temperature
DPPC	1,2-Dipalmitoyl-sn-glycero-3-phosphocholine	314 K
DOPC	1,2-Dioleoyl-sn-glycero-3-phosphocholine	256 K
POPC	1-Palmitoyl-2-oleoyl-sn-glycero-3-phosphocholine	271 K
PSM	Palmitoyl sphingomyelin	314 K
CHOL	Cholesterol	422 K

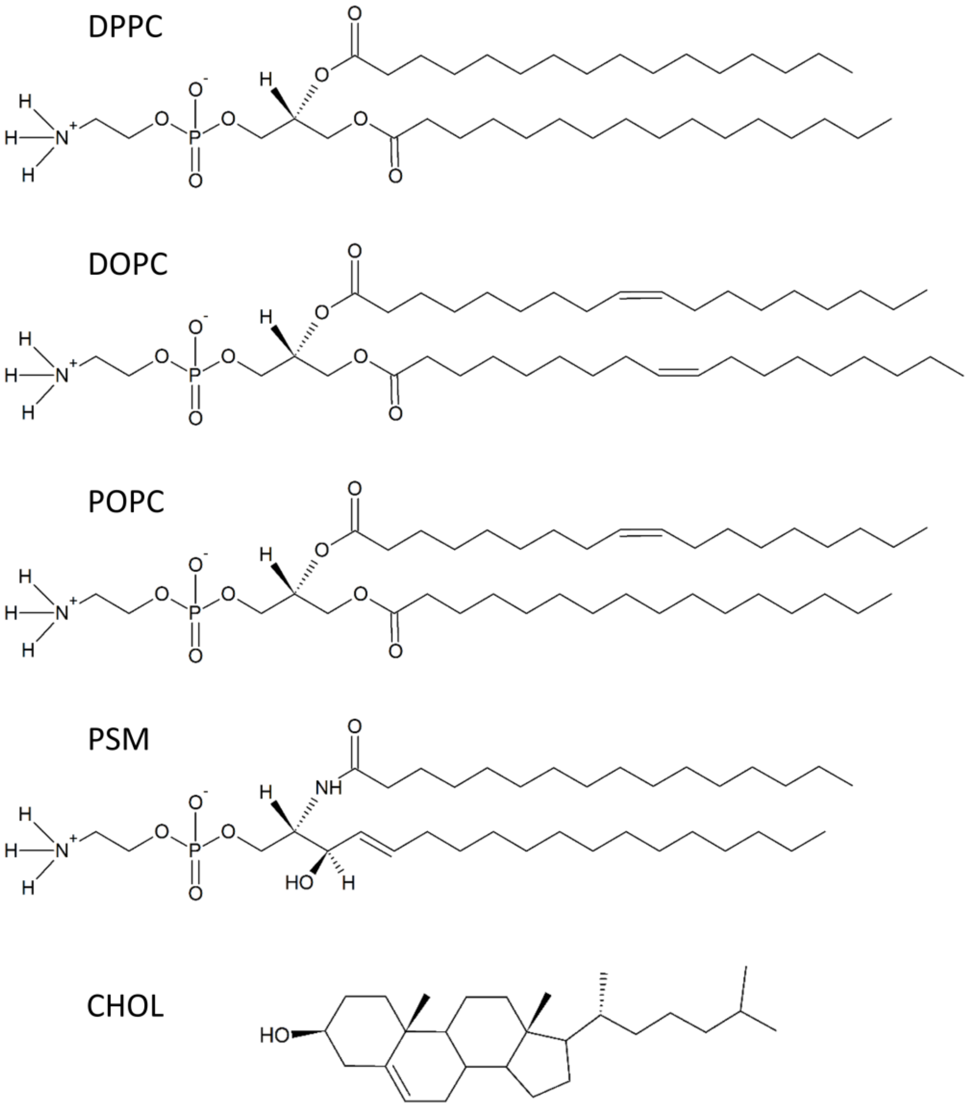

Figure 1.1: Chemical structures of lipids referenced throughout this document (Images created using the ChemSketch v14.01 [18]).

1.3 The Case for Simulation

Experimental methods to observe the spatiotemporal dynamics of membrane proteins and lipids have advanced significantly over the last decade, and especially in the last few years. In live cells, single particle tracking (SPT) [19]–[21] and fluorescence correlation spectroscopy (FCS) with subdiffraction detection volumes (achieved by stimulated emission depletion (STED) microscopy) [22] have revealed the plasma membrane to be heterogeneous on tens of nanometers, with correspondingly heterogeneous dynamics. Taken together, these results suggest a hierarchical membrane organization, with the cytoskeleton influencing transport above 80 nm length scales [23], and lipid-protein interactions operating below this length scale [24].

In model systems, mixtures that are comparatively simple are also heterogeneous. Neutron scattering reveals nanoscale liquid-ordered domains in vesicles comprised of a mixture of 3 or 4 components (including cholesterol) [25]. In a ternary mixture that supports liquid-ordered/liquid-disordered (Lo/Ld) coexistence nanoscale heterogeneities in composition and dynamics are observed by STED-FCS [26], provided the mixture is deposited on a glass support, which pins a fraction of lipids facing the support. In similar mixtures of a uniform Lo phase, heterogeneities are observed on yet smaller length scales and time scales by molecular dynamics simulations [27], [28] and interferometric scattering (iSCAT) based SPT [29].

While these experimental results point to essential aspects of spatiotemporal organization — the role of the cytoskeleton in partitioning the membrane, the existence of nanoscale compositional heterogeneity — they are mostly silent on the details of the underlying mechanism. For example, how does actin create a barrier to diffusion? By a simple steric mechanism [21], or by modifying the membrane viscosity in the neighborhood of actin binding proteins [30], or in some other way? How do nanoscale heterogeneities in composition modify local protein diffusion when observed (i.e., averaged) over the longer length and time scales relevant to signaling?

Answers to these questions will come from computational modeling approaches. Based on the above discussion, an appropriate modeling approach must fulfill certain criteria: (i) It must resolve protein-protein and lipid-protein interactions with reasonable chemical specificity, (ii) It must span length scales from individual lipids to the 100 nm length scale of the cortical cytoskeleton mesh, (iii) It must accurately resolve dynamics. In order to achieve this last point, a modeling approach must account properly for the hydrodynamics of the solvent adjacent to the membrane. This is clear from continuum arguments originally proposed by Saffman and Delbrück [7], and later extended by Hughes, Palinthorpe, and White [31].

1.4 Research Objectives

As computational capabilities have grown in recent decades, the scope of accessible molecular dynamics (MD) simulations has expanded from studies of hard spheres [32] and simple fluids [33] to virus capsids [34], coarse-grained virions [35], and millisecond simulations of single domain proteins [36]. Early simulations involving lipid bilayers were chiefly concerned with demonstrating self-assembly and obtaining sensible thermodynamic properties such as the thickness, area per lipid, NMR observables, and bending modulus of single component membranes [37]–[39]. Contemporary simulations have achieved sufficient complexity to study phase separation in multi-component mixtures [28], [40], [41] and the dynamic properties of relatively large membranes [42], [43].

For MD simulations with periodic boundary conditions, the long-range nature of hydrodynamic interactions in membranes leads to coupling between periodic images, resulting in significant hydrodynamic finite-size effects. An important consequence is the deviation in observed diffusion rates of lipids and transmembrane proteins when compared to predictions from Saffman-Delbrück (SD) theory [7], [31], [44]–[46]. In order to perform quantitative, predictive simulations of dynamics in complex membranes, it is necessary to operate in a regime that minimizes finite-size effects due to hydrodynamic interactions, which requires unconventionally large systems [44]. For lateral diffusion, simulated systems should be at least an order of magnitude larger than the Saffman-Delbrück length in all three dimensions.

Since all-atom force fields have SD lengths comparable to experiment (i.e., ≥ 100 nm), simulating sufficiently large all-atom systems for any appreciable length of time is infeasible for the foreseeable future. In contrast, coarse-grained force fields are tractable due to their softer lipid-lipid interactions, which result in lower membrane surface viscosity. For instance, membranes of Martini DPPC have a SD length of roughly 8.6 nm, which implies sufficiently large systems are 100-200 nm. Even so, a 100 nm cubic Martini system has roughly 8.5 million interaction sites, 95% of which are water. The majority of the computational effort is devoted to resolving interactions within the solvent. A more efficient treatment would be to replace the coarse-grained water particles with a mesoscopic hydrodynamics model coupled to the Martini lipids.

The first goal of my research was to produce a proof-of-concept implementation of this idea using a modified GROMACS v5.01 [47]. To forgo reparametrizing the lipid model, an implicit-solvent variation of Martini called Dry Martini [48] was used. The hydrodynamic momentum transport of the solvent was introduced through a mesoscopic hydrodynamics model called stochastic rotation dynamics (SRD), the resulting combination being called "stochastic thermostatted rotation dynamics" (STRD) Martini.

Once the code was finished, significant effort was required to optimize its communication protocol for efficient parallelization. This was an iterative process, featuring several promising protocols which were implemented and promptly discarded as better protocols were designed. The final communication code achieves a dramatic performance improvement relative to the originally published code, along with a crucial correction to the integrator. Taken together, the improvements allowed STRD Martini to perform 200 nm coarse-grained membrane simulations with scalable performance approaching the efficiency of Dry Martini, while including solvent hydrodynamics.

Recent work has provided a theoretical description for computing lateral diffusion in periodic membranes called periodic Saffman-Delbrück (PSD) theory [44], [46]. Diffusion in the membrane simulations has been successfully shown to follow this theory [45], but it lacks predictive power due to a dependence on two membrane parameters which are not readily available in the literature. A third goal of my research was to develop a method to independently calculate these parameters from non-equilibrium simulations and compare them with values fitted to the PSD theory. If the shear viscosity of a membrane is known, the theory is reduced to only one free parameter: the effective hydrodynamic radius of the diffusing object. When applied to lipid diffusion, this may provide an avenue for understanding the length scale where molecular and continuum hydrodynamics for lipids intersect.

1.5 Dissertation Outline

The next three chapters provide an overview of the background material relevant to this work. Principles of molecular dynamics simulations are discussed in Chapter 2, including algorithms for integration, temperature, and pressure control. A discussion of the force fields used throughout this work is also included. Chapter 3 develops the theory for low Reynolds number hydrodynamics, finishing with a description of membrane hydrodynamics and its application to a periodic system. This is followed by a discussion of the MPC algorithm in Chapter 4 along with a brief survey of other mesoscopic simulation models.

The final three chapters present results for the STRD Martini membrane simulation model and membrane viscosity measurements from nonequilibrium shearing simulations. Chapter 5 discusses the GROMACS implementation of STRD Martini, giving rationales for the coupling method and parameter selection before showing the results of validation testing. It ends with an application of STRD Martini that demonstrates an important hydrodynamic finite size effect on lateral diffusion in the membrane. Significant effort went into optimizing the communication code to make the performance of STRD Martini competitive with Dry Martini. This work is detailed in Chapter 6, along with results from a series of performance benchmarks. Chapter 7 presents membrane surface viscosity results along with the theory required to calculate them.

Chapter 2

MOLECULAR DYNAMICS SIMULATION

2.1 Principles of Molecular Dynamics

Molecular dynamics (MD) simulations study the dynamics of many-particle molecular systems by numerically integrating the classical laws of motion [49]. MD particles may represent whole molecules, atoms, or groups of atoms, depending on the level of chemical specificity required by the system under consideration. Typical lipid membrane simulations either involve atoms (***all-atom*** simulations) or groups of atoms (***coarse-grained*** simulations, in the parlance of the field). The positions and velocities of each particle are represented with continuous variables, while time is discretized into regular intervals of duration Δt called timesteps. At each timestep, the force acting on each particle is calculated through its interactions with other particles, parameterized by a ***force field***. The equations of motion are integrated for each particle with these forces to produce new positions and velocities for the next timestep. This procedure is repeated many times to gradually generate ***trajectories*** for each particle in the simulation (see **Fig 2.1**).

There are many well-known algorithms for numerical integration of ordinary differential equations using finite differences, including the Euler, Runge-Kutta, Verlet, and leapfrog methods [50]. MD programs call code responsible for integrating the equations of motion ***integrators***. Careful algorithmic considerations are necessary to minimize integration error and produce physical trajectories. Not every algorithm for numerical integration is suitable for MD. Some produce trajectories which do not conserve energy. Others require multiple force computations per timestep, which is overwhelmingly the most computationally laborious part of any MD program. Other popular integrators do not conserve energy, modelling the dynamics of particles coupled to a heat bath via random impulses and dissipative forces.

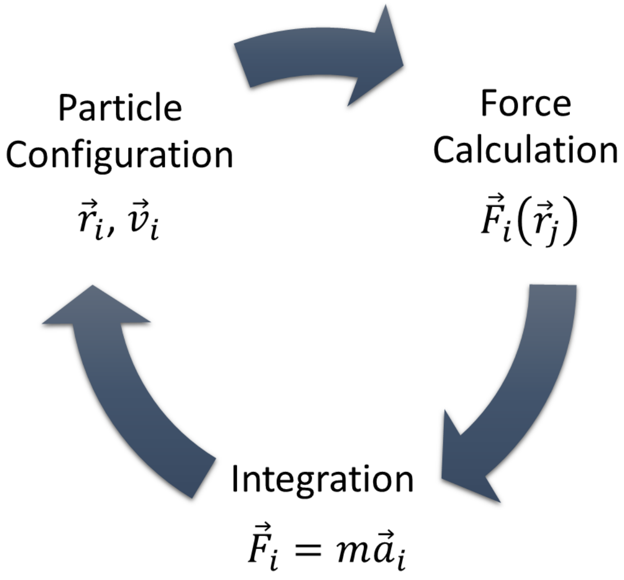

Figure 2.1: The central loop of an MD simulation. Forces are computed based on the particle configuration and the equations of motion are integrated using these forces, producing and new configuration. Millions to billions of these iterations are required to produce nano- to microsecond trajectories.

Interactions between particles are divided between intramolecular and intermolecular. Intramolecular interactions are responsible for maintaining realistic molecular structure and providing rotational and vibrational internal degrees of freedom. Intermolecular interactions include electrostatic and van der Waals interactions. Atoms within molecules have partial charges which reflect their average electronic structure in the molecule, but no further attempt is made to model electronic degrees of freedom. In the most widely used MD models, the intermolecular interactions are assumed to be pairwise-additive. Particle definitions and their interactions are defined using a set of parameterized potential energy functions, collectively called the "force field" of the model. Popular force fields have been developed at various levels of granularity, include those which include every atom in the system [51]–[53], those which use coarse-grain interaction sites [48], [54], and those in between [55].

As a practical matter, tractable MD simulations of meaningful duration are limited to $\leq 10^8$ particles, often much fewer on modest computing resources. Despite steady and impressive growth in simulation scale over time [34], [35], [42], [43], these still represent fairly small systems (10^8 atoms comprise a 100 nm cube of water). Consequently, our MD simulations cannot properly model bulk systems unless periodic boundary conditions (PBC) are employed (see **Fig 2.2**). Using PBC eliminates boundary effects from unphysical surfaces at the edges of the simulation box in exchange for finite size effects imparted through self-interactions with periodic images. In many contexts, this can be ameliorated through the use of larger simulation sizes. However, since periodic boundary conditions break rotational symmetry, total angular momentum is not generally conserved in MD simulations.

Rigorous integration results in MD trajectories which sample configurations from the microcanonical (NVE) ensemble. Simulation of other ensembles is possible through temperature coupling (NVT) [56]–[60] or pressure coupling (NPT) [57], [61]–[65]. These coupling methods are particularly useful for relaxing systems to a state of equilibrium prior to longer "production" simulations. Careful consideration should be applied when selecting a coupling algorithm to ensure the equilibrium fluctuations generate the correct ensemble and produce a trajectory which satisfies ergodicity [60], [66], [67].

When used properly, MD simulations can be used as a "computational microscope" [68] to calculate quantitative dynamics of complex biological systems and to understand their qualitative molecular behavior on length and time scales that are inaccessible to experiment. Popular MD software packages include GROMACS [47], NAMD [69], CHARMM [70], LAMMPS [71], and others [72], [73]. This work was performed exclusively in the context of GROMACS v5.0.1.

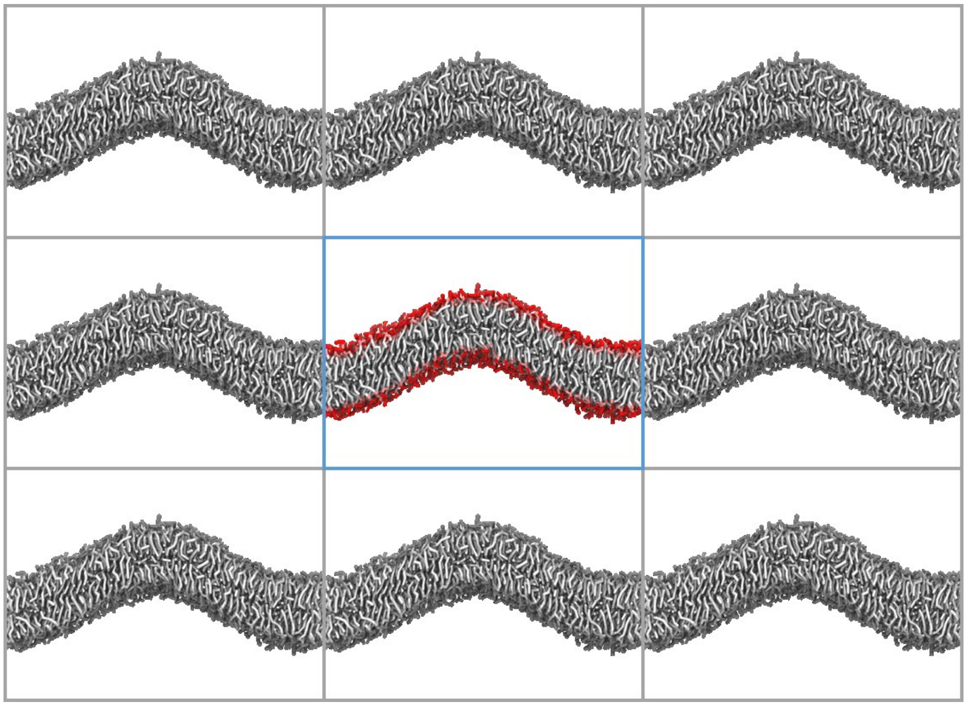

Figure 2.2: Periodic boundary conditions illustrated in two dimensions with an exaggerated membrane undulation. The central cell (in color) contains the real system which interactions with an infinite lattice of periodic images (greyscale). Particles which exit one side of the periodic box emerge on the opposite side.

2.2 Integrators

At the heart of every molecular dynamics simulation is a simple numerical integration of Newton's third law (with $\vec{f} = \vec{F}/m$)

$$\frac{d^2}{dt^2}\vec{r} = \vec{f} \tag{2.1}$$

for each particle given the set of initial positions $\vec{r}(0)$ and velocities $\vec{v}(0)$. Forces are assumed to depend only on the instantaneous particle positions $\vec{f} = \vec{f}(\vec{r}_1, \vec{r}_2, \vec{r}_3, \dots)$. Many techniques have been developed for solving this type of initial value problem based on finite differences [50], [74]. Since the most computationally expensive part of any MD code is the calculation of forces, we are limited to integrators which only require a single force computation per step. This restriction rules out the popular Runge-Kutta method along with various iterative predictor-corrector methods [74] (e.g. Gear [75]).

The simplest integrator involving a single force evaluation is the Euler method, invented in 1768 based on the forward difference in position

$$\vec{r}_{n+1} = \vec{r}_n + \Delta t\, \vec{v}_n + \frac{\Delta t^2}{2}\vec{f}_n \tag{2.2}$$

$$\vec{v}_{n+1} = \vec{v}_n + \Delta t\, \vec{f}_n \tag{2.3}$$

with the subscripts denoting the timestep. The forward difference is analogous to a Taylor expansion of $\vec{r}(t)$ in time. Despite its simplicity, this method is not used in molecular dynamics due to a serious problem: it does not conserve energy, which can be inferred from the fact that it breaks time-reversal symmetry.

In order to conserve energy, integrators used in MD programs must be symplectic. That is, they must transform the generalized coordinates $(\vec{q}, \vec{p})$ of the system's Hamiltonian in a manner that conserves phase-space volume. GROMACS implements two equivalent integrators which fulfill this requirement, ***leap frog*** [76] and ***velocity Verlet*** [77]. To obtain them, we can add together the Taylor series for the forward and backward finite differences

$$\vec{r}_{n+1} = \vec{r}_n + \Delta t \frac{d}{dx}\vec{r}_n + \frac{\Delta t^2}{2}\frac{d^2}{dx^2}\vec{r}_n + \frac{\Delta t^3}{6}\frac{d^3}{dx^3}\vec{r}_n + O(\Delta t^4) \tag{2.4}$$

$$\vec{r}_{n-1} = \vec{r}_n - \Delta t \frac{d}{dx}\vec{r}_n + \frac{\Delta t^2}{2}\frac{d^2}{dx^2}\vec{r}_n - \frac{\Delta t^3}{6}\frac{d^3}{dx^3}\vec{r}_n + O(\Delta t^4) \tag{2.5}$$

$$\vec{r}_{n+1} + \vec{r}_{n-1} = 2\vec{r}_n + \Delta t^2 \frac{d^2}{dx^2}\vec{r}_n \tag{2.6}$$

Solving for $\vec{r}_{n+1}$ gives the Verlet [78] integrator

$$\vec{r}_{n+1} = 2\vec{r}_n - \vec{r}_{n-1} + \Delta t^2 \vec{f}_n \tag{2.7}$$

$$\vec{v}_n = \frac{\vec{r}_{n+1} - \vec{r}_{n-1}}{2\Delta t} \tag{2.8}$$

This form is inconvenient for MD code because it requires storing three sets of positions in order to compute the velocity at each step. Though this computation is optional for integration, velocities are needed to compute important scalar quantities such as kinetic energy, temperature, and pressure.

Another problem is the loss of precision from force term due to the Δt^2 factor, which may be small compared to the position terms. It is desirable to reformulate the Verlet integrator to be linear in Δt. This can be accomplished by solving for the velocities between timesteps, called the off-step or midpoint velocities. This yields the ***leapfrog*** integrator [76]

$$\vec{v}_{n+1/2} = \vec{v}_{n-1/2} + \Delta t\, \vec{f}_n \tag{2.9}$$

$$\vec{r}_{n+1} = \vec{r}_n + \Delta t\, \vec{v}_{n+\frac{1}{2}} \tag{2.10}$$

Which can be shown to equal the Verlet integrator if we define

$$\vec{v}_{n-\frac{1}{2}} = \frac{\vec{r}_n - \vec{r}_{n-1}}{\Delta t} \tag{2.11}$$

This integrator is fast, easy to implement, accurate to third-order, time-reversible, and memory-efficient. It is the default integrator of GROMACS. However, it does not supply simultaneous values for $\vec{r}_n$ and $\vec{v}_n$, with important consequences for the mesoscopic hydrodynamics model implemented in this work. Fortunately, there is an equivalent, synchronized form of leapfrog implemented by GROMACS which yields $\vec{r}_n$ and $\vec{v}_n$ called the ***velocity Verlet*** integrator [77]

$$\vec{v}_{n+1/2} = \vec{v}_n + \frac{\Delta t}{2}\vec{f}_n \tag{2.12}$$

$$\vec{r}_{n+1} = \vec{r}_n + \Delta t\,\vec{v}_{n+\frac{1}{2}} \tag{2.13}$$

$$\vec{v}_{n+1} = \vec{v}_{n+\frac{1}{2}} + \frac{\Delta t}{2}\vec{f}_{n+1} \tag{2.14}$$

Since these two integrators are equivalent, they will produce the same trajectory given the same initial conditions.

Choosing an appropriate timestep is essential for stable integration. Though we seek the longest feasible timesteps for computational efficiency, stable integration requires the timestep to be smaller than the period of oscillation for the fastest degree of motion in the simulation. This imposes a maximum timestep duration of about 2 fs for all-atom simulations and 20 fs for coarse-grain simulations like the Martini models [48], [54] used in this work.

2.3 Temperature and Pressure Control

Accurate integration produces particle trajectories that conserve energy. These MD simulations sample particle configurations from the microcanonical (NVE) ensemble. This is not always desirable. In some situations, we might rather simulate systems at constant temperature T_0 or constant pressure P_0. For example, one might be interested in non-equilibrium simulations where driving forces or deformations are applied, and work done on the system must be dissipated as heat, or in situations where dissipative forces are important. One might also be interested in fluctuation-dependent properties sampled from the canonical ensemble. On a practical basis, enforcing a prescribed temperature over long simulations is often necessary to combat the slow energy drift from accumulated numerical noise in the integration. Allowing pressure and temperature to change is also useful for relaxing the initial configuration of a system to its equilibrium state. This section will discuss the various temperature and pressure coupling schemes used throughout the rest of this work.

2.3.1 Temperature

The algorithm used for temperature control is called the ***thermostat***. There are many ways to control temperature in a simulation. One can add or remove kinetic energy by rescaling particle velocities, adjust the equations of motion to include dissipation or coupling to a heat bath, or simply draw new velocities from a Maxwell-Boltzmann distribution at random.

The instantaneous temperature of system with N_{df} degrees of freedom is related to the kinetic energy of the constituent particles (indexed by α) with

$$\frac{1}{2}N_{df}k_BT = \frac{1}{2}\sum m_\alpha\, v_\alpha^2 \qquad (2.15)$$

$$T = \frac{1}{N_{df}k_B}\sum m_\alpha\, v_\alpha^2 \qquad (2.16)$$

Since different parts of a simulation may be coupled to independent heat baths, N_{df} is equal to three times the number of particles in the subsystem under consideration less the number of constraints among those particles.

The most straightforward way to adjust the temperature of a group is to rescale all velocities by a global scaling factor $\lambda(T)$

$$T' = \frac{1}{N_{df} k_B} \sum m_\alpha \, (\lambda v_\alpha)^2 \tag{2.17}$$

The scaling factor required for a prescribed temperature T_0 may be found by dividing this expression by the instantaneous temperature

$$\frac{T_0}{T} = \left(\frac{1}{N_{df} k_B} \sum m_\alpha \, (\lambda v_\alpha)^2 \right) \left(\frac{1}{N_{df} k_B} \sum m_\alpha \, v_\alpha^2 \right)^{-1} \tag{2.18}$$

$$\frac{T_0}{T} = \lambda^2 \left(\sum m_\alpha \, v_\alpha^2 \right) \left(\sum m_\alpha \, v_\alpha^2 \right)^{-1} \tag{2.19}$$

$$\lambda = \sqrt{T_0/T} \tag{2.20}$$

The prescribed temperature is then maintained by periodically recalculating the instantaneous temperature and rescaling all particle. This does not have to be done after every timestep, as the average temperature drift per step is small. This approach is perfectly valid in the thermodynamic limit but suffers from an important drawback when used for finite systems: it does not admit fluctuations in the kinetic energy. We may improve upon this by coupling the system to an external heat bath and allowing the prescribed temperature to gradually change.

The well-known ***Berendsen*** thermostat [57] implements this coupling with a first-order differential equation using a relaxation time τ_T

$$\frac{dT}{dt} = \frac{T_0 - T}{\tau_T} \tag{2.21}$$

It can be recast into a velocity rescaling factor applied every timestep given by

$$\lambda = \sqrt{1 + \frac{\Delta t}{\tau_T}\left(\frac{T_0}{T} - 1\right)} \tag{2.22}$$

Since the equation is first-order, the solution is non-oscillatory and any deviation from the prescribed temperature decay exponentially with time. Both properties make this thermostat ideally suited to initial equilibration (i.e. relaxation). Unfortunately, the exponential decay excessively dampens thermal fluctuations; the resulting dynamics do not sample from the canonical ensemble.

The ***Bussi-Donadio-Parrinello*** thermostat [60] (colloquially referred to as "Bussi" or "v-rescale" in GROMACS) is a modified Berendsen thermostat with thermodynamically correct kinetic energy fluctuations introduced through a Wiener noise term dW in the coupling equation

$$dT = \frac{(T_0 - T)}{\tau_T} dt + 2\sqrt{\frac{T_0 T}{N_{df}\tau_T}} dW \tag{2.23}$$

Just like Berendsen, this equation is first order in time, decaying exponentially to the prescribed temperature when the system is far from equilibrium. One satisfying property of this thermostat is that it has a conserved quantity analogous to total energy, which is useful when error checking new code. This is the thermostat we use when running STRD Martini simulations.

An alternative approach having its own conserved quantity is the ***Nosé-Hoover*** thermostat [58], [59]. Rather than rescaling the velocities, this thermostat couples particle dynamics to an external heat bath through the introduction of a damping term in the particle equations of motion

$$\frac{d^2}{dt^2}\vec{r}_i = \vec{f}_i - \frac{p_\gamma}{Q}\vec{v}_i \qquad (2.24)$$

With γ being a new dimensionless degree of freedom associated with the heat bath having momentum p_γ. Q is the "mass" of the heat bath, which controls the strength of the coupling (large Q corresponds to weak coupling). The momentum of the heat bath is evolved in time according to

$$\frac{dp_\gamma}{dt} = (T - T_0) \qquad (2.25)$$

We can write a Hamiltonian for this extended system including the heat bath degree of freedom and identify a conserved quantity with it. In fact, this thermostat predates v-rescale; it was the first widely-used thermostat to correctly sample the microcanonical ensemble while maintaining a conserved quantity. However, the original Nosé-Hoover thermostat has problems with ergodicity; given infinite time, it will not exhaustively sample phase space. This may be alleviated by recursively coupling multiple Nosé-Hoover thermostats together in a chain [66]. This is the thermostat used in this work when running simulations with the CHARMM force field.

The Dry Martini force field uses the ***stochastic dynamics*** thermostat based on the Langevin equation

$$\frac{d^2}{dt^2}\vec{r}_i = \vec{f}_i - \gamma\vec{v}_i + \sqrt{2\gamma k_B T}\,\xi_i(t) \qquad (2.26)$$

$$\langle \xi_i(t) \cdot \xi_j(t+\tau) \rangle = \delta(\tau)\delta_{ij} \qquad (2.27)$$

Where the damping constant γ controls the coupling strength and the ξ_i force is a delta correlated Gaussian process. This thermostat is both a thermostat and an integrator, which may be used to mimic the effects of viscous drag and thermal noise of a solvent for implicit-solvent models like Dry Martini. A major drawback of the Langevin and other stochastic approaches for hydrodynamics problems is that they scramble the velocity correlations responsible for hydrodynamic flows [79].

2.3.2 Pressure

The algorithm used for pressure control is called the ***barostat***. The pressure in a simulation may be controlled by coupling the size and shape of the simulation box to the instantaneous pressure tensor. When the pressure is greater than the prescribed value P_0, the box is allowed to gradually expand. The 3x3 virial pressure tensor $P_{\alpha\beta}$ in an MD simulation is defined to be the difference between the energy tensor $E_{\alpha\beta}$ and the virial tensor $\Xi_{\alpha\beta}$, each defined as a sum over atoms i, j

$$P_{\alpha\beta} = \frac{2}{V}\left(E_{\alpha\beta} - \Xi_{\alpha\beta}\right) \tag{2.28}$$

$$E_{\alpha\beta} = \frac{1}{2}\sum_{i} m_i(\vec{v}_i \otimes \vec{v}_i) \tag{2.29}$$

$$\Xi_{\alpha\beta} = -\frac{1}{2}\sum_{j}\sum_{i<j} \vec{r}_{ij} \otimes \vec{F}_{ij} \tag{2.30}$$

Where $\otimes$ denotes an outer product of two vectors, $C_{\alpha\beta} = A_\alpha \otimes B_\beta$, and $\vec{F}_{ij}$ gives the pairwise force between atoms i and j.

Hence, we may write the total expression for the pressure as

$$P_{\alpha\beta} = \frac{1}{V}\left(\sum_{i} m_i(\vec{v}_i \otimes \vec{v}_i) + \sum_{j}\sum_{i<j} \vec{r}_{ij} \otimes \vec{F}_{ij}\right) \tag{2.31}$$

This is just the instantaneous virial stress after a change in sign, i.e., a positive stress is expansive while a positive stress is compressive. The scalar pressure is given by

$$P = \frac{1}{3}\mathrm{Tr}\left(P_{\alpha\beta}\right) \tag{2.32}$$

Pressure coupling can be performed uniformly in all dimensions using this scalar pressure (isotropic coupling), independently for each axis using the full pressure tensor (anisotropic coupling) or separately for the lateral (x and y) and normal (z) axes using the pressure tensor (semiisotropic coupling).

In the ***Berendsen*** [57] scheme, this is done by rescaling the simulation box and all particle coordinates by a scaling matrix $\mu_{\alpha\beta}$, calculated from a first-order equation based on the difference between the instantaneous and prescribed pressure tensor

$$\mu_{\alpha\beta} = \delta_{\alpha\beta} - \beta_{\alpha\beta}\frac{\delta t}{\tau_p}\frac{\left(P_{0,\alpha\beta} - P_{\alpha\beta}\right)}{3} \tag{2.33}$$

Where δt is the time between pressure coupling steps, τ_p is the time constant of the relaxation, and $\beta_{\alpha\beta}$ the isothermal compressibility tensor. Like the Berendsen thermostat, this barostat is particularly well suited to equilibration simulations because it converges exponentially toward P_0. Other barostats should be used in situations where pressure fluctuations are important, such as Parrinello-Rahman [61], [62] or Martyna-Tuckerman-Tobias-Klein [63], [64]. Aside from equilibrium CHARMM simulations, which used Parrinello-Rahman, Berendsen was the only barostat used in this work, as pressure coupling was predominantly used for relaxation purposes, with “production” simulations running in the constant volume (NVT) ensemble.

2.4 Interactions and the Force Field

MD simulations use classical interactions to calculate forces between atoms. These interactions are classified as either bonded (intramolecular) or nonbonded (intermolecular), with the bonded interactions describing conformational and vibrational degrees of freedom. Since using classical interactions is an approximation, their mathematical form often follows from a compromise between physical accuracy and computational efficiency, with the task of reproducing relevant physical properties being relegated to the force field parameterization.

The total potential energy in a GROMACS simulation may be written as the sum of six major contributions:

$$\begin{aligned} U_{TOT} = &\sum_{i}\sum_{i<j} 4\varepsilon_{ij}\left[\left(\frac{\sigma_{ij}}{r_{ij}}\right)^{12} - \left(\frac{\sigma_{ij}}{r_{ij}}\right)^{6}\right] + \sum_{i}\sum_{i<j}\frac{q_i q_j}{4\pi\varepsilon_r r_{ij}} \\ &+ \sum_{bonds} K_{b,i}(l_i - l_{0i})^2 + \sum_{angles} K_{a,i}(\theta_i - \theta_{0i})^2 \\ &+ \sum_{dihedrals} K_{\varphi,i}(1 + \cos(n\varphi - \delta_0)) \\ &+ \sum_{impropers} K_{\omega,i}(\omega_i - \omega_{0i})^2 \end{aligned} \tag{2.34}$$

The first term is the pairwise sum of van der Waals interactions, modeled with 12-6 Lennard-Jones potentials, and the second term is the pairwise sum of electrostatic interactions. The next two terms quantify energetics for vibrational degrees of freedom, starting with the energy cost of individual bond stretching in the *bonds* term and bending in the *angles* term. The fifth *dihedral* term describes rotations about the axis defined by a sequence of four covalently bonded atoms. The last term restrains certain planar bonding geometries to preserve molecular structure. For each sum, the K_x coefficients give the associated energy, while zero-subscripted quantities are equilibrium values. Both are supplied as force field parameters.

2.4.1 Bonded Interactions

There are four common types of “bonded” interactions: stretching, bending, dihedral, and improper. Some force fields include additional intramolecular interactions to impose further restraints on molecular structure. For example, the

CHARMM force field includes a virtual harmonic bond between the outer (1-3) atoms of a bonded triplet called the Urey-Bradley force.

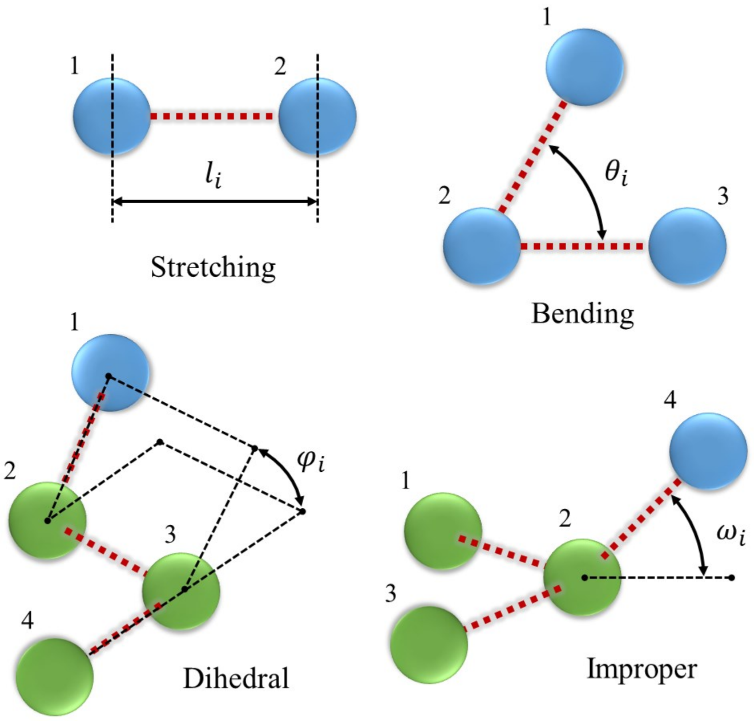

Figure 2.3: Types of bonded interactions with covalent bonds indicated by dashed red lines. Green atoms for dihedral and improper bonds are co-planar.

Fig. 2.3 illustrates the four bonded interaction terms from equation 2.34. Bonds between individual molecules are modeled as harmonic springs having an equilibrium length l_0. The angle formed by a bonded triplet is also restrained using a harmonic potential about the equilibrium bond angle θ_0. The dihedral (also called the torsion) is defined as the angle between the planes formed by the 1-2-3 and 2-3-4 triplets in **Fig. 2.3**. Impropers are another form of dihedrals, defined in the same manner but used for bond geometries other than torsions, usually to maintain planarity of chemical groups. One configuration is depicted in **Fig. 2.3** which may be used to enforce a specific 3D structure. Other configurations include the plane formed by three atoms connected to a central atom (i.e. for one side of a tetrameter) or the plane formed by three consecutive atoms in a four-atom chain. The latter may be used to enforce planarity for ring structures.

Due to the inherently quantum nature of their bonded interactions, nonbonded interactions are typically omitted among atoms connected over a series of one or two covalent bonds (called 1-2 and 1-3 exclusions respectively). Neglecting interactions among third-nearest (1-4 exclusion) atoms is uncommon. However, a scaling factor is often used to reduce the forces between such atoms. This feature is used by the GROMOS and OPLS force fields.

2.4.2 Nonbonded Interactions

2.4.2.1 Van der Waals

The van der Waals (dispersion) and exchange repulsion forces between atoms *i* and j in GROMACS are usually modeled with the well-known 12-6 Lennard-Jones interaction, given by

$$U_{ij}^{LJ}(r_{ij}) = 4\varepsilon_{ij}\left[\left(\frac{\sigma_{ij}}{r_{ij}}\right)^{12} - \left(\frac{\sigma_{ij}}{r_{ij}}\right)^{6}\right] \tag{2.35}$$

With the interaction parameters ε_{ij} and σ_{ij} depending on the specific atom types of the pair under consideration. These parameters may either be defined on a pairwise basis for each pair of M atom types in a $M \times M$ interaction matrix (as in Martini), or individually for each atom type (as in CHARMM). In the latter case, a ***combination***

rule is required to determine the pairwise parameters from the atomistic parameters given atom types i and j. A commonly-used combination rule is

$$\varepsilon_{ij} = \sqrt{\varepsilon_i \varepsilon_j} \tag{2.36}$$

$$\sigma_{ij} = \frac{1}{2}\left(\sigma_j + \sigma_j\right) \tag{2.37}$$

The Lennard-Jones potential is illustrated in **Fig. 2.4**. The attractive part scales as r^{-6}, which has a physical interpretation as the dispersion force arising from the correlated motions of electrons in each molecule (sometimes described as coupling between instantaneous molecular dipoles). The repulsive part scales as r^{-12} for computational efficiency, as this is simply the square of the attractive part.

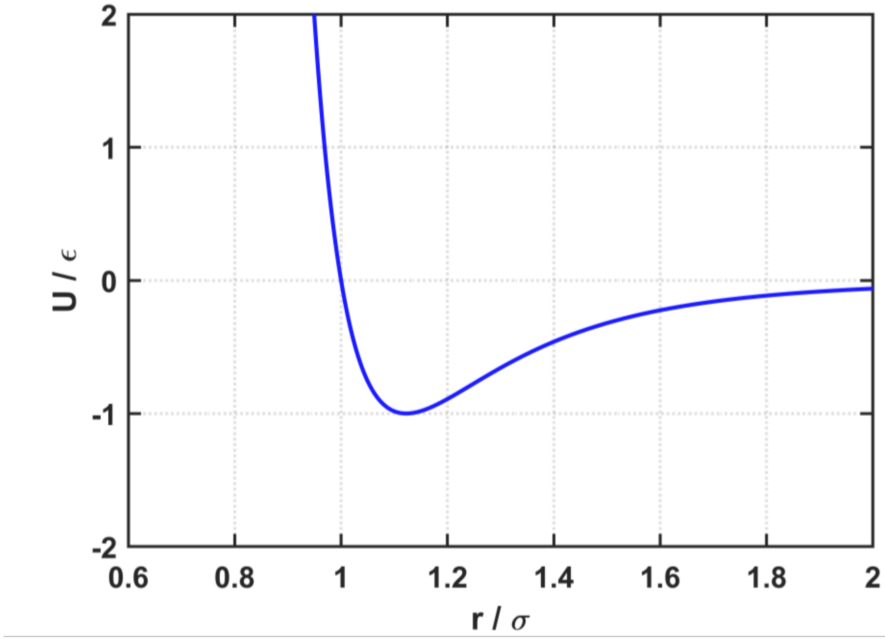

Figure 2.4: The 6-12 Lennard-Jones potential in natural units σ and ε. The potential is zero at $\sigma = 0$ and the depth of the well is $-\varepsilon$.

Since this is a short-range interaction, enormous computational effort may be spared by ignoring all interactions involving interparticle distances over a certain cutoff length scale R_c. A naive implementation of the short-range force computation requires $O(N^2)$ pairwise distance calculations to iterate over all particle pairs. When cutoffs are employed, this can be improved to $O(N)$ through the use of ***neighbor lists***: per-atom lists of all other atoms within a certain distance $R_{nl} \geq R_c$ [78]. When using these lists, each atom needs only consider interactions with particles from its neighbor list. Since the number of particles in this $\sim {R_{nl}}^3$ region is independent of the total system size, the time required for force evaluation becomes linear. These lists may also be constructed in linear time in a two-stage process. First, all particles are binned in a spatial cell grid with bins $\geq R_{nl}$. Second, the neighbor list for each particle is constructed from among the atoms of the 27 nearest bins (including its own home bin). When using a neighbor list cutoff larger than the force cutoff, these lists may be calculated infrequently, so long as particles from outside the neighbor list region do not diffuse into the force cutoff radius. The required frequency of neighbor list updates is thus dependent upon the chosen thickness of the boundary region $\delta = R_{nl} - R_c$.

2.4.2.2 Electrostatics

Charged particles (e.g., dissolved ions or atoms with partial charges) have pairwise electrostatic interactions given by the well-known Coulomb potential

$$\phi_{ij}(r_{ij}) = \frac{q_i q_j}{4\pi\varepsilon_r r_{ij}} \qquad (2.38)$$

In contrast with the Lennard-Jones potential, this is a long-ranged interaction. Applying a cutoff leads to serious artifacts [80]. Summing over all particles in a periodic system is prohibitively expensive unless an Ewald sum [81] is used, decomposing the interaction into short-range and long-range parts according to

$$\phi_{ij}(r_{ij}) = \phi_{ij}^{SR}(r_{ij}) + \phi_{ij}^{LR}(r_{ij}) \tag{2.39}$$

$$\phi_{ij}^{SR}(r_{ij}) = \operatorname{erfc}(\beta r_{ij}) \frac{q_i q_j}{4\pi \varepsilon_r r_{ij}} \tag{2.40}$$

$$\phi_{ij}^{LR}(r_{ij}) = \operatorname{erf}(\beta r_{ij}) \frac{q_i q_j}{4\pi \varepsilon_r r_{ij}} \tag{2.41}$$

With the parameter β controlling the length scale of the decomposition. The sum over short-range contributions now rapidly converges. In reciprocal space, the long-range contribution is given by

$$\phi_{ij}^{LR}(k) = \frac{q_i q_j}{4\pi \varepsilon_r k^2} \exp\left(\frac{-k^2}{4\beta}\right) \tag{2.42}$$

which also rapidly converges. Using this method, we may impose a cutoff in real space and compute the long-range contribution in reciprocal space. Using a short cutoff in real space yields $O(N^2)$ scaling in reciprocal space. The optimal choice of cutoff with this method is to equal the square root of the periodic box size, resulting in $O(N^{3/2})$ scaling [82].

This procedure is further improved by computing the reciprocal contribution on a discrete grid using a fast Fourier transform, complete in $O(N \log N)$ time. This method is called the particle mesh Ewald (PME) method [83]. Charges in the system are distributed over grid points using an interpolation scheme and the resulting forces are interpolated back afterward. This is the standard electrostatics method used in GROMACS. A more efficient method based on multipole sums scales with $O(N)$ [84]–[86], but the prefactor is prohibitively large unless large systems are being simulated. Detailed comparisons among these and other methods are available in the literature [87], [88].

2.5 Force Fields

The set of all atom types and their interactions in a MD simulation is called the chemical "force field". The selection of these parameters involves an arcane and arduous process called "parameterization" which may include quantum mechanics calculations and repeated MD simulations to tune the thermodynamic properties. There are three common levels of modeling resolution: all atom (AA) models [52], [53], [89], which account for every atom in the system; united atom (UA) models [55], which combine certain hydrogen and carbon atoms (e.g., the hydrogens on methylene bridges and terminal methyl groups); and coarse-grained (CG) models, which combine groups of atoms into effective interaction sites [48], [54]. **Fig. 2.5** illustrates the difference in model resolution between all-atom and coarse-grain force fields for a DPPC lipid.

Treatment of the solvent is an important component of the parameterization. Many treatments exist for water molecules, from complex models including many-body interactions [90] to simple ones which treat each molecule as a single Lennard-Jones interaction site with extra sites for electrostatics [91]. The design and parameterization of these water models for biophysical simulation is a rich field of study in its own right [92], [93]. Certain force fields designed for membrane simulations are parameterized without including water at all [48], [94]–[96], opting instead to incorporate the influence of solvation and hydrophobicity into the lipid-lipid interactions. These are called "implicit-solvent" models. Another recent effort has been to properly account for polarizability in the solvent (and solute) model. Such efforts have been applied in the context of all-atom [97], [98] and coarse-grain [99] force fields. Other work has taken the opposite approach, using additional computation effort to incorporate many-body effects [100], [101].

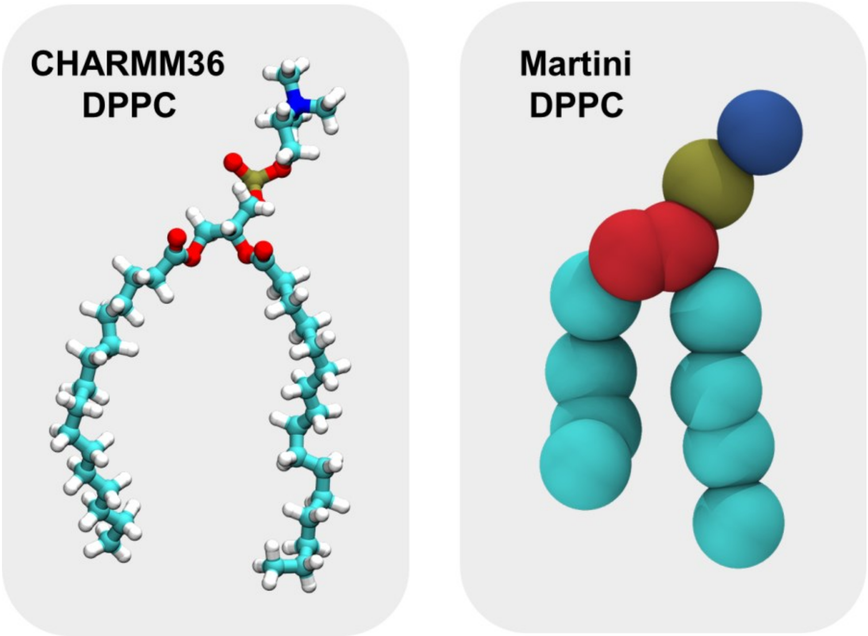

Figure 2.5: A lipid represented in both all-atom and coarse-grain force fields.

CHARMM36 [51], [89], [102]–[104] is the latest version of an all-atom force field which was originally developed for the CHARMM [70] MD program. The CHARMM lipid force field has been supported for many years. In that time, it has been through multiple revisions [102], [103] and has seen the introduction of proteins [89] and nucleic acids [102]. Excellent reviews are available in the literature concerning the meticulous parameterization process [105] and the development of the lipid force field [106]. The CHARMM36 force field is fully compatible with the

GROMACS simulation code. Proper integration of the vibrational modes requires 1 fs timesteps. The fastest of these are the hydrogen bond stretching modes. These are often constrained to their equilibrium length when using CHARMM to permit 2 fs timesteps. GROMACS implements these constraints in parallel using the LINCS constraint algorithm [107], [108].

The most popular coarse-grain forcefield is ***Martini*** [54], [109], [110], which maps groups of 3-4 heavy atoms to chemically distinct coarse-grain interaction sites, informally called "beads". The solvent is modeled with a bath of such beads, each representing four water molecules. An excellent review is available by the architects of the force field [111]. It consists of nearly two dozen modular "building blocks" characterized by their polarity (e.g. polar, nonpolar, apolar, and charged) and hydrogen bonding affinity or degree of polarity. A single interaction matrix defines all interactions among these building blocks with parameters obtained largely though calculations of partitioning free energies for small organic compounds constructed from them. All-atom structures are converted to coarse-grain structures by mapping the center of mass for a group of atoms to a single bead. Bonded interactions for the coarse-grain structure are determined by comparing distribution functions of these beads to those of the corresponding centers of mass from all-atom simulations.

Martini does not use long-range electrostatics. The electrostatic potential is shifted to zero between 0 and 12 Å and a relative dielectric constant of 15 is used for screening. Interactions between the Martini beads are softer than their all-atom counterparts with fewer degrees of freedom. Generally speaking, reducing the friction and smoothing out the energy landscape in this way accelerates the pace of the simulation, speeding up the kinetics and permitting faster conformational changes.

Identifying a global time-rescaling factor to obtain “real” times is often not feasible as the speedup factor is not consistent among different degrees of freedom within the system [112]. A smoother energy landscape and slower vibrations also means the fastest timescale of the system is considerable slower than in an all-atom simulation, permitting the use of time steps from 20 to 40 fs.

Despite its simplicity, the Martini forcefield has been incredibly successful in studies of membrane dynamics [113], [114] and has even been extended to model proteins [109].

Dry Martini [48] is a variant of Martini parameterized for implicit-solvent simulations. The interaction levels have been adjusted to ensure that a stable bilayer will self-assemble even in the absence of solvent. Other membrane properties such as area per lipid, area compressibility, and tail order parameter are in agreement with ordinary Martini. One advantage of avoiding a full reparameterization is that many lipids, and even some proteins, parameterized for Martini can often be used with Dry Martini simply by rescaling interactions. Removal of the solvent necessitates the addition of a stochastic term to the integration in order to provide a source of friction and thermal noise. This has the side effect of drastically reducing lipid self-diffusion. The difference between Martini and Dry Martini for a small membrane is illustrated in **Fig. 2.6**.

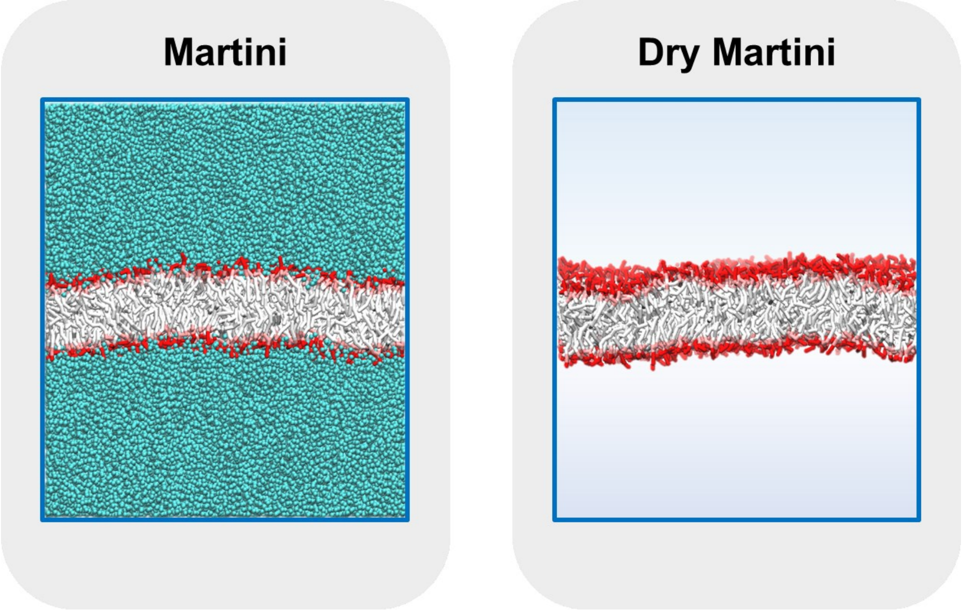

Figure 2.6: A small membrane in Martini and Dry Martini. Removing the solvent greatly reduces the computational burden of force evaluation.

2.6 Standard GROMACS Parameters for Selected Force Fields

Default simulation parameters used in this work are given in **Tables 2.1** to **2.3**. Unless otherwise indicated, assume these parameters for their relevant simulations.

Table 2.1: Standard parameters for Martini membrane simulations in GROMACS.

Integration	Velocity-Verlet with $\Delta t = 20$ fs Center of mass motion removed every timestep, treating membrane and solvent separately Long range forces calculated every step
Neighbor Lists	Group scheme, recalculated every 10 timesteps; Cutoff at 14 Å
Van der Waals	Potential shifted to zero over 9-12 Å.
Electrostatics	Electrostatic potential shifted to zero over 0-12 Å; No long-range electrostatics; Relative permittivity $\varepsilon_r = 15$

Thermostat	Bussi velocity rescaling, treating water and lipids separately; $T_0 = 323$ K, $\tau_T = 1$ ps

Barostat	Berendsen, semiisotropic coupling;
	$P_0 = 1$ bar, $\tau_P = 12$ ps
	Compressibility = 3.0×10^{-4} bar^{-1}

Table 2.2: Standard parameters for Dry Martini membrane simulations in GROMACS.

Integration	Stochastic dynamics with $\Delta t = 20$ fs Center of mass motion removed every timestep Long range forces calculated every step
Neighbor Lists	Group scheme, recalculated every 10 timesteps; Cutoff at 14 Å
Van der Waals	Potential shifted to zero over 9-12 Å.
Electrostatics	Electrostatic potential shifted to zero over 0-12 Å; No long-range electrostatics; Relative permittivity $\varepsilon_r = 15$
Thermostat	Provided by stochastic dynamics integrator; $\tau_T = 4$ ps

Barostat	Berendsen, semiisotropic coupling;
	$P_0 = 0$ bar, $\tau_P = 12$ ps
	Compressibility = 3.0×10^{-4} bar^{-1} (lateral, $= 0$ for normal)

Table 2.3: Standard parameters for CHARMM36 membrane simulations in GROMACS.

Integration	Velocity-Verlet with $\Delta t = 2$ fs Center of mass motion removed every 100 timesteps, treating membrane and solvent separately Long range forces calculated every step
Neighbor Lists	Verlet scheme, recalculated every 10 timesteps; Cutoff at 12 Å

Van der Waals	Force switched to zero over 8-12 Å
Electrostatics	PME used for long-range electrostatics
Bonds	All hydrogen bonds constrained with LINCS algorithm
Thermostat	Nosé-Hoover, treating water and lipids separately; $T_0 = 323$ K, $\tau_T = 1$ ps

Barostat	Parrinello-Rahman, semiisotropic coupling;
	$P_0 = 1$ bar, $\tau_P = 5$ ps
	compressibility = 4.5×10^{-5} bar^{-1}

2.7 Self-Diffusion Calculations

A brief aside is necessary to explain how lateral self-diffusion coefficients of lipids or membrane-associated proteins are calculated from MD simulations. These calculations involve a relatively simple analysis of MD trajectories. The diffusion coefficient for lateral diffusion may be calculated from the particle positions using the mean square displacement

$$MSD(\tau) = \langle [r(t+\tau) - r(t)]^2 \rangle_t \tag{2.43}$$

In two dimensions, this is related to the diffusion coefficient via

$$4D = \lim_{\tau \to \infty} \frac{MSD(\tau)}{\tau} \tag{2.44}$$

In practice, this means that we can compute $MSD(\tau)$ from the particle trajectories, averaged over all particles and all lag time pairs, and obtain *D* from a linear fit of $MSD(\tau)$ vs τ. Before doing this, the trajectories must be "unwrapped", that is, periodic boundary hopping must be removed.

Multiple fits are necessary for error estimation. This may be done by running several simulations in parallel or running a single long simulation and breaking the trajectory into segments. These are only equivalent if ergodicity holds for the process being studied. The fit should only be taken over the linear region of $MSD(\tau)$. In some membrane simulations, depending on resolution and lipid composition, a subdiffusive regime may be observed at short time scales, which should be omitted when computing long-time diffusion constants [115].

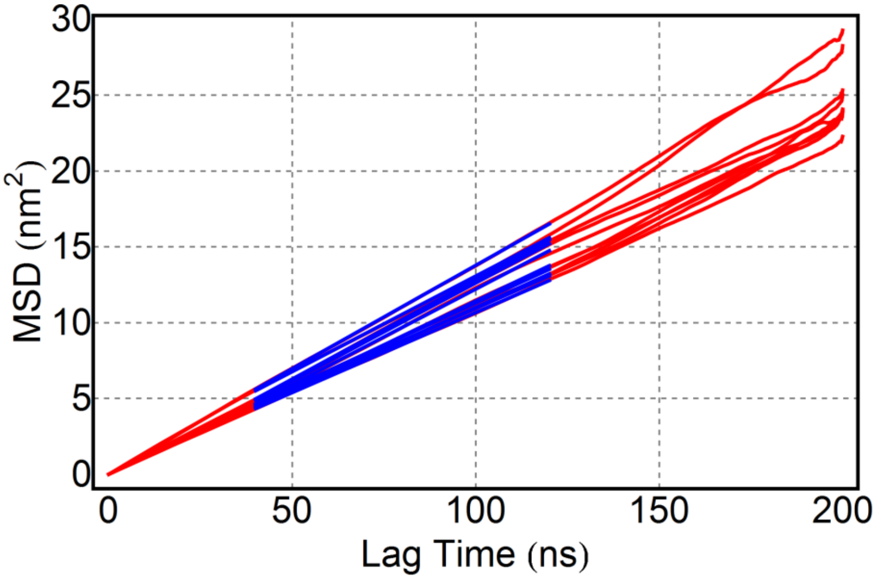

Figure 2.7: Example MSD fits from a single 2 μs membrane simulation broken into ten separate 200 ns sub-trajectories with independent fits to each. The blue lines indicate the linear fits along with the region of the data over which the fits were performed.

Chapter 3

MEMBRANE HYDRODYNAMICS

3.1 Introduction

Under physiological conditions, the plasma membrane is a quasi-2D viscous fluid composed mainly of lipids and proteins. The flow fields of this fluid and the resultant protein transport are confined to a 2D surface. However, these flows exchange momentum with the surrounding solvent, coupling the slow lateral flows of lipids in the membrane to much faster flows in the lower viscosity, bulk 3D solvent. Hydrodynamic interactions between proteins are mediated by both fluids, leading to an interesting interplay at different length scales. This is especially relevant in periodic systems (such as MD simulations) where long-distance interactions lead to self-interaction through the periodic image lattice.

This chapter will introduce and briefly discuss the hydrodynamic theory relevant to membranes and computer simulations of membranes. We start with general fluid mechanics and see how certain conservation laws produce the Navier-Stokes equation. Then we'll take a closer look at the limiting case of linear, low Reynolds number hydrodynamics that apply over the length scales of proteins and cells. We will see how the membrane geometry leads to a hybrid of 2D and 3D hydrodynamics and consider the impact of periodic boundary conditions on protein diffusion.

For additional information, many excellent resources exist on fluid dynamics [116], low Reynolds number hydrodynamics [117], [118], and membrane hydrodynamics [119], [120].

Table 3.1: Symbols used throughout this chapter.

For a general fluid:

$\vec{u}$	Fluid velocity field
P	Fluid pressure field
ρ	Fluid density field
$\sigma_{\alpha\beta}$	Viscous stress tensor
η	Fluid viscosity

For membrane hydrodynamics:

η_w	Solvent viscosity
η_m	Membrane surface viscosity
η_m^b	Effective 3D membrane viscosity
h	Membrane thickness
b	Membrane interleaflet friction

3.2 Navier-Stokes Hydrodynamics

Consider a homogenous, isothermal fluid. At time t, we can define its state using three fields: the scalar density $\rho(\vec{r},t)$, the scalar pressure $P(\vec{r},t)$, and the velocity vector $\vec{u}(\vec{r},t)$. These three quantities completely characterize the state of the fluid. We can use them to find equations of motion that describe its state at some later time t' for a particular set of boundary conditions. To derive those equations, we only need to assume that the fluid locally conserves mass and momentum.

3.2.1 Conservation of Mass

Demanding that mass be conserved locally at every point in the fluid results in a continuity equation. Conservation requires that the change in mass within an arbitrary volume is equal to the amount that flows through its surface. Consider an arbitrary volume V enclosing a fluid of uniform density ρ. The total change in mass within *V* is given by the total mass flux $\rho\vec{u}$ flowing through the volume's surface *S*. Defining $\hat{n}$ to be the normal vector pointing *out of* the volume, we may write

$$\frac{\partial}{\partial t}\left(\int_V \rho\, dV\right) = -\oint_S \rho\vec{u}\cdot\hat{n}\, dA \tag{3.1}$$

Apply the divergence theorem to convert the surface integral into a volume integral

$$\frac{\partial}{\partial t}\left(\int_V \rho\, dV\right) = -\int_V \nabla\cdot\rho\vec{u}\, dV \tag{3.2}$$

$$\int_V \left(\frac{\partial\rho}{\partial t} + \nabla\cdot\rho\vec{u}\right) dV = 0 \tag{3.3}$$

Since this holds over an arbitrary volume, the integrands must be equal to one another. This gives us the continuity equation

$$\frac{\partial\rho}{\partial t} = -\nabla\cdot(\rho\vec{u}) \tag{3.4}$$

In the special case of an incompressible fluid with uniform, fixed density, the continuity equation can be simplified

$$\nabla\cdot\vec{u} = 0 \tag{3.5}$$

3.2.2 Conservation of Momentum

Let’s now assume that momentum is locally conserved. The momentum of a fluid element is given by $\rho\vec{u}$. Since this is a vector, the momentum flux through the surface bounding this element must be written as a second-rank tensor $\mathbb{F}$. Local conservation of momentum is satisfied by

$$\frac{\partial}{\partial t}(\rho u_i) = -\partial_j \mathbb{F}_{ij} \tag{3.6}$$

This equation is sometimes called the ***Cauchy momentum equation***. It resembles the equation we found for conservation of mass. The interesting physics are contained within the momentum flux tensor. It can be split into two parts. One part arises from advection; fluid with momentum ρu_i being transported along the flow field, which can be written $\rho u_i u_j$. The remaining part is called the ***Cauchy stress tensor***, denoted Π_{ij}. It contains contributions from the surface and volume forces acting on the fluid element.

$$\frac{\partial}{\partial t}(\rho u_i) = -\partial_j\left(\rho u_i u_j\right) - \partial_j \Pi_{ij} \tag{3.7}$$

3.2.3 The Stress Tensor

We can get a better understanding of the Cauchy stress tensor by integrating over a volume and applying the divergence theorem

$$\frac{\partial}{\partial t}(\rho u_i) = -\partial_j\left(\rho u_i u_j\right) - \partial_j \Pi_{ij} \tag{3.8}$$

$$\frac{\partial}{\partial t}\int_V \rho u_i \, dV = -\int_V \partial_j\left(\rho u_i u_j\right) dV - \int_V \partial_j \Pi_{ij} \, dV \tag{3.9}$$

$$\frac{\partial}{\partial t}\int_V \rho u_i \, dV = -\oint_S \left(\rho u_i u_j\right) \hat{n}_j \, dA - \oint_S \Pi_{ij} \hat{n}_j \, dA \tag{3.10}$$

The first surface integral is from the advective term. It simply integrates the momentum carried through the surface. The second surface integral picks out the part of Π_{ij} along the surface. For a ***viscous fluid***, we can decompose the stress tensor into two contributions: an isotropic stress from the scalar pressure *P* and a shear stress $\sigma_{\alpha\beta}$ from viscosity.

$$\Pi_{ij} = P\delta_{ij} - \sigma_{ij} \tag{3.11}$$

This new tensor σ_{ij} is called the ***viscous stress tensor***. If we insert this expression into our equation for conservation of momentum, we find

$$\frac{\partial(\rho u_i)}{\partial t} = -\partial_j(\rho u_i u_j) - \partial_j(P\delta_{ij} - \sigma_{ij}) \tag{3.12}$$

$$\frac{\partial \rho}{\partial t}u_i + \rho\frac{\partial u_i}{\partial t} = -u_i\partial_j(\rho u_j) - (\rho u_j)\partial_j u_i - \partial_j P\delta_{ij} + \partial_j\sigma_{ij} \tag{3.13}$$

We can use the continuity equation to rewrite the first term on the RHS

$$\frac{\partial \rho}{\partial t}u_i + \rho\frac{\partial u_i}{\partial t} = u_i\frac{\partial \rho}{\partial t} - (\rho u_j)\partial_j u_i - \partial_j P + \partial_j\sigma_{ij} \tag{3.14}$$

$$\rho\frac{\partial u_i}{\partial t} + (\rho u_j)\partial_j u_i = -\partial_j P + \partial_j\sigma_{ij} \tag{3.15}$$

In vector notation, this reads

$$\rho\left(\frac{\partial \vec{u}}{\partial t} + (\vec{u}\cdot\nabla)\vec{u}\right) = -\nabla P + \nabla\cdot\sigma_{ij} \tag{3.16}$$

To further simplify, we need to make some assumptions about the fluid and how its viscous stress tensor is structured. Since viscous forces do not appear in quiescent fluids or in those under uniform flow, those forces must depend on velocity gradients. ***Newtonian fluids*** assume a linear relationship between derivatives of the velocity and the resulting shear stress. The modulus is called the viscosity, which in general is a rank-4 tensor with 81 components, $\sigma_{ij} = \eta_{ijkl}\partial_k u_l$.

Fortunately, the 81 components of a Newtonian viscous stress tensor are not independent. It can be shown that the stress tensor must be symmetric for angular momentum to be conserved. This leaves only two possible combinations involving first derivatives of the velocity: $\partial_i u_j + \partial_j u_i$ and $\delta_{ij}\partial_k u_k$. We can write the viscous stress tensor with just two viscosity coefficients

$$\sigma_{ij} = \eta\left(\partial_i u_j + \partial_j u_i - \frac{2}{3}\delta_{ij}\partial_k u_k\right) + \eta'(\delta_{ij}\partial_k u_k) \tag{3.17}$$

η is called the ***dynamic (or shear) viscosity*** and η' is called the ***second viscosity***. The extra term multiplying the dynamic viscosity has been added to cleanly separate isotropic contributions into the second viscosity only. This form ensures the stress tensor will vanish under uniform flow, $\vec{u}(\vec{r}) = a\hat{n}$, and under uniform rotation, $\vec{u}(\vec{r}) = \vec{\Omega} \times \vec{r}$ where $\partial_i u_j + \partial_j u_i = 0$.

3.2.4 The Navier-Stokes Equation

We are now ready to assemble the fluid equations of motion to arrive at the full Navier-Stokes equation. We begin by inserting our expression for the viscous stress tensor into the Cauchy momentum equation derived earlier

$$\rho\left(\frac{\partial u_i}{\partial t} + u_j \partial_j u_i\right) = -\partial_j P + \partial_j \left[\eta\left(\partial_i u_j + \partial_j u_i - \frac{2}{3}\delta_{ij}\partial_k u_k\right) + \eta'\left(\delta_{ij}\partial_k u_k\right)\right]$$

In general, the viscosity coefficients can depend on the spatial coordinates, but we'll assume that they do not (i.e., that the fluid is homogeneous) and to bring them outside the derivative.

$$\rho\left(\frac{\partial u_i}{\partial t} + u_j \partial_j u_i\right) = -\partial_j P + \eta\left(\partial_j \partial_i u_j + \partial_j \partial_j u_i - \frac{2}{3}\delta_{ij}\partial_j \partial_k u_k\right) + \eta'\left(\delta_{ij}\partial_j \partial_k u_k\right)$$

The Kronecker deltas vanish, and we can rewrite the dummy indices over k to run over j instead

$$\rho\left(\frac{\partial u_i}{\partial t} + u_j \partial_j u_i\right) = -\partial_j P + \eta\left(\partial_j \partial_i u_j + \partial_j \partial_j u_i - \frac{2}{3}\partial_i \partial_j u_j\right) + \eta'\left(\partial_i \partial_j u_j\right)$$

$$\rho\left(\frac{\partial u_i}{\partial t} + u_j \partial_j u_i\right) = -\partial_j P + \eta\, \partial_j \partial_i u_j + \eta\, \partial_j \partial_j u_i - \frac{2}{3}\eta\, \partial_i \partial_j u_j + \eta'\, \partial_i \partial_j u_j$$

$$\rho\left(\frac{\partial u_i}{\partial t} + u_j \partial_j u_i\right) = -\partial_j P + \eta\, \partial_j \partial_j u_i + \left(\eta' + \frac{1}{3}\eta\right)\, \partial_i \partial_j u_j$$

The first and second viscosities in the last term may be combined to form the ***bulk viscosity***, $\beta = \eta' + \eta/3$. We can also add a term for body forces to the RHS. This is the Navier-Stokes equation, which may be written in vector notation as

$$\rho\left(\frac{\partial\vec{u}}{\partial t}+(\vec{u}\cdot\nabla)\vec{u}\right)=-\nabla P+\eta\nabla^2\vec{u}+\beta\,\nabla(\nabla\cdot\vec{u})+\vec{f} \qquad (3.18)$$

In this form, it is clear that bulk viscosity is relevant only to compressible fluids, since $\nabla\cdot\vec{u}=0$ for an incompressible fluid. Bulk viscosity can be understood as an internal friction that resists flow under uniform compression or expansion. Even in compressible fluids, it is only relevant when dealing with rapid changes in volume. Thus, for incompressible fluids

$$\rho\left(\frac{\partial\vec{u}}{\partial t}+(\vec{u}\cdot\nabla)\vec{u}\right)=-\nabla P+\eta\nabla^2\vec{u}+\vec{f} \qquad (3.19)$$

Taken together with the continuity equation, this equation provides a general theoretical description of Newtonian fluid mechanics. A proof that smooth solutions exist for arbitrary boundary conditions is one of the great unsolved problems in mathematical physics. The difficulty arises from the nonlinearity of the advective term, which leads to turbulent behavior.

3.3 Low-Reynolds Number Hydrodynamics

Fluid dynamics on the length scale of cells, membranes, and proteins take on a considerably simpler form [121]. At this scale, the dissipative forces within the fluid are much larger than the inertial forces. Intuitively, this means that an object in motion will stay in motion only so long as a force is acting on it. When the force stops, the object immediately comes to rest. This limit is called the ***creeping flow*** or ***low Reynolds number*** regime and is governed by the Stokes equation

$$\eta\nabla^2\vec{u}-\nabla P=-\vec{f} \qquad (3.20)$$

Not only is this equation linear, which means there is no turbulence, but it also lacks explicit time dependence. Flows are instantaneously driven by pressure gradients, body forces, and boundary conditions. Solutions do not depend on the history of the fluid and are both unique and reversible (i.e., $-\vec{F}$ yields $-\vec{u}$).

To see why the terms on the left-hand side of the Navier-Stokes equation vanish in this limit, we may nondimensionalize it by introducing reduced units for distance $r' = r/L$ and velocity $u' = u/v$. From these, we can construct additional reduced units for time $t' = t\ (L/v)^{-1}$ and pressure $P' = P\ (\eta v/L)^{-1}$. Note that all terms in the Navier-Stokes equation have units of force/volume. Gathering the unit scaling factors from each term, we find

$$\left(\frac{\rho v^2}{L}\right)\left(\frac{\partial \vec{u}'}{\partial t'} + (\vec{u}' \cdot \vec{\nabla}')\vec{u}'\right) = \left(\frac{\eta v}{L^2}\right)\left(-\vec{\nabla}' P' + \nabla'^2 \vec{u}' - \vec{f}'\right) \qquad (3.21)$$

Dividing both sides by $\eta v/L^2$ gives an expression involving a dimensionless number $Re = \rho v L/\eta$ called the Reynolds number.

$$Re\left(\frac{\partial \vec{u}'}{\partial t'} + (\vec{u}' \cdot \vec{\nabla}')\vec{u}'\right) = -\vec{\nabla}' P' + \nabla'^2 \vec{u}' - \vec{f}' \qquad (3.22)$$

$$Re \equiv \frac{\rho v L}{\eta} \qquad (3.23)$$

The Reynolds number characterizes the ratio of viscous forces to inertial forces. The terms on the left-hand side can be neglected for $Re \ll 1$, leaving only the Stokes equation (3.20) on the right. Since $\rho/\eta \cong 1.25 \times 10^6$ s/m^2 for a water at 30°C, the product vL must be much smaller than 8×10^{-7} m^2/s to apply low Reynolds number hydrodynamics. Thus, the Stokes equation is relevant to both microscopic volumes of water moving at appreciable flow rates and slow-moving flows of far more viscous macroscopic fluids. The latter case illustrates why the low Reynolds number limit is sometimes called the creeping flow limit.

Since the Stokes equation is a linear differential equation, we can find the velocity field resulting from an arbitrary force distribution with a Green's function $\mathbb{T}(\vec{r})$ which satisfies

$$\vec{u}(\vec{r}) = \iiint \mathbb{T}(\vec{r} - \vec{r}') \cdot \vec{f}(\vec{r}')\, dV' \qquad (3.24)$$

There are several methods for calculating this Green's function using a point force source [122] such that

$$\vec{f}(\vec{r}) = \vec{F}\, \delta(\vec{r}) \qquad (3.25)$$

$$\vec{u}(\vec{r}) = \mathbb{T}(\vec{r}) \cdot \vec{F} \qquad (3.26)$$

The method suggested by Zapryanov and Tabakova [123] is particularly illuminating and relevant to systems with periodic boundary conditions. It begins by eliminating the velocity field by taking the divergence of Equation 3.20

$$\eta \nabla \cdot \nabla^2 \vec{u} - \nabla \cdot \nabla P = -\nabla \cdot \left(\vec{F}\, \delta(\vec{r})\right) \qquad (3.27)$$

$$\eta \nabla^2 (\nabla \cdot \vec{u}) - \nabla^2 P = -\vec{F} \cdot \nabla \delta(\vec{r}) \qquad (3.28)$$

$$\nabla^2 P = \vec{F} \cdot \nabla \delta(\vec{r}) \qquad (3.29)$$

Recall that $\nabla \cdot \vec{u} = 0$ for an incompressible fluid. We can solve for the pressure in reciprocal space by taking a spatial Fourier transform

$$k^2 P(\vec{k}) = -i\vec{k} \cdot \vec{F} \qquad (3.30)$$

$$P(\vec{k}) = -\frac{i\vec{k} \cdot \vec{F}}{k^2} \qquad (3.31)$$

Returning to the Stokes equation, we take the spatial Fourier transform and substitute the expression for $P(\vec{k})$ to find the velocity field in reciprocal space

$$\eta \nabla^2 \vec{u} - \nabla P = -\vec{f} \qquad (3.32)$$

$$-\eta k^2 \vec{u}(\vec{k}) - i\vec{k}\left(-\frac{i\vec{k}\cdot\vec{F}}{k^2}\right) = -\vec{F} \tag{3.33}$$

$$\vec{u}(\vec{k}) = \frac{1}{\eta k^2}\left[\vec{F} - \vec{k}\left(\frac{\vec{k}\cdot\vec{F}}{k^2}\right)\right] \tag{3.34}$$

$$\vec{u}(\vec{k}) = \frac{1}{\eta k^2}\left(\mathbb{I} - \frac{\vec{k}\vec{k}}{k^2}\right)\cdot\vec{F} \tag{3.35}$$

We may then find $\vec{u}(\vec{r})$ or $\mathbb{T}(\vec{r})$ with an inverse transform:

$$\vec{u}(\vec{r}) = \iiint \frac{d\vec{k}}{(2\pi)^3} e^{-i\vec{k}\cdot\vec{r}} \frac{1}{\eta k^2}\left(\mathbb{I} - \frac{\vec{k}\vec{k}}{k^2}\right)\cdot\vec{F} \tag{3.36}$$

$$\mathbb{T}_{\infty}^{3D}(\vec{r}) = \iiint \frac{d\vec{k}}{(2\pi)^3} e^{-i\vec{k}\cdot\vec{r}} \frac{1}{\eta k^2}\left(\mathbb{I} - \frac{\vec{k}\vec{k}}{k^2}\right) \tag{3.37}$$

This can be expressed in real space by assuming the velocity field vanishes as $r \to \infty$ to obtain the well-known ***Oseen tensor*** [117], [118], [124]

$$\mathbb{T}_{\infty}^{3D}(\vec{r}) = \frac{1}{8\pi\eta r}\left(\mathbb{I} + \frac{\vec{r}\vec{r}}{r^2}\right) \tag{3.38}$$

The Oseen tensor is the spatial response function connecting forces acting on a low Reynolds number fluid and the flow fields that result from them. The $1/r$ dependence leads to long-ranged interactions between bodies suspended in the fluid. These long-range interactions are problematic in molecular dynamics simulations due to necessity of using periodic boundary conditions [125].

The modified Green's function applicable to periodic systems is evident from Equation 3.37. We need only change the integral to a Fourier series ranging over all non-zero box vectors $\vec{k} = \langle n\pi/L_x\,, m\pi/L_y\,, \ell\pi/L_z\rangle$

$$\mathbb{T}_{pbc}^{3D}(\vec{r}) = \frac{1}{V}\sum\nolimits_{\vec{k}\neq 0} e^{-i\vec{k}\cdot\vec{r}} \frac{1}{\eta k^2}\left(\mathbb{I} - \frac{\vec{k}\vec{k}}{k^2}\right) \tag{3.39}$$

This differs from the infinite-system case by cutting off the small wavenumber, and hence long wavelength, modes. To understand the consequences of this, consider a point force at the origin directed along the z-axis

$$\vec{f}(\vec{r}) = \delta(0) F_0 \hat{z} \tag{3.40}$$

The resulting velocity field for the infinite case is

$$\vec{u}_\infty(\vec{r}) = \frac{F_0}{8\pi\eta r}\left(\hat{z} + \hat{r}\frac{r_z}{r}\right) \tag{3.41}$$

Note that this equation is radially symmetric about the z-axis. The direction of the velocity field is always oriented in the sense of the applied force, that is $\vec{u}_\infty \cdot \hat{z} > 0$ for all $\vec{r}$. Consequently, streamlines for the flow field move in the direction of the applied point force and never circulate back around. The uniform field dominates as $r \to \infty$ and its magnitude decays as $1/r$. Streamlines for this field are plotted in **Fig. 3.1**.

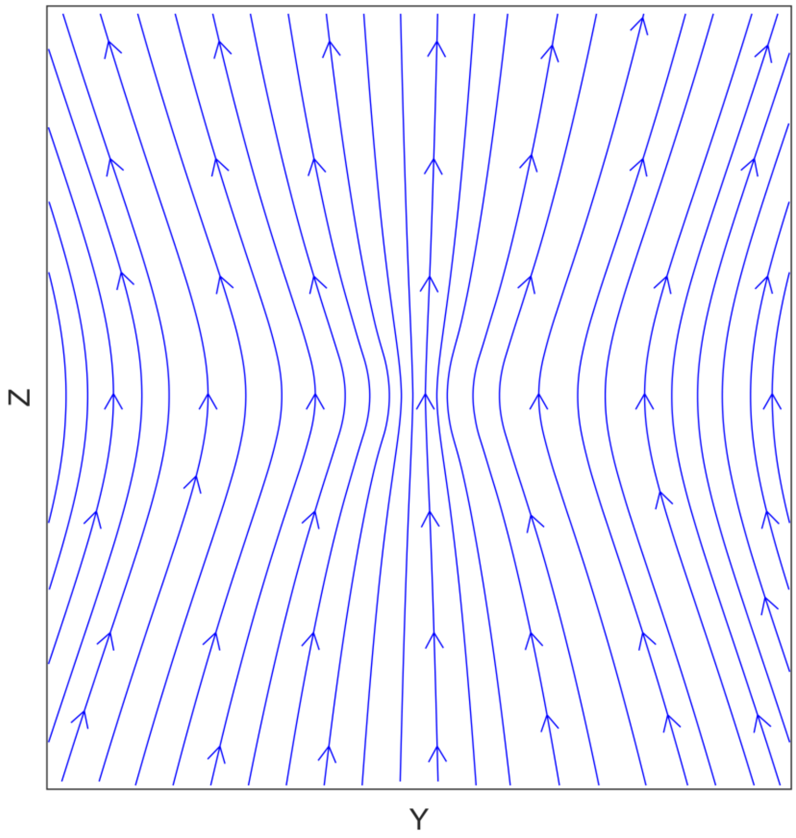

Figure 3.1: Streamlines for the velocity field resulting from a point force at the origin of an infinite 3D fluid. The velocity field is radially symmetric about the z-axis, but the streamlines never recirculate. Tickmarks are not displayed because these streamlines are identical at all length scales (although the *magnitude* does fall off).

The velocity field resulting from a point force is strikingly different for the periodic case (see **Fig. 3.2**). To satisfy the periodic boundary conditions, the fluid recirculates in a toroidal flow pattern with streamlines close to the z-axis resembling those in the infinite case. This recirculating vortex pattern appears in the same way for periodic boxes of any size. The way it scales with the periodic box suggests it arises from the cutoff length imposed by the size of the periodic box.

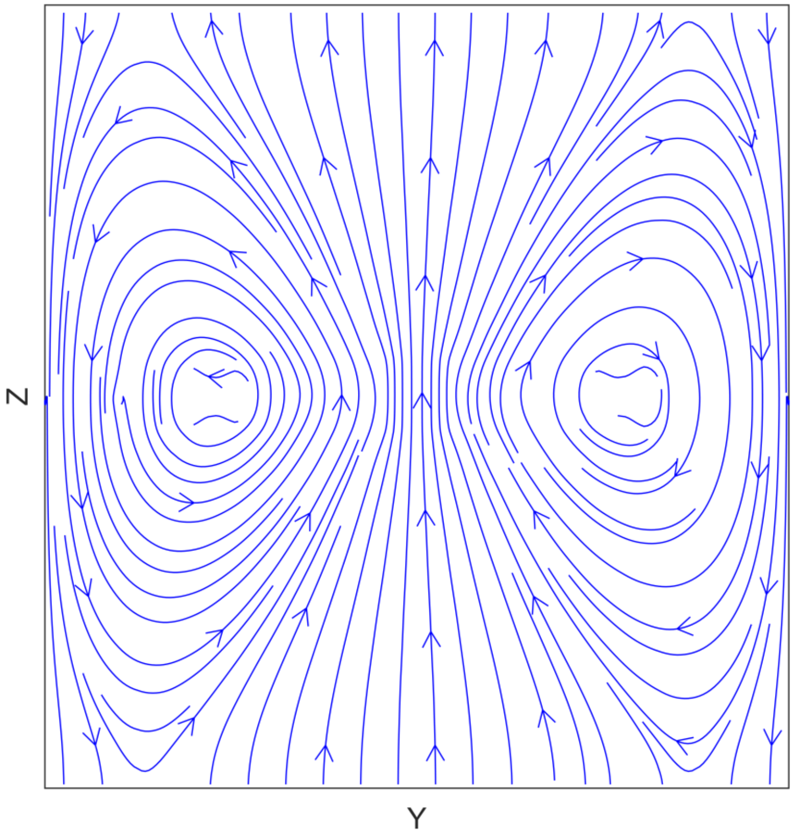

Figure 3.2: Streamlines for the velocity field resulting from a point force at the origin of a 3D fluid in a periodic box. Tickmarks are not displayed because these streamlines are identical at all length scales.

Should we expect fluid flows in molecular dynamics simulations to follow predictions from low Reynolds number hydrodynamics theory? The theoretical flow fields are derived from continuum models operating at length scales much larger than individual fluid molecules. Even an infinitesimal fluid parcel is assumed to contain a molecular multitude. In contrast, even large molecular dynamics simulations contain only a few million molecules. One way to check the applicability of low Reynolds number hydrodynamics theory to MD simulation is to add a steady point force to the center of the periodic box and check whether the time-averaged, steady-state response follows the flow field predicted by the periodic Oseen tensor (Equation 3.39).

To test this proposition, a 30 nm cubic box containing roughly 167,000 Martini water beads was prepared and equilibrated to 1 bar pressure at 323 K using standard Martini MD parameters. The point force was crudely represented by a constant force applied at every timestep to all particles within a spherical volume at center of the simulation box. The position and velocity for each particle were recorded every 10 ps for 200 ns (20,000 frames). The average velocity field for each frame was calculated on a discretized spatial grid of 25 x 25 x 25 points, with the velocity at each point determined by a Gaussian-weighted average of the nearest particle velocities using their distance to each grid point $\vec{g}$. The weights were selected according to

$$w_i = \exp\left[-\frac{1}{2}\left(\frac{\vec{r}_i - \vec{g}}{\Delta g/4}\right)^2\right] \tag{3.42}$$

with standard deviation equal to the average distance between grid points Δg divided by 4. A cutoff distance equal to Δg was imposed, with no particles contributing to the average for each point from beyond the cutoff. The grids for each frame were averaged together for all 20,000 frames.

Streamlines for the resulting velocity fields are depicted in **Figs. 3.3** and **3.4**, demonstrating the necessity of a relatively strong point force to cleanly resolve the flow field. The clean agreement with theory suggests low Reynolds number hydrodynamic theory is an adequate description of the fluid dynamics in MD simulation.

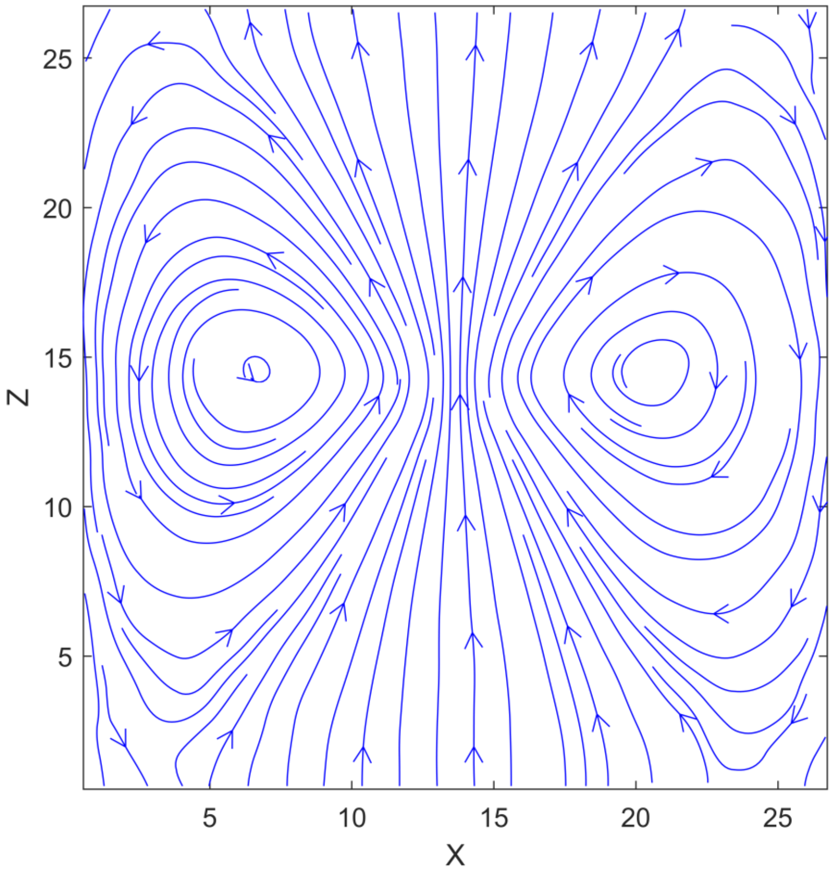

Figure 3.3: Streamlines for the average flow produced by a 32.0 kJ/(mol-nm) point force applied every 20 fs to all particles within a spherical region at the center of the simulation box with r = 1.0 nm. The fluid consists of Martini water beads. The flow pattern closely matches the theory for low Reynolds number fluids.

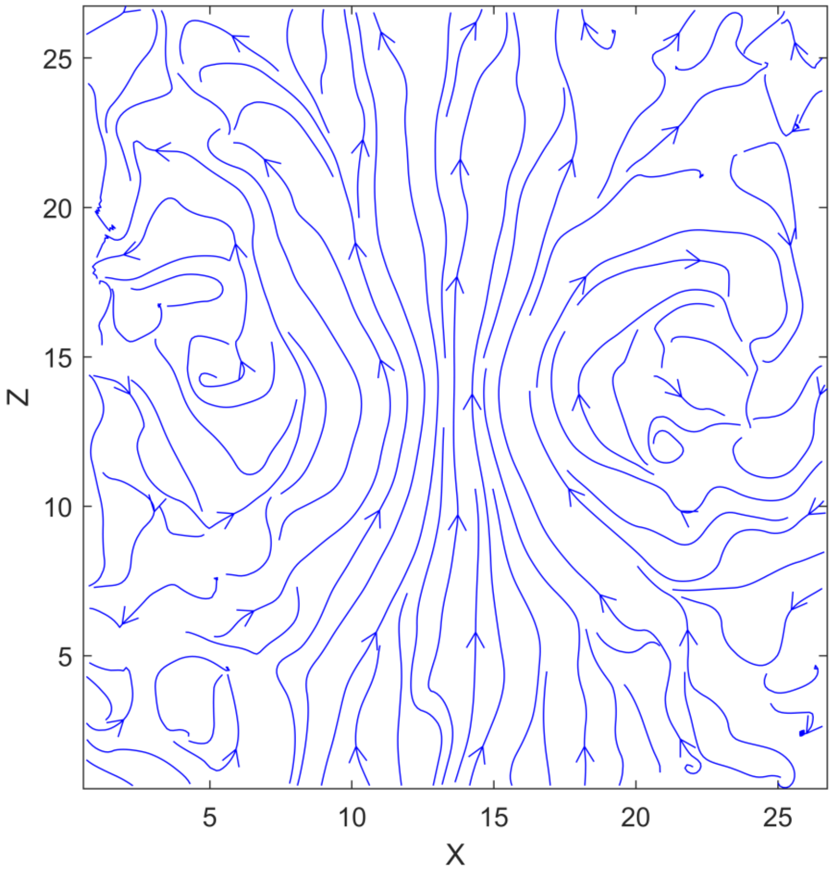

Figure 3.4: Streamlines for the average flow produced by a 2.0 kJ/(mol-nm) point force applied every 20 fs to all particles within a spherical region at the center of the simulation box with r = 1.0 nm. The fluid consists of Martini water beads. Using a weaker point force results in a much nosier flow field.

3.4 Quasi-2D Membrane Hydrodynamics

Suppose we want to solve for the mobility of a protein embedded in a membrane. At physiological temperatures, the membrane can be treated as a 2D viscous fluid. This requires solving for the drag force exerted by the fluid as it moves through the fluid. If it were a sphere of radius a moving in an unbounded 3D fluid at rest far away from the sphere, this problem has a simple, well-known solution obtained by solving the Stokes equation [117]

$$F = 6\pi\eta a V$$

where F is the viscous drag force exerted by the fluid when the body moves through it with constant velocity V. However, there is no analogous solution for a disk moving in an unbounded 2D fluid. This is the “Stokes paradox”.

We can see the problem with dimensional analysis. In three dimensions, the fluid exerts a force on the sphere, but in two dimensions it exerts a force per unit length. The only parameters of the model are the radius and velocity of the disk and the viscosity of the fluid. The only combination of these parameters that yields a force per unit length is ηV, which does not involve radius. In such a scenario, a disk of vanishing size will feel the same force/length as a large one, clearly nonsense. Another length scale L is needed to resolve the paradox. If we restrict our attention to 2D fluids of finite extent, this new length scale arises from the system size (as essentially a cutoff length). A 2D Oseen tensor can be written with this approach [118], [126]

$$\mathbb{T}^{2D}_{\text{finite}}(\vec{r}) = \frac{1}{4\pi\eta}\left[-\ln\left(\frac{r}{L}\right)\mathbb{I} + \frac{\vec{r}\vec{r}}{r^2}\right] \tag{3.43}$$

Notice that it scales with $\ln(r)$ rather than $1/r$, diverging as $r \to \infty$. In an unbounded fluid, this divergence is the source of the Stokes paradox.

To see how this comes about, consider how the 2D Oseen tensor differs from the 3D case. The derivation in 2D closely follows the 3D case outlined above, arriving at an identical expression involving 2D vectors and a 2D Fourier transform. The problem arises when we try to integrate this expression

$$\mathbb{T}^{2D}_{\infty}(\vec{r}) = \iint \frac{d\vec{k}}{(2\pi)^2} e^{-i\vec{k}\cdot\vec{r}} \frac{1}{\eta k^2}\left(\mathbb{I} - \frac{\vec{k}\vec{k}}{k^2}\right) \tag{3.44}$$

$$
\begin{aligned}
&= \iint \frac{k\, dk\, d\theta}{(2\pi)^2} e^{-i\vec{k}\cdot\vec{r}} \frac{1}{\eta k^2}\left(\mathbb{I} - \frac{\vec{k}\vec{k}}{k^2}\right) \\
&= \frac{1}{2\pi\eta}\int_0^\infty dk\, e^{-i\vec{k}\cdot\vec{r}} \frac{1}{k}\left(\mathbb{I} - \frac{\vec{k}\vec{k}}{k^2}\right)
\end{aligned}
$$

This differs from the 3D case due to the incomplete cancellation of $1/k$, yielding an integral that diverges for small k. This arises from the 2D Fourier transform being a surface integral rather than a volume integral. The divergence can be avoided by setting a finite lower limit of the integral, i.e. imposing a cutoff length.

In the case of the cell membrane, the Stokes paradox is avoided because flows in the plane couple to the surrounding bulk solvent in 3D. This was first recognized by Saffman and Delbrück (SD) [7], who considered a membrane protein as a cylinder of radius a spanning an infinite uniform membrane of thickness h and surface viscosity η_m. Flows in the membrane are coupled to an infinite bulk fluid surrounding the membrane with viscosity η_w through stick boundary conditions (see **Fig. 3.5**). These boundary conditions constrain the lateral velocity fields to be equal at the interface. Lateral flows in the membrane remain 2D, but the solvent mediates additional long-distance interactions which impart some characteristics of 3D hydrodynamics on longer length scales. For this reason, SD hydrodynamics are called quasi-2D.

Solving the Stokes' equation for this geometry results in a well-known tensor

$$
\mathbb{T}_\infty^{SD}(\vec{k}) = \frac{1}{\eta_m(k^2 + k/L_{SD})}\left(\mathbb{I} - \frac{\vec{k}\vec{k}}{k^2}\right) \tag{3.45}
$$

Featuring a new length scale called the Saffman-Delbrück length, defined as

$$
L_{SD} = \frac{\eta_m}{2\eta_w} \tag{3.46}
$$

With η_m being the surface viscosity of the membrane, equal to its effective 3D viscosity η_m^b multiplied by h, and η_w being the viscosity of the solvent. For typical membranes, this length ranges from 100-1000 nm depending on lipid composition, much larger than the radii of typical protein transmembrane domains. Note the denominator of equation 3.45 can also be written $\eta_m k^2 + 2\eta_w k$.

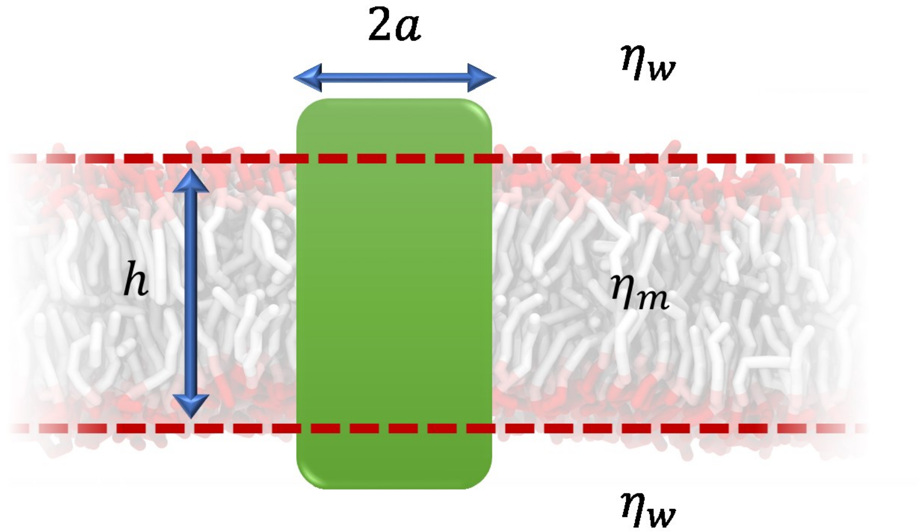

Figure 3.5: Saffman-Delbrück theory models a protein as a cylinder of radius a spanning a uniform membrane of thickness h and viscosity η_m. The infinite solvent surrounding the membrane has viscosity η_w and is coupled to the membrane with stick boundary conditions.

We can gather physical intuition for quasi-2D hydrodynamics by considering limiting cases of equation 3.45 (keeping in mind that it is a 2D tensor). First, consider the $k \ll {L_{SD}}^{-1}$ case: the first term in the denominator vanishes and the expression scales as $(\eta_w k)^{-1}$. This is the 2D transform of $1/r$, indicating that long-wavelength interactions are mediated by the solvent and have a similar form to the 3D Oseen tensor. Second, consider the $k \gg {L_{SD}}^{-1}$ case: now the second term in the denominator vanishes and the response scales as $(\eta_m k)^{-2}$. The short-wavelength interactions occur through the membrane. Just like the simple 2D fluid, this form diverges as $k \to 0$ when we take the Fourier transform. However, since we are considering the limit of $k \gg L_{SD}{}^{-1}$, we may naturally interpret L_{SD} as a cutoff length similar to the one employed in equation 3.43. Thus, equation 3.45 tells us that quasi-2D membrane hydrodynamics behave as a 2D fluid over distances smaller than L_{SD} (due to the membrane) and as a 3D fluid over distances longer than L_{SD} (due to the solvent).

For proteins with radii $a \ll L_{SD}$, Saffman-Delbrück theory allows us to calculate a self-diffusion coefficient

$$D = \frac{k_B T}{4\pi\eta_m}\left[\ln\left({}^{2L_{SD}}/_{a}\right) - \gamma\right] \tag{3.47}$$

Where γ is the Euler–Mascheroni constant ~ 0.57721 and $k_B T$ is the thermal energy. Interestingly, SD theory predicts that lateral protein mobility is only weakly dependent on the size of the protein's transmembrane domain, scaling as $\ln(1/a)$ rather than the $1/a$ scaling found in bulk diffusion. SD theory has been successfully employed to interpret experimental diffusion measurements of both proteins [127] and lipid domains [128]. However, others have found $1/a$ dependence [129] which may be explained by local membrane deformations imposed by the protein [130]. Such deformations scale the local membrane viscosity by $1/a$, recovering SD theory.

3.5 Periodic Membrane Hydrodynamics

Since its initial publication in 1975 [7], SD theory has been extended to account for larger proteins [31], to situations where the membrane is supported above a rigid surface [131], [132] or pinned with immobile inclusions [133], and others. Of particular interest to MD simulation is the recent work by Camley et al. [44], [46] which extend SD theory to systems with periodic boundaries, termed ***periodic***

Saffman-Delbrück (PSD) theory. This theory predicts a significant finite size effect on diffusion coefficients calculated from simulation due to the interplay between the long-range nature of SD hydrodynamic interactions and the length scale cutoff imposed by the boundary conditions. Subsequent simulations investigating this finite size effect have supported this theory [45].

PSD theory considers a periodic membrane patch of lateral size *L* surrounded by a solvent of depth H (see **Fig. 3.6**, note that H refers to the distance from the solvent-membrane interface to the periodic boundary, not the distance from one interface to the next). This theory differs from the infinite case by (1) requiring solvent flows to match along the vertical boundaries at $z = 0$ and $z = H$ and (2) changing the Fourier transform which converts $\mathbb{T}(\vec{k})$ to real space into a Fourier series with the smallest *k* vector corresponding to the cutoff length imposed by the box size *L*. The resulting response tensor in reciprocal space is remarkably similar to the infinite case (equation 3.45) with the only difference found in the solvent term of the denominator

$$\mathbb{T}_{pbc}^{SD}(\vec{k}) = \frac{1}{\eta_m k^2 + 2\eta_w k \tanh(kH)}\left(\mathbb{I} - \frac{\vec{k}\vec{k}}{k^2}\right) \qquad (3.48)$$

An interesting consequence of this difference arises in the limit $H \ll L$, where $\tanh(kH) \approx kH$ and the denominator becomes $k^2(h\eta_m^b + 2H\eta_w)$. The system is reduced to a 2D fluid with an effective viscosity given by the sum of membrane and solvent surface viscosities.

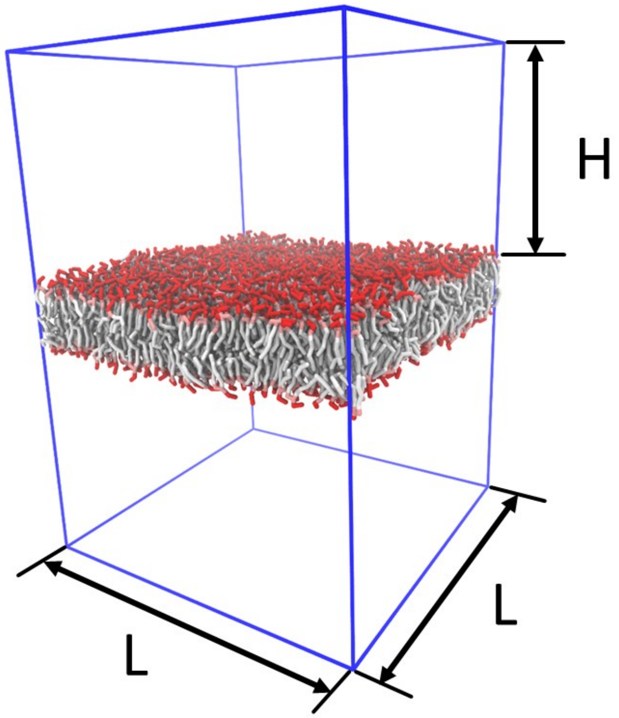

Figure 3.6: A simulation snapshot with dimensions of the periodic cell referenced by Periodic Saffman-Delbrück theory (solvent particles omitted for clarity).

Once again, we can gather some physical insight into this tensor by plotting the streamlines resulting from a point force (**Figs. 3.7-3.9**). As in the 3D case, recirculating flows emerge when periodic boundaries are used. Unlike the 3D case, the infinite-size PSD tensor is expressed as an integral over all 2D wavevectors. In practice, this requires numerical integration over a finite range of k_x and k_y values. Properly plotting the $\vec{v}_\infty(\vec{r})$ far-field requires sufficient resolution and range for the integration. Incomplete integration will cause the appearance of recirculating currents, demonstrating how the cutoff of long-wavelength modes in the periodic boundary case causes the formation of recirculating flows, with the cutoff length dictating their characteristic size.

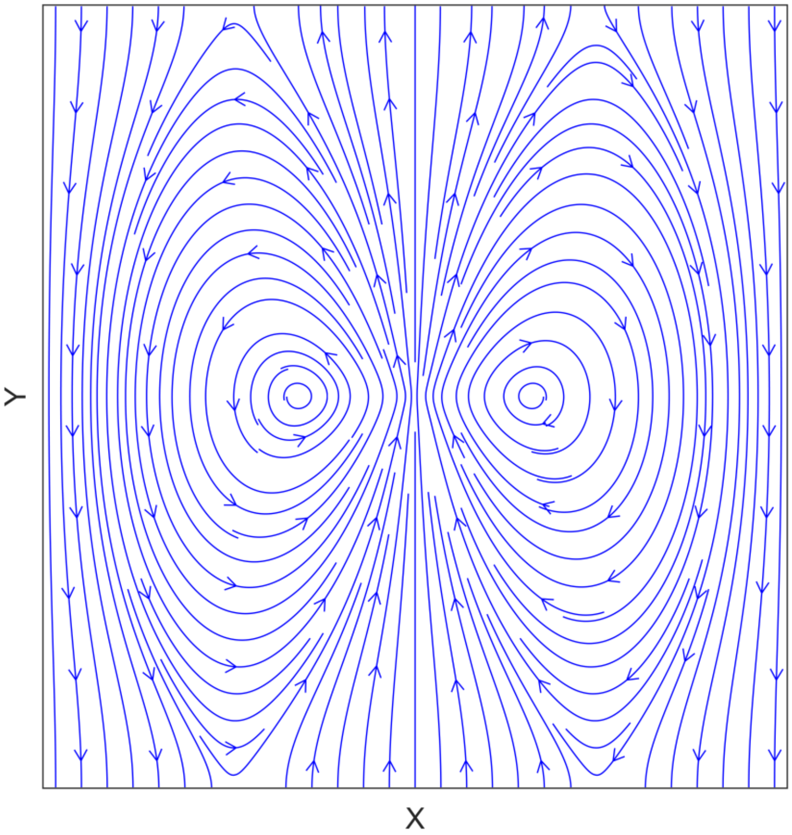

Figure 3.7: Streamlines for the 2D velocity field resulting from a point force at the center of a 10 nm membrane in a periodic box based on periodic Saffman-Delbrück theory. Tickmarks are not displayed because these streamlines are identical at all length scales.

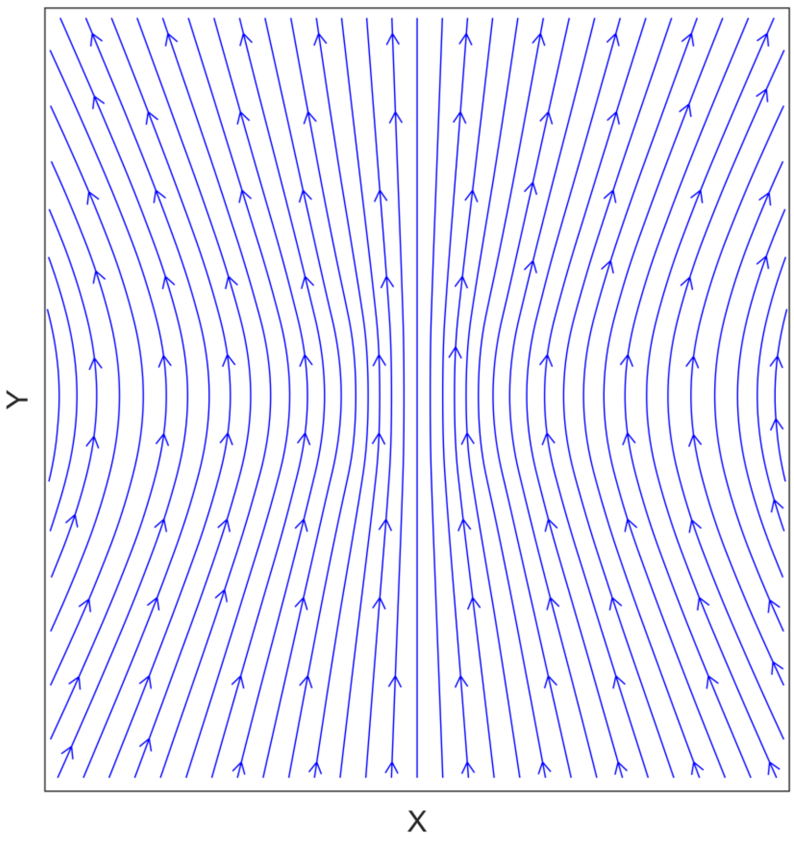

Figure 3.8: Streamlines for the 2D velocity field resulting from a point force at the origin of a membrane in the infinite-size limit of periodic Saffman-Delbrück theory. The integration has sufficient resolution to model the far-field flows.

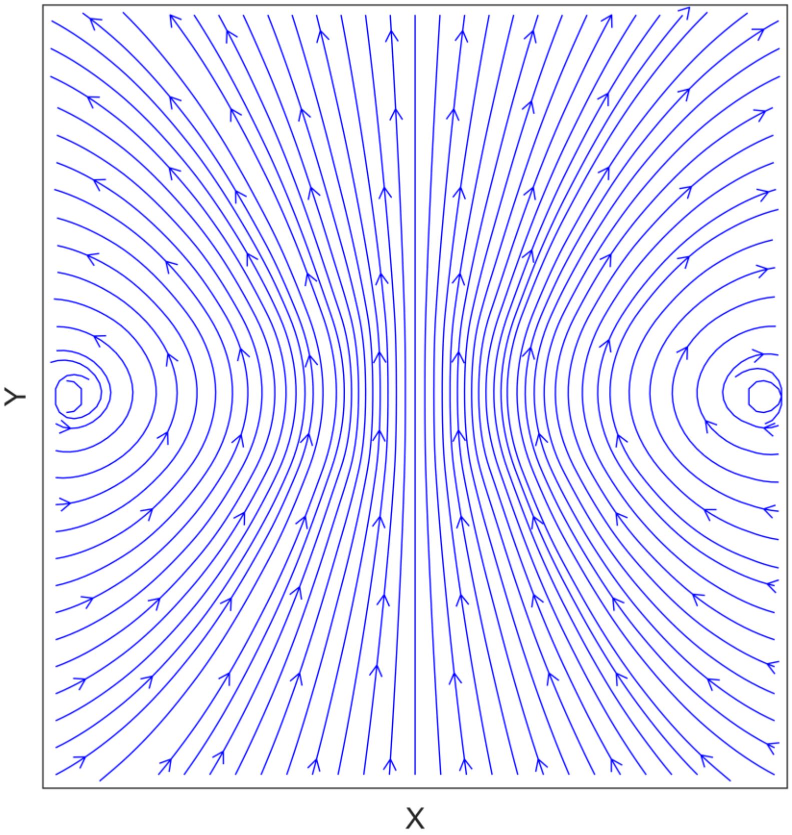

Figure 3.9: Streamlines for the 2D velocity field resulting from a point force at the origin of a membrane in the infinite-size limit of periodic Saffman-Delbrück theory. The integration step has insufficient resolution, leading to recirculating flows in the far field.

Analytic solutions for protein mobility based on equation 3.48 using no-slip boundary conditions are unknown and certainly complicated if they exist. We can find an approximate solution by modeling the protein as a fluid region in the membrane using the "immersed boundary method" [134], [135]. Applying a Gaussian force distribution, we can compute the mobility and use the Einstein relation to find the self-diffusion coefficient

$$D_{pbc}^{PSD} = \frac{k_B T}{2L^2} \sum_{k \neq 0} \frac{1}{\eta_m k^2 + 2\eta_w k \tanh(kH)} \exp\left(-\frac{(k\beta a)^2}{2}\right) \tag{3.49}$$

With the parameter $\beta = 0.828494$ used to approximate results from the Hughes-Pailthorpe-White [31] (HPW) extension to larger protein radii. The expression for an infinite system can be obtained by taking the limit of infinite L and H

$$D_{\infty}^{PSD} = \frac{k_B T}{2} \iint \frac{d\vec{k}}{(2\pi)^2} \frac{1}{\eta_m k^2 + 2\eta_w k} \exp\left(-\frac{(k\beta a)^2}{2}\right) \tag{3.50}$$

The authors also derive an expression for diffusion of a half-cylinder, i.e. an object spanning only one leaflet of the membrane

$$D_{PBC} = \frac{k_B T}{2L^2} \sum_{k \neq 0} \frac{A(k)}{A(k)^2 + B(k)^2} \exp\left(-\frac{(k\beta a)^2}{2}\right) \tag{3.51}$$

$$A(k) = b + \eta_w k \coth(2kH) + \frac{\eta_m}{2} k^2$$

$$B(k) = b + \eta_w k \operatorname{csch}(2Hk)$$

Which now includes a second parameter b called the interleaflet friction. For two leaflets gradually and continuously sliding past one another, this relates the friction force on the leaflets to their relative velocity

$$F = b\Delta v \tag{3.52}$$

The finite-size error predicted from this theory is plotted in **Fig. 3.10**.

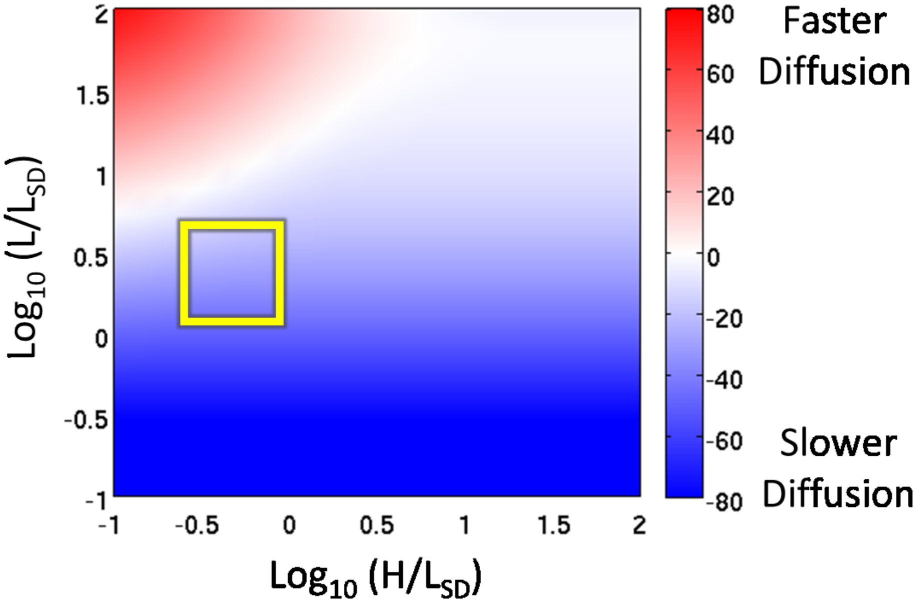

Figure 3.10: Finite-size percent error for diffusion in periodic membranes. The yellow box indicates the range of system sizes typically accessible for coarse-grain simulations.

The ratio of periodic (equation 3.49) and infinite (equation 3.50) expressions for self-diffusion can be used to compute a correction factor which can be applied to diffusion calculations from simulation to correct for the finite size effect and compare directly compare with experimental results. This requires a priori knowledge of the membrane surface viscosity for the lipid and force field used in the simulation, for which few results have been reported in the literature. This situation is rectified in Chapter 7.

Chapter 4

MULTI-PARTICLE COLLISION DYNAMICS

4.1 Introduction

Multi-particle collision (MPC) dynamics are a class of efficient, particle-based mesoscopic hydrodynamic simulation methods [136]–[139]. The fluid flow field is modeled with tracer particles which do not have mutual pairwise interactions. Instead, the transfer of momentum through collisions between MPC particles is coarse-grained into collective ***collision events,*** whereby all MPC particles in a local region exchange momentum simultaneously, modeling the cumulative outcome of many microscopic inter-particle collisions. Collision events are designed to conserve mesoscopic hydrodynamic flows through local conservation of both mass and momentum – the same conservation laws used to derive the Navier-Stokes equations in Chapter 3. Consequently, the mesoscopic flows modeled by the MPC particles reproduce Navier-Stokes hydrodynamics above a certain length scale.

Since forces between MPC particles are not needed, this method is inherently more efficient than either treating the solvent as an explicit molecular fluid or employing dissipative particle dynamics (DPD) [140]. Since MPC is particle-based, thermal fluctuations within the fluid arise naturally and the tracer particles cleanly interface with existing MD code and analysis tools. Just like other MD particles, MPC particles exist in continuous space with discretized time. Approaches confined to a discrete spatial lattice [141], [142] or those which rely on the explicit calculation of hydrodynamic interaction tensors are considerably more complicated to implement in an existing MD program [143], [144].

The dynamics of MPC particles are divided into two alternating steps. Between collective collisions, MPC particles “stream” along linear trajectories by simply integrating their velocities. At fixed intervals in time, called ***collision steps***, collision events are performed. Collision events randomize the relative velocity of each MPC particle with respect to the local flow velocity. At the collision step, the simulation volume is divided into a regular lattice of cubic cells called ***collision cells***. MPC particles within each cell participate in collective collisions together. The average MPC particle velocity within each cell is regarded as the mesoscopic velocity of that cell. The set of all cell velocity vectors forms the MPC fluid flow field, resolved on a length scale slightly larger than the cell size.

In addition to preserving the average velocity of each collision cell, the collision step must also conserve mass and momentum within each collision cell to reproduce Navier-Stokes hydrodynamics. Aside from these constraints, there is considerable freedom in how new velocities are chosen during the collision step. Since MPC collisions operate on the cell-relative velocities of MPC particles, we can write the general prescription for computing new velocities $\vec{v}_i'$ for particle *i* in a particular collision cell *c* with

$$\tilde{v}_i = \vec{v}_i - \vec{u}_c \tag{4.1}$$

$$\vec{v}_i' = \vec{u}_c + \tilde{v}_i' \tag{4.2}$$

where $\vec{u}_c$ is mean velocity of all MPC particles within cell *c*. Tilda accents denote cell-relative quantities and primes denote post-collision quantities. The transformation mapping $\tilde{v}_i \to \tilde{v}_i'$ may depend on both relative positions and velocities of all MPC particles in the cell. The only constraint on this operation is that the new relative velocities must sum to zero in order to satisfy conservation of momentum. Its particular form is called the ***collision rule***. Two widely used collision rules are MPC-Andersen (MPC-A) [145] and stochastic rotation dynamics (SRD) [136].

4.2 Collision Rule Notation

The following notation will be used throughout the next two sections, with subscript c denoting the cth cell and subscript i denoting the ith particle within a cell. Tilda accents denote relative quantities and primes denote post-collision quantities.

N_c	number of particles in cell
$\vec{u}_c$	mean cell velocity
$\vec{r}_c$	mean cell position
$\vec{v}_i$	absolute particle velocity
$\vec{r}_i$	absolute particle position
$\tilde{v}_i$	relative particle velocity (relative to $\vec{u}_c$)
$\tilde{r}_i$	relative particle position (relative to $\vec{r}_c$)
$\vec{v}_i'$	new particle velocity (after collision step)
$\tilde{v}_i'$	new relative particle velocity (after collision step)

Cell mean vectors are defined

$$\vec{r}_c = \frac{1}{N_c}\sum_{i=1}^{N_c} \vec{r}_i \qquad\qquad \vec{u}_c = \frac{1}{N_c}\sum_{i=1}^{N_c} \vec{v}_i$$

Relative particle vectors are defined

$$\tilde{r}_i = \vec{r}_i - \vec{r}_c \qquad\qquad \tilde{v}_i = \vec{v}_i - \vec{u}_c$$

4.3 Stochastic Rotation Dynamics (SRD)

4.3.1 Collision Rule

The stochastic rotation dynamics collision rule chooses new velocities by rotating the existing relative velocity vectors $\tilde{v}_i$ by a fixed angle α about a randomly chosen axis on the unit sphere $\hat{a}_c$

$$\vec{v}_i' = \vec{u}_c + R_{\alpha,\hat{a}_c}(\tilde{v}_i) \tag{4.3}$$

The rotation axis is sampled uniformly from the unit sphere with a different axis chosen for each cell. This collision rule conserves energy and momentum. It may be expressed without a rotation matrix as

$$\vec{v}_i' = \vec{u}_c + \tilde{v}_i^{\parallel} + \tilde{v}_i^{\perp}\cos(\alpha) + (\hat{a}_c \times \tilde{v}_i)\sin(\alpha) \tag{4.4}$$

$$\tilde{v}_i^{\parallel} = (\hat{a}_c \cdot \tilde{v}_i)\hat{a}_c \tag{4.5}$$

$$\tilde{v}_i^{\perp} = \tilde{v}_i - \tilde{v}_i^{\parallel} \tag{4.6}$$

Axes are chosen independently for each collision cell at every collision step. All colliding particles within a cell use the same axis. Since this collision step conserves energy exactly, MPC dynamics using the SRD collision rule do not require a thermostat. A thermostat can be applied, provided that it does not disrupt the mesoscopic flow field – it must only operate on the relative velocities [146].

4.3.2 Energy Conservation

An interesting, well-known property of the SRD collision rule is that it locally conserves the kinetic energy of MPC particles. This can be shown by examining the change in kinetic energy of the MPC particles within a given cell before and after a collision takes place

$$\frac{2}{m}\Delta E_{kin} = \sum \vec{v}_i' \cdot \vec{v}_i' - \sum \vec{v}_i \cdot \vec{v}_i \tag{4.7}$$

$$= \sum (\vec{u}_c + \tilde{v}_i') \cdot (\vec{u}_c + \tilde{v}_i') - \sum (\vec{u}_c + \tilde{v}_i) \cdot (\vec{u}_c + \tilde{v}_i) \tag{4.8}$$

$$= \sum \left[{u_c}^2 + 2(\vec{u}_c \cdot \tilde{v}_i') + \tilde{v}_i'^2\right] - \sum \left[{u_c}^2 + 2(\vec{u}_c \cdot \tilde{v}_i) + {\tilde{v}_i}^2\right] \tag{4.9}$$

Summing each term independently, we immediately see that the ${u_c}^2$ sums cancel and the dot product in the cross terms may be brought outside the sum

$$= \left[2\vec{u}_c \cdot \sum \tilde{v}_i' + \sum \tilde{v}_i'^2\right] - \left[2\vec{u}_c \cdot \sum \tilde{v}_i + \sum {\tilde{v}_i}^2\right] \tag{4.10}$$

The cross terms vanish because the relative velocities sum to zero (by definition), leaving only the sums over relative velocity squared

$$\frac{2}{m}\Delta E_{kin} = \sum \tilde{v}_i'^2 - \sum {\tilde{v}_i}^2 = 0 \tag{4.11}$$

These sums must also cancel since $\tilde{v}_i' = R_{\alpha,\hat{a}_c}(\tilde{v}_i)$ and the length of each vector does not change under rotation. Hence, the change in kinetic energy is zero. The reason this works for SRD and not necessarily for other collision rules arises from this final step. Other MPC collision rules do not conserve energy if they change the length of the relative velocity vectors, even though they must still sum to zero.

4.3.3 Parameters and Transport

The SRD collision rule features five adjustable collision parameters – four general to MPC methods and one particular to SRD. These parameters are listed in **Table 4.1**. Only the collision angle is particular to SRD. Transport properties of SRD fluids are controlled by adjusting these simulation parameters. When running MPC fluid simulations, collision parameters should be chosen that produce a reasonable set of dimensionless hydrodynamic numbers and physical fluid properties [137].

Table 4.1: SRD collision parameters.

Symbol	**Description**
m	MPC particle mass
M	MPC particles per collision cell
a	Collision cell size
τ_c	Collision interval / frequency
α	SRD rotation angle

The simplicity of the SRD collision step admits analytic solutions for its transport properties as a function of the collision parameters. Both streaming and collision steps contribute to the shear viscosity [138]

$$\eta_{SRD} = \left(\frac{mM}{a^3}\right)(\upsilon_{col} + \upsilon_{str}) \tag{4.12}$$

$$\upsilon_{col} = \left(\frac{a^2}{\tau_c}\right)\left(\frac{M - 1 + e^{-M}}{18M}\right)(1 - \cos\alpha) \tag{4.13}$$

$$\upsilon_{str} = \left(\frac{k_B T}{2m}\tau_c\right)\left(\frac{5M}{(M - 1 + e^{-M})(2 - \cos\alpha - \cos 2\alpha)} - 1\right) \tag{4.14}$$

Note the collision interval dependence: the streaming contribution dominates in the large τ_c (infrequent collision) regime, while the collision step contribution dominates in the small τ_c (frequent collision) regime. Varying these parameters within practical limits, one can access a range of viscosities spanning two orders of magnitude from 0.1 cP, to 0.7 cP (water at 310 K), to 10 cP (50% glycerol).

We may also find an expression for the self-diffusion coefficient as a function of collision parameters [147]. Interestingly, only the streaming step contributes; it has the same expression for all MPC collision rules

$$D_{SRD} = \left(\frac{k_B T}{2m}\tau_c\right)\left(\frac{3M}{(M - 1 + e^{-M})(1 - \cos\alpha)} - 1\right) \tag{4.15}$$

4.3.4 Hydrodynamic Cutoff Length

The cutoff length for SRD hydrodynamics has been identified as the length at which the time integral of transverse velocity correlations among SRD particles goes from an Oseen-like dependence on wavenumber $\propto (\eta k^2)^{-1}$ to an asymptotic constant [148]. It can be expressed using the dynamic viscosity

$$\lambda_c = \pi\sqrt{2\tau_c \upsilon_{SRD}} \tag{4.16}$$

This expression is plotted in **Fig. 4.1** as a function of the collision interval using typical STRD collision parameters. In the frequent collision regime $\upsilon_{SRD} \propto {\tau_c}^{-1}$, leading to a nearly constant cutoff length $\lambda_c \approx 1.44\text{a}$. When collisions are infrequent, the streaming contribution dominates υ_{SRD}; the cutoff length grows as τ_c, scaling with the mean free path of the MPC particles.

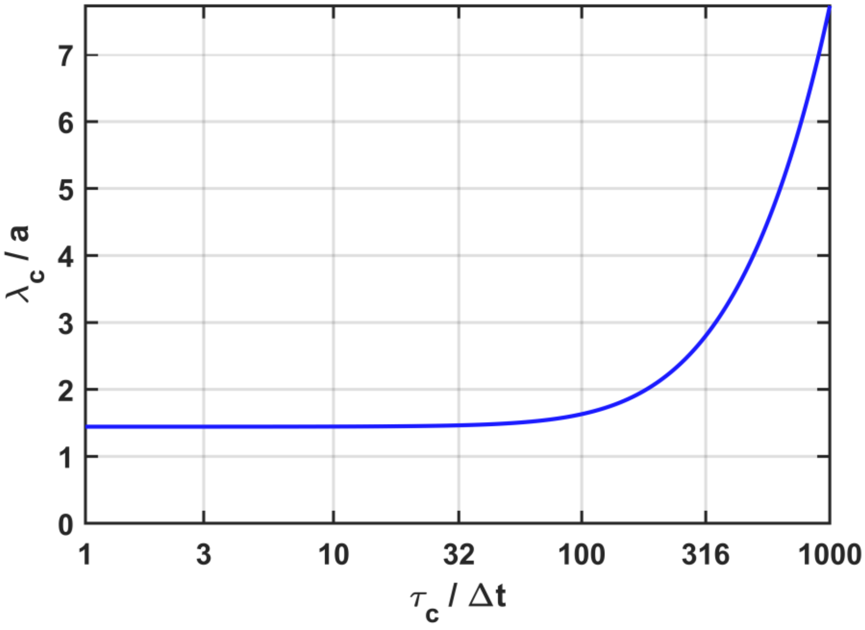

Figure 4.1: The hydrodynamic cutoff length for an SRD fluid at 310 K in units of collision cell size as a function of the collision interval in units of MD timestep. The SRD other collision parameters are *m* = 72 amu, α = 180°, *a* = 2.0 nm, and *N* = 2.5 nm^{-3}.

4.3.5 Momentum-Preserving Thermostat

Thermostats which operate on SRD particle velocities must be designed not disrupt the mesoscopic flow field by only modifying the relative velocities, leaving the mean velocity of the cell unchanged. One such thermostat has been developed using a Monte-Carlo type procedure to randomly rescale the relative velocities in each cell at the end of the SRD collision step [146]. This thermostat does not alter SRD viscosity. Within each cell, a scaling factor S is chosen, with equal probability, to be either $(1+\varepsilon)$ or $(1+\varepsilon)^{-1}$ where ε is an input parameter less than one. Then a uniform random number is chosen $p \in [0,1]$. If this number is less than the acceptance probability p_A = min(1, A), the relative velocities of SRD particles in the cell are rescaled by S. The parameter A depends on the number of particles in the cell N_c, the target temperature T_0, and the relative kinetic energy of SRD particles in the cell

$$A = S^{3(N_c-1)} \exp\left(-\frac{m\,(S^2-1)}{2\,k_B T_0}\sum_{i=1}^{N_c}(\vec{v}_i - \vec{u}_c)^2\right) \tag{4.17}$$

4.4 Andersen Collisions

MPC-Andersen assigns new relative particle velocities by drawing random components $\vec{v}_{i,rand}$ from a Gaussian distribution with a variance $\sqrt{k_B T/m}$. The new relative velocities are adjusted to ensure they sum to zero by subtracting their average

$$\vec{v}_i' = \vec{u}_c + \vec{v}_{i,rand} - \frac{1}{N_c}\sum_{j=1}^{N_c}\vec{v}_{j,rand} \tag{4.18}$$

where N_c is the number of particles in the collision cell. Since this collision rule automatically applies an Andersen thermostat to the fluid, simulations using it will sample configurations from the canonical ensemble. Once again, the viscosity has an analytic expression based on the simulation parameters from **Table 4.1** [149]

$$\eta_{MPCA} = \left(\frac{mM}{a^3}\right)(\upsilon_{col} + \upsilon_{str}) \tag{4.19}$$

$$\upsilon_{col} = \left(\frac{a^2}{12\tau_c}\right)\left(\frac{M - 1 + e^{-M}}{M}\right) \tag{4.20}$$

$$\upsilon_{str} = \left(\frac{k_B T}{2m}\tau_c\right)\left(\frac{2M}{M - 1 + e^{-M}} - 1\right) \tag{4.21}$$

4.5 Angular Momentum Conservation

The collision step changes the angular momentum of each cell by $\Delta\vec{L}_c$, which may be expressed in terms of the cell-relative particle coordinates and velocities as

$$\Delta\vec{L}_c = \sum_{j=1}^{N_c} \tilde{r}_j \times m_j \tilde{v}'_j - \sum_{j=1}^{N_c} \tilde{r}_j \times m_j \tilde{v}_j \tag{4.22}$$

$$= \sum_{j=1}^{N_c} \tilde{r}_j \times m_j\left(\tilde{v}'_j - \tilde{v}_j\right) \tag{4.23}$$

This leads to change in the angular velocity of the cell $\Delta\vec{\omega}_c$ given by

$$\Delta\vec{\omega}_c = \mathbb{I}_c^{-1}\Delta\vec{L}_c = \mathbb{I}_c^{-1}\sum_{j=1}^{N_c} \tilde{r}_j \times m_j\left(\tilde{v}'_j - \tilde{v}_j\right) \tag{4.24}$$

Using the cell's moment of inertia tensor $\mathbb{I}_c$ defined using the cell-relative particle positions $\tilde{r}_i$ with

$$\mathbb{I}_{c,xx} = \sum_{j=1}^{N_c} m_j\left({\tilde{r}_{j,y}}^2 + {\tilde{r}_{j,z}}^2\right) \qquad \mathbb{I}_{c,xy} = \mathbb{I}_{c,yx} = -\sum_{j=1}^{N_c} m_j\left(\tilde{r}_{j,x}\tilde{r}_{j,y}\right)$$

$$\mathbb{I}_{c,yy} = \sum_{j=1}^{N_c} m_j\left({\tilde{r}_{j,x}}^2 + {\tilde{r}_{j,z}}^2\right) \qquad \mathbb{I}_{c,yz} = \mathbb{I}_{c,zy} = -\sum_{j=1}^{N_c} m_j\left(\tilde{r}_{j,y}\tilde{r}_{j,z}\right)$$

$$\mathbb{I}_{c,zz} = \sum_{j=1}^{N_c} m_j\left({\tilde{r}_{j,x}}^2 + {\tilde{r}_{j,y}}^2\right) \qquad \mathbb{I}_{c,xz} = \mathbb{I}_{c,zx} = -\sum_{j=1}^{N_c} m_j\left(\tilde{r}_{j,x}\tilde{r}_{j,z}\right)$$

Thus, the change in particle velocities due to the collision step has a component which can be written

$$\Delta\vec{v}_{i,rot} = \Delta\vec{\omega}_c \times \tilde{r}_i \tag{4.25}$$

The equation for the collision step may be rewritten with an extra term to cancel this change in angular momentum:

$$\vec{v}_i' = \vec{u}_c + \tilde{v}_i' - \Delta\vec{\omega}_c \times \tilde{r}_i \tag{4.26}$$

This new form of the collision step conserves angular momentum within each collision cell.

4.6 Grid Shifting

In its original formulation, the SRD algorithm did not respect Galilean invariance due to anisotropy introduced by the collision cell grid [150], [151]. As nearby particles repeatedly collide with one another, they build up velocity correlations that couple to the mesoscopic flow field. This oversight is corrected by sampling different groups of MPC particles at each collision step, typically implemented with a random shift of the collision cell lattice before each collision using a uniform random shift of $(-a/2, a/2)$ along each axis. This procedure is generally not necessary when the MPC mean free path approaches or exceeds the size of the collision cells.

4.7 Alternative Hydrodynamics Models

The computational fluid dynamics literature contains a rich spectrum of other hydrodynamic solvers operating at various levels of abstraction; from pure continuum methods which solve the Navier-Stokes equations with finite differences to fully atomistic "explicit-solvent" MD simulations at the most granular level of description.

Each level of abstraction, involves tradeoffs between computational efficiency and its realm of suitable applications. Mesoscopic techniques bridge the gap between atomic or molecular fluid simulation and continuum Navier-Stokes hydrodynamics. Broadly, these may be divided into those methods which model underlying dynamics through tracer particles (Dissipative Particle Dynamics, Multi-particle Collision Dynamics, Lattice-Boltzmann), those which couple to an external momentum field (Fluctuating Hydrodynamics, particle-in-mean-field) and those which explicitly calculate forces from the hydrodynamic interaction tensors (Brownian and Stokesian dynamics).

Early mesoscopic fluid simulations used cellular automata to model fluid particle dynamics on a lattice. In the lattice gas automata (LGA) method [152], [153], tracer particles are constrained to hopping between points on a discrete spatial lattice, which interaction occurring through discrete collisions on the lattice sites. Particle occupancy on each site is binary in the lattice gas method. This was later extended by modeling the site occupancy with particle distributions in the lattice Boltzmann method [141], [154]–[156]. This method is efficient, but has difficulty handling complex boundary conditions or incorporating proper thermal fluctuations (though the latter may be including through additional noise terms [157]).

Dissipative particle dynamics (DPD) [140], [158], [159] is another popular approach which models a fluid with coarse-grained beads subject to conservative, dissipative, and random forces. This is similar to a coarse-grained explicit solvent coupled to a thermostat with one crucial difference: all three forces are designed to conserve momentum. This requires the viscous and random forces to be pair-correlated among DPD particles. A major drawback to this approach is that it still requires pairwise distance calculations among the constituent particles in order to evaluate the forces between them.

Forces between solute particles in dilute solution may be directly calculated without modeling the flow field using Brownian [160] or Stokesian dynamics [161]. Methods from this family model solute dynamics with the Langevin equation, explicitly including hydrodynamic interaction forces through the calculation of many-body diffusion tensors. Calculating the full diffusion tensor is computationally prohibitive, as it depends on all particle positions in the system [162]. This often motivates the use of a simplified pair-wise approximation called the Rotne-Prager-Yamakawa (RPY) tensor [163]–[165]. Full evaluation of this interaction tensor still scales as $O(N^2)$, but is amenable to a decomposition into Laplace potentials which may be calculated using the fast multiple method [166]. Brownian dynamics has been used to study the self-assembly of coarse-grain membranes modeled with lipids each consisting of a single head and tail bead [143].

Other methods use a continuum description of the solvent for hydrodynamics. For example, the fluctuating hydrodynamics approach [167] couples solute dynamics to a continuum hydrodynamic field evolved on a discrete grid using finite element methods. This method has been implemented for implicit-solvent coarse-grain models in the LAMMPS MD software package [168]. The particle-in-mean-field or MD-SCF method takes this approach a step further, representing all interactions in the simulation with a self-consistent mean field [169]–[171] and mapping this back to molecular dynamics through forces for time evolution.

Among these, the MPC algorithm was selected for several reasons: (1) Its particle-based nature readily admits an implementation in existing MD software; (2) it inherently contains local fluctuations in density and temperature, which are relevant on the length scales of interest; (3) it easily supports complicated boundary conditions by various coupling methods; and (4) it efficiently scales to thousands of processors and millions of solute particles. MPC methods have been successfully applied to study a wide variety of soft matter systems, including colloidal suspensions [137], [146], polymer solutions [172]–[174], self-propelled solutes (i.e. model sperm cells) [175], [176], red blood cells [177], [178], and vesicles [179]. In the literature, colloids are often modeled with large spheres (relative to the collision cells) while membranes (cells and vesicles) are represented with dynamically triangulated meshes. Coupling these systems to MPC fluids may be accomplished through incorporating the solute particles in the MPC collision steps [178], adding repulsive forces between MPC particles and solutes [180], or explicitly modeling hard-sphere collisions between MPC particles and large solute particles (typically colloids) [146].

Chapter 5

STRD MARTINI IMPLEMENTATION

5.1 Introduction

There has been a growing appreciation in recent years for the importance of hydrodynamic interactions in membrane dynamics [45], [119], [143], [181], [182]. At the same time, the length and time scales accessible to simulation has begun to converge with those accessible to experiment, particular in the study of lateral diffusion [29], [183]. However, accurate modeling of hydrodynamic transport using traditional molecular dynamics (i.e. with explicit solvent particles) is not feasible due to the system sizes (and consequently number of particles) required by the long-range nature of hydrodynamic interactions in the low-Reynolds number regime. Calculating pairwise forces between the solvent particles demands an overwhelming majority of the available computational resources at the requisite system sizes. This predicament constitutes an unmet scientific need as novel algorithms and software implementations are required for accurate and efficient modeling of hydrodynamic interactions at scale.

We have met that need by implementing the efficient MPC algorithm in the popular open-source MD software package GROMACS v5.0.1 [47], opening the way to further studies of membrane dynamics with proper accounting for hydrodynamic interactions. Our MPC implementation is not specific to membrane simulation and may be adapted to colloids, polymers, or any other molecular system which can be modeled in GROMACS. Since our interest is focused upon membrane dynamics, and lateral diffusion in particular, our first application for the MPC-GROMACS code is to reintroduce hydrodynamic momentum transport to coarse-grain membranes modeled with an implicit-solvent force field called Dry Martini [48]. We call this combination "Stochastic Thermostatted Rotation Dynamics" (STRD) Martini, in keeping with the naming convention set forth by Martini and Dry Martini.

As discussed in section 4.7, the literature contains a plethora of hydrodynamic solvers coupled to colloids, proteins, and continuum membranes. In formulating our approach for STRD Martini, we selected Dry Martini as the membrane model because it provides chemically-resolved models of individual lipid species parametrized in the absence of an explicit solvent. The MPC algorithm was chosen as the mesoscopic hydrodynamics model due to its natural interoperability with existing GROMACS code: MPC dynamics are based upon particles moving though continuous space in discrete timesteps – just like other particles in a GROMACS simulation. As such, GROMACS may treat MPC particles just as any other particle for the purposes of integration, parallelization, trajectory writing, analysis, and force calculation (when desired). When combined with domain decomposition, STRD Martini scales to thousands of processors, providing accurate hydrodynamics while running at least an order of magnitude faster than equivalent explicit-solvent simulations.

Using STRD Martini begins with adding a bath of MPC tracer particles to a Dry Martini membrane (illustrated in **Fig. 5.1**). At runtime, their mutual pairwise interactions are disabled and replaced with MPC collisions. Collisions are performed after an integer number of timesteps with user-supplied collision parameters. MPC streaming steps are handled by the GROMACS integrator. This inter-dependence is not an impediment to either MD or MPC dynamics, as typical collision intervals are an order of magnitude longer than the MD timestep.

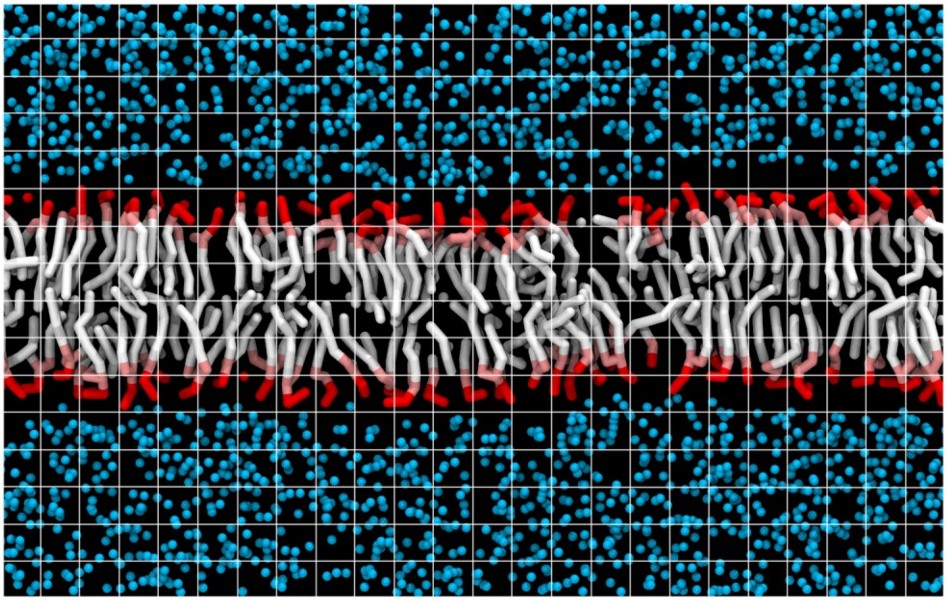

Figure 5.1: Depiction of Martini membrane and SRD solvent overlaid with 1 nm SRD grid to illustrate collision cell size. Radius of SRD particles *does not* reflect their effective size in the repulsive coupling scheme.

The collision cell grid is selected at the start of the simulation based on the requested grid size input parameter. The simulation box must be spanned by an integer number of collision cells in order to respect periodic boundary conditions. The number of cells selected along each axis is the integer number which most closely matches the requested size. Because this is determined independently for each axis (GROMACS boxes may be anisotropic) the final collision cells may be slightly anisotropic. This is a minor issue which is only relevant in the limit of small simulation sizes where the number of collision cells along each axis is small. Once a collision cell grid is selected, it does not dynamically resize. If pressure coupling is employed and the simulation box changes shape, collision cells deform accordingly.

Both SRD and MPC-A collision rules are implemented for STRD Martini. The thermostat described in section 4.3.5 and angular momentum constraint from section 4.5 are also implemented. Two schemes for coupling the MPC fluid to the lipids are available: one based on repulsive forces and another which incorporates lipid particles into the collision step. In the latter case, users may select which particles interact through the collisions by specifying a GROMACS atom group. Despite its name, STRD Martini is not specific to the Martini force field. With very little effort, it may be adapted to any implicit solvent force field for GROMACS simulations.

5.2 Standard Simulation Parameters

MD simulation parameters for STRD Martini are largely inherited from Dry Martini (see **Table 5.1)**. This is necessary maintain consistency with the lipid force field parameterization. However, there are three major differences between the set of parameters used for STRD Martini and Dry Martini.

First, the velocity-Verlet integrator is used with STRD Martini in place of the stochastic integrator used with Dry Martini. Velocity-Verlet is required to conserve energy with MPC collisions (see section 5.4). The stochastic integrator was originally selected for Dry Martini to provide a source of friction and thermal noise to the membrane [48]. Both functions are fulfilled by the MPC fluid in STRD Martini.

Second, STRD Martini simulations thermalize the membrane through interactions with the solvent. No explicit thermostat is applied to the membrane. Instead, the MPC fluid temperature is maintained during the MPC collisions by a special momentum-conserving thermostat. Temperature coupling for Dry Martini is provided by the stochastic integrator.

Third, pressure coupling must be handled with care when using STRD Martini. Both the collision cell size and the MPC particle density are important parameters for controlling MPC fluid properties. As such, STRD simulations should avoid significant changes in volume. Fixed volume is encouraged for all simulations after equilibration. Semiisotropic pressure coupling may be employed for interfacial systems, provided the vertical size of the box is held fixed as the lateral size varies. In this case, coupling the box size to surface-tension is required for the cancellation of isotropic pressure contributions arising from the kinetic energy of streaming MPC particles. Pressure coupling is a lesser concern in very large systems, as NVT and NPT ensembles with the same average pressure become equivalent in the thermodynamic limit.

Table 5.1: Standard MD parameters for STRD Martini simulations in GROMACS.

Integration	Velocity-Verlet with $\Delta t = 20$ fs Center of mass motion removed every timestep, treating membrane and solvent separately
Neighbor Lists	Group scheme, recalculated every 10 timesteps; Cutoff at 14 Å
Van der Waals	Potential shifted to zero over 9-12 Å.
Electrostatics	Electrostatic potential shifted to zero over 0-12 Å; Relative permittivity $\varepsilon_r = 15$
Thermostat	Collision cell thermostat for the MPC fluid, no thermostat for the membrane; $T_0 = 310$ K, $\epsilon = 0.10$
Barostat (Production)	Semiisotropic coupling only, fixed size along Z axis; Better to use fixed volume to keep MPC cell size fixed
Barostat (Equilibration)	Berendsen, semiisotropic coupling, fixed size along Z axis; $P_0 = 1$ bar, $\tau_P = 12$ ps Compressibility = 3.0×10^{-4} bar^{-1} (lateral), 0 (normal)

Additional parameters controlling the MPC / SRD collision parameters are listed in **Table 5.2**. Collision parameter selection is up to the user, though the mass of MPC particles is fixed at 72 amu – equivalent to the mass of Dry Martini beads. Parameters should be selected in a systematic way, balancing the requirements for achieving desirable characteristic hydrodynamic numbers, transport coefficients (e.g. viscosity), hydrodynamic resolution, and performance.

When using the SRD collision rule, particle density should be chosen such that the number of particles per cell is between 3 and 20 [138]. This requirement imposes a tradeoff between particle density and collision cell size. Another tradeoff involving the particle density comes about from performance considerations. STRD Martini incurs two major performance penalties: (1) inter-process communication can become significant when short collision intervals are used; and (2) the addition of MPC particles adds significant book keeping requirements for GROMACS, especially for simulations with large MPC solvent baths. Thus, better performance is available when using lower MPC particle density and longer collision intervals. However, both tend to lower viscosity and Schmidt number (see section 5.5.3). The collision interval is also key to obtaining shorter hydrodynamic cutoff lengths, which depend on both viscosity and collision interval. Lowering the cutoff delivers hydrodynamic flows with finer spatial resolution. This cutoff can also be lowered by increasing particle density, but this requires smaller collision cells in compensation, tying back into requirements for the average number of particles per cell.

A reasonable set of collision parameters that strikes a good balance between these considerations is $N = 2.5$ nm^{-3}, $a = 2.0$ nm, $\tau_c = 200$ fs, and $\alpha = 180°$. At 310 K, these yield a viscosity of 0.63 cP, a cutoff length of 2.9 nm, and Schmidt number of

1021 while attaining good performance. For the remainder of this document, these will be referred to as the ***standard STRD collision parameters***. Starting from this set of parameters, minor adjustments can be made, principally by tuning SRD collision angle and collision interval.

Table 5.2: Additional GROMACS input file (mdp) parameters for STRD Martini.

General input parameters for MPC collisions:	
mpc-type	Selected collision rule (off | srd | at | at+a).
mpc-grps	Atom groups to include in collision step.
mpc-freq	How often to do collisions (# timesteps).
mpc-cellsize	Desired collision cell size (nm). True size at runtime will depend on periodic box size.
mpc-seed	Seed for collision RNG (-1 for random seed).
Additional input parameters for the SRD collision rule:	
srd-angle	Collision angle (degrees)
srd-tcouple	Enable collision step thermostat (yes | no)
srd-tc-str	Thermostat strength parameter ϵ
srd-tc-ref-t	Thermostat reference temperature $T0$ (K)
Additional input parameters for collision cell data output:	
mpc-out-freq	How often to output cell data (# timesteps)
mpc-out-samp	How often to sample cell data for output averages (# timesteps)
mpc-out-vel	Write cell velocities to binary output file (yes | no)
mpc-out-num	Write cell occupancies to binary output file (yes | no)

5.3 MPC-Membrane Coupling Methods

Since the Dry Martini lipid model has been parameterized in the absence of solvent, indirect lipid-lipid interactions mediated by nearby solvent particles are incorporated into the lipid-lipid interaction parameters. This parameterization can include thermodynamic effects such as hydrophobicity, but not long-ranged hydrodynamic interactions. The object of STRD Martini is to restore hydrodynamic momentum transport while minimally perturbing the thermodynamic properties of Dry Martini membranes. Two methods for coupling the SRD fluid to Dry Martini lipids were explored. First, the ***repulsive coupling*** method introduces a pairwise force acting between MPC particles and lipid headgroups. This force uses a short ranged, purely repulsive interaction to effect momentum transfer with the membrane. Second, the ***collisional coupling*** method allows lipid headgroups to participate in MPC collision events, with no other interactions between lipids and MPC particles.

5.3.1 Repulsive Coupling

Perhaps the simplest way to interface the SRD fluid with Dry Martini is to define a purely repulsive van der Waals interaction between MPC particles and lipid beads. A suitable interaction is provided by the repulsive Weeks-Chandler-Anderson (WCA) potential [184], sometimes called the truncated Lennard-Jones potential. It is simply a Lennard-Jones potential truncated at its global minimum $r_{\min} = 2^{1/6}\sigma$ and shifted to zero (see **Fig. 5.2**).

$$U_{WCA}^{R}(r) = \begin{cases} U_{LJ}(r) + \varepsilon, & r < r_{\min} \\ 0, & r \geq r_{\min} \end{cases} \tag{5.1}$$

This interaction has been previously used in the MPC literature [185], [186] for studies of spherical solute particles suspended in a bath of MPC particles. When applying this interaction to interfacial systems, one must be mindful of the normal pressure exerted by the solvent.

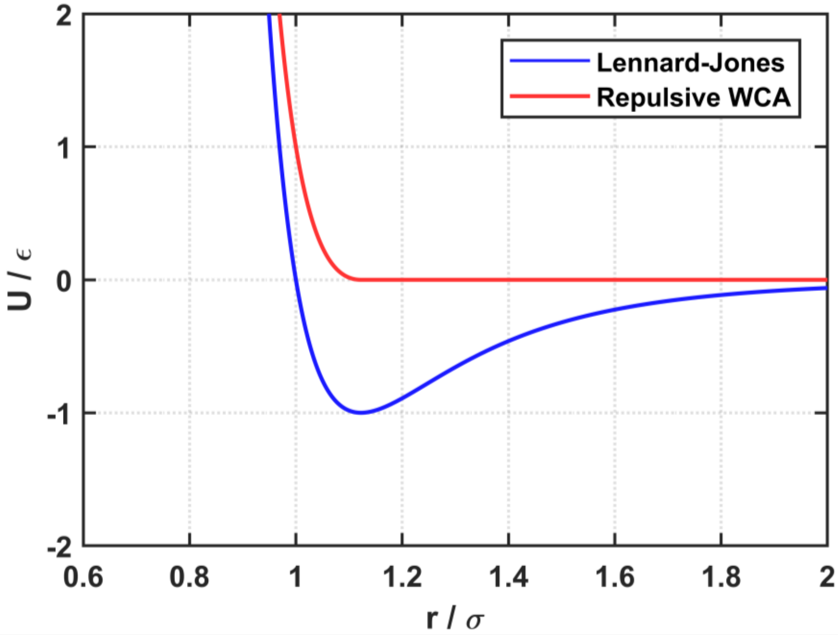

Figure 5.2: Lennard-Jones and repulsive WCA potentials in natural units.

For simplicity of implementation, we mimic a WCA potential using a shallow-well Lennard-Jones potential with σ = 0.80 nm and ε = 0.001 kJ/mol. Note that the WCA potential reduces to the LJ potential for $r < r_{\text{min}}$ in the limit of small ε. This functional form yields an effective MPC particle radius comparable to the size of Dry Martini lipid beads (see **Fig. 5.3**). The choice of parameters involves a delicate trade-off with the particle density in the interest of minimizing the normal pressure. If the MPC particles are too small, they penetrate the membrane and become unphysically trapped between the leaflets. When too many congregate inside the membrane, their repulsive interactions drive the lipids apart, ultimately lysing the membrane. However, if the MPC particles are too large or too dense, the additional pressure at the interface drives the membrane into a gel phase when zero surface tension is enforced.

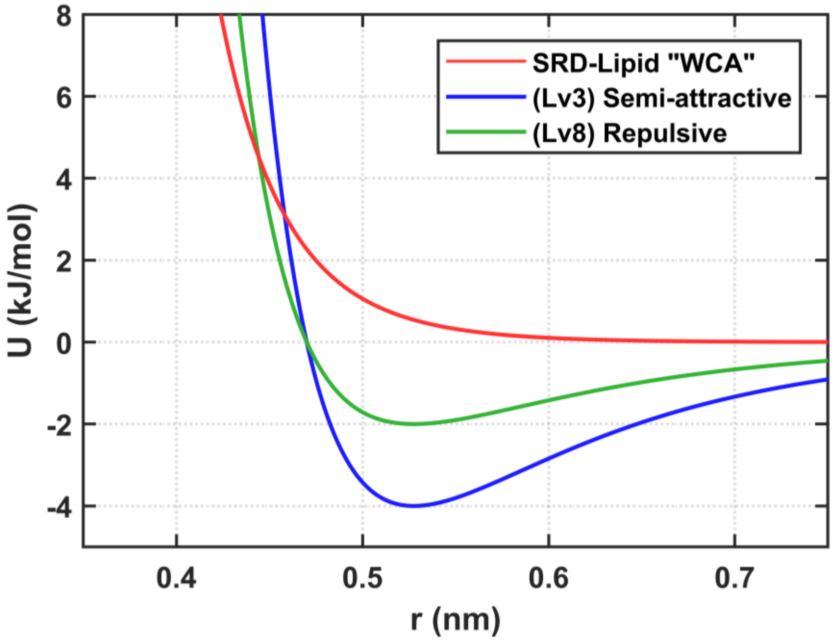

Figure 5.3: SRD-headgroup interaction compared with two sample van der Waals interaction types from the Dry Martini force field. Level 3 is the "semi-attractive" potential and level 8 is the "repulsive" potential.

Membrane perturbations from the SRD fluid depend on the effective size of the MPC particles, which may be regarded as the radius of their potential at a half $k_B T$

$$U_{LJ}\left(r_{ef}\right) = \frac{1}{2} k_B T \tag{5.2}$$

This expression depends on both LJ parameters, with larger values of σ required to reach the same effective radius when ε is small. The effective size as a function of Lennard-Jones parameters is illustrated in **Fig. 5.4**, with the twelve solid points indicating the subset of parameters considered during the STRD repulsive coupling parameterization which did not either crash the simulation, nor drive the membrane into the gel phase during 1 μs of zero surface tension equilibration. The same sample parameter set is presented in **Fig. 5.5** showing the equilibrium area per lipid and average density of MPC particles trapped between the leaflets. Numeric values for area per lipid and membrane thickness are reported in **Table 5.3** for each set of LJ parameters. The SRD collision parameters used during this equilibration are largely irrelevant, as the equilibrium membrane properties are overwhelmingly determined by the selected LJ parameters of the MPC-membrane interaction.

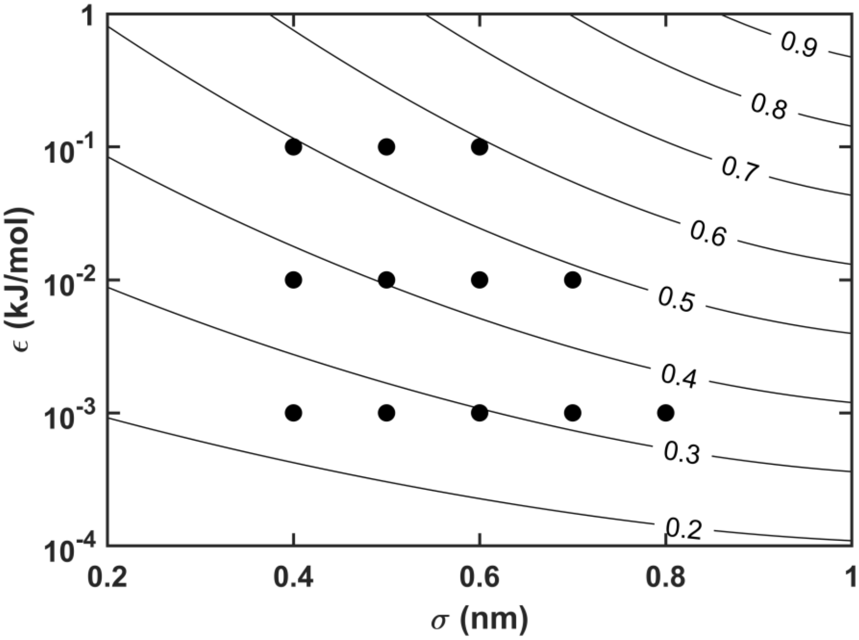

Figure 5.4: Contours show effective MPC radius r_{eff} in nm as a function of Lennard-Jones parameters σ and ε. Solid circles indicate parameter sets shown in **Fig. 5.5**.

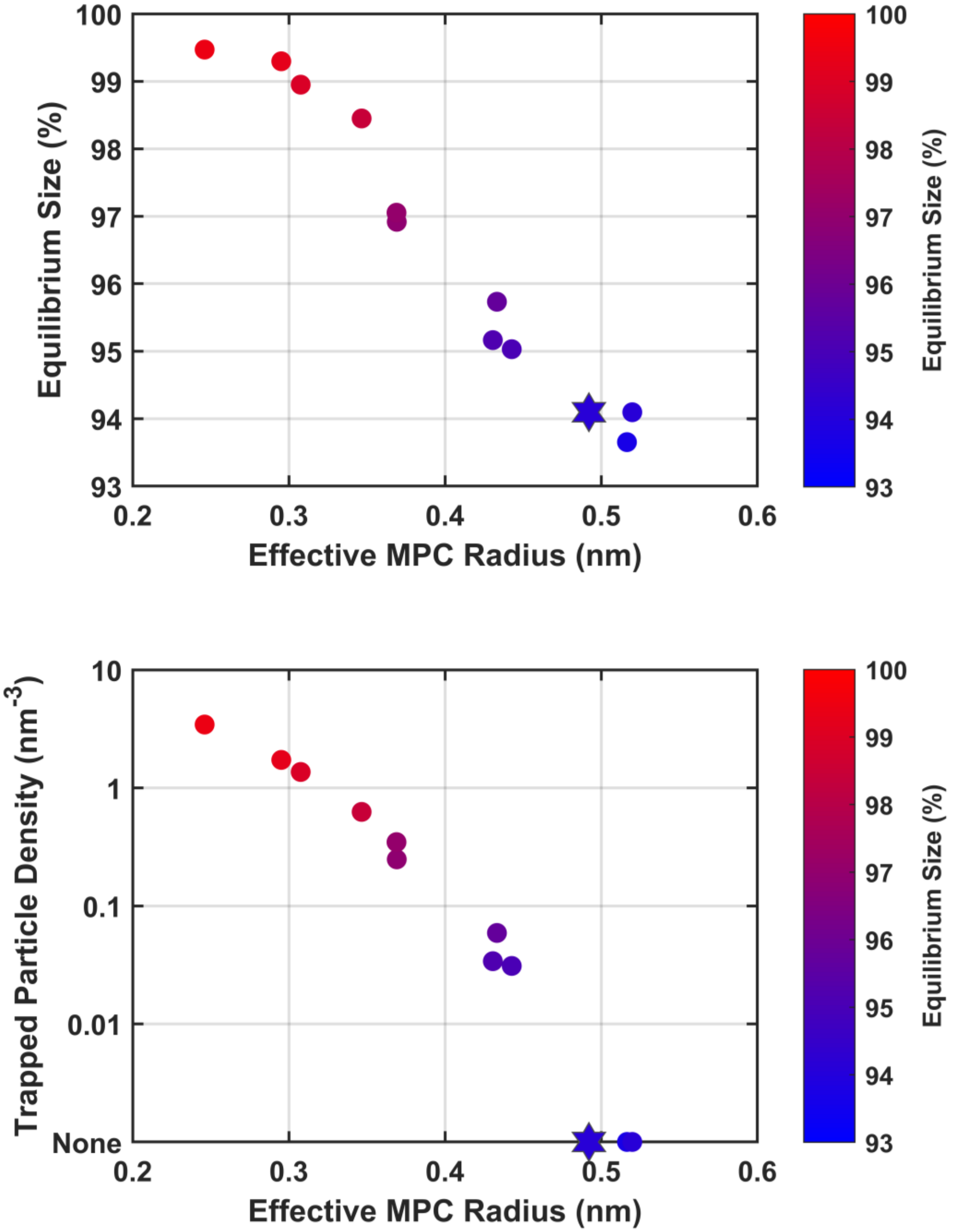

Figure 5.5: Equilibrium size (top) and density of trapped MPC particles (bottom) after 1 μs for POPC membranes coupled to high-density (N = 10 nm^{-3}) STRD fluid through repulsive forces. Each point represents a set of LJ parameters from **Fig. 5.4**, shaded according to the equilibrium size given in the top panel. Standard STRD parameters (σ = 0.80 and ε = 0.001) are indicated by a six-point star.

Table 5.3: Equilibrium properties of POPC membranes for Dry Martini and STRD Martini with repulsive coupling at high SRD particle density (= 10 nm^{-3}). Lennard-Jones parameters which result in a gel phase membrane are not included. Thickness is measured from phosphate to phosphate bead.

σ (nm)	**ε (kJ/mol)**	**APL (Å^2)**	**Thickness (nm)**
0.4	0.001	63.092 ± 0.037	4.232 ± 0.026
0.5	0.001	62.434 ± 0.038	4.228 ± 0.025
0.6	0.001	59.896 ± 0.039	4.303 ± 0.026
0.7	0.001	57.748 ± 0.040	4.399 ± 0.028
0.8	0.001	56.458 ± 0.045	4.465 ± 0.031
0.4	0.01	62.873 ± 0.035	4.222 ± 0.025
0.5	0.01	60.063 ± 0.039	4.295 ± 0.027
0.6	0.01	57.585 ± 0.039	4.406 ± 0.027
0.7	0.01	55.927 ± 0.050	4.494 ± 0.033
0.4	0.1	61.804 ± 0.041	4.232 ± 0.027
0.5	0.1	58.440 ± 0.042	4.362 ± 0.029
0.6	0.1	56.458 ± 0.046	4.464 ± 0.032
Dry Martini (Reference)		*63.766 ± 0.039*	*4.210 ± 0.027*

From the results reported in **Table 5.3** and illustrated in **Fig. 5.5**, we see that only those interaction parameters with the largest effective size prevent MPC particles from unphysically infiltrating the membrane and becoming trapped. At the same time, such “large” MPC particles contribute an enormous normal pressure, which cause the membrane to contract by up to 10%. Using even larger MPC particles forces the membrane to gel. It is worth remarking that this effect is rather counterintuitive. When one imagines compressing a membrane, it is reasonable to expect lateral expansion rather than contraction with the total volume of the membrane remaining fixed. The fact that our membranes contract is a consequence of the barostat enforcing the zero surface tension. In an MD simulation, surface tension is determined by a balance of normal (z) and lateral (x-y) stresses via

$$\gamma_{bilayer} = L_z\left(P_{zz} - \frac{P_{xx} + P_{yy}}{2}\right) \tag{5.3}$$

where L_z is the size of the simulation box normal to the bilayer and $P_{\alpha\beta}$ is the pressure tensor. Thus, to require zero surface tension is to require the lateral pressure to equal the normal pressure. When normal pressure is increased, the simulation box must contract to induce a commensurate increase in the lateral pressure.

MPC particles contribute to both the kinetic energy and virial terms of the pressure tensor. However, due to the isotropy of the MPC kinetic energy contribution, only the virial term contributes to the surface tension. This term is dependent upon both the range of the MPC-lipid interaction (as we have seen in **Table 5.3**) and the density of the MPC particles, both of which may be controlled to minimize the normal pressure. The parameters selected for the STRD repulsion are those which yield the smallest effective size while still preventing MPC particles from becoming trapped within the membrane (see **Fig. 5.5**). To further reduce the pressure and recover sensible membrane properties, the MPC density must be decreased. Attaining water-like viscosity at such low density is challenging; we must use larger collision cells and shorter collision intervals. This is explored further in section 5.3.3.

5.3.2 Collisional Coupling

An alternative strategy for coupling the MPC fluid to Dry Martini without applying spurious normal pressure to the membrane is to simply include some lipid headgroup beads in the collision step. This “direct coupling” or “collisional coupling” scheme has been used in the literature to study the dynamics of polymer chains [173], [187] and triangulated meshes representing vesicles [177], [178]. Since the solute particle masses are not necessarily equal to the MPC particle mass, the mean collision cell velocities must be calculated using mass-weighted particle velocities to satisfy conservation of momentum within each collision cell

$$\vec{u}_c = \frac{\sum m_i \vec{v}_i}{\sum m_i} \tag{5.4}$$

When using the collisional coupling method, the total solvent and solute masses within a collision cell should be of the same order [188]. If the total solvent momentum significantly exceeds that of the solute, fluctuations from the MPC solvent become a source of friction for the solute, with the collision step effectively adding a noise term (i.e. random force) to the solute equations of motion. Since MPC particles in STRD Martini have the same mass as Dry Martini beads, a limiting range is set on the acceptable MPC particle density, it should also be of the same order as the lipid headgroup density within the interfacial collision cells. Consider a membrane with an APL of 0.6 nm^{-2} coupled to STRD with 2 nm collision cells. This corresponds to an average of ~6.7 lipid headgroups per interfacial collision cell. Requiring an equivalent 6.7 MPC particles per cell corresponds to an average MPC density of 0.84 nm^{-3}. This sets an upper limit on acceptable density around N = 2.5 nm^{-3}, which is ~0.5 orders of magnitude larger than a state of solvent-solute mass balance.

Using even larger cells is problematic in two respects (1) it decreases the resolution of the mesoscopic flow field, degrading its ability to model hydrodynamic interactions among nearby lipids and (2) it introduces additional coupling among the dynamics of nearby lipids through their mutual participation in the collision step. For

2 nm collision cells, nearest and second-nearest neighbors will often be located in the same collision cell. Resolving the dynamics of individual lipids would require ~0.5 nm collision cells, along with very high SRD particle densities to adequately fill the cells (seriously degrading performance) and drastically lower MPC particle mass to satisfy the momentum balance requirement. As a compromise, a = 2 nm and N = 2.5 nm^{-3} are used as the standard cell size and density in STRD simulations.

Furthermore, without some form of repulsive interaction, MPC particles can freely stream *through* the membrane. This enhances coupling between the leaflets and tightly couples solvent flows on either side. This may be advantageous in the context of the Saffman-Delbrück theory, where (1) the bilayer is modeled as a uniform slab having no variation in its velocity profile along z and (2) stick boundary conditions are assumed, requiring the flow field of the membrane and solvent to exactly match at the interface. Problems may arise when modeling other membrane properties, such as the interleaflet friction, is required. For longer collision intervals, density waves normal to the membrane may pass through it. This may be a problem when the Mach number is high enough, corresponding to a compressible fluid, or when parameters are chosen which result in a gas-like low Schmidt number.

Another important consequence of including lipid headgroups in the collision step is a tight coupling of the lipid dynamics to the SRD collision parameters. To demonstrate this, lipid self-diffusion coefficients were calculated for five simulations of 10 nm POPC membranes with different collision angles. The other collision parameters were held fixed at N = 0.75 nm^{-3}, a = 2.0 nm, τ_c = 100 fs, and α = 180. Each simulation was run at fixed volume for 2 μs with a cell thermostat applied to MPC colliding particles only and no COM removal with 2 fs timesteps. Diffusion coefficients were found by breaking each trajectory into ten 200 ns segments and calculating the MSD for sub-trajectory. The reported uncertainty is the standard error of these ten calculations. From the results presented in **Fig. 5.6**, it is clear that lipid diffusion slows as the collision angle is increased. Including lipid headgroup beads in MPC collisions increases the membrane surface viscosity through the introduction of random noise forces.

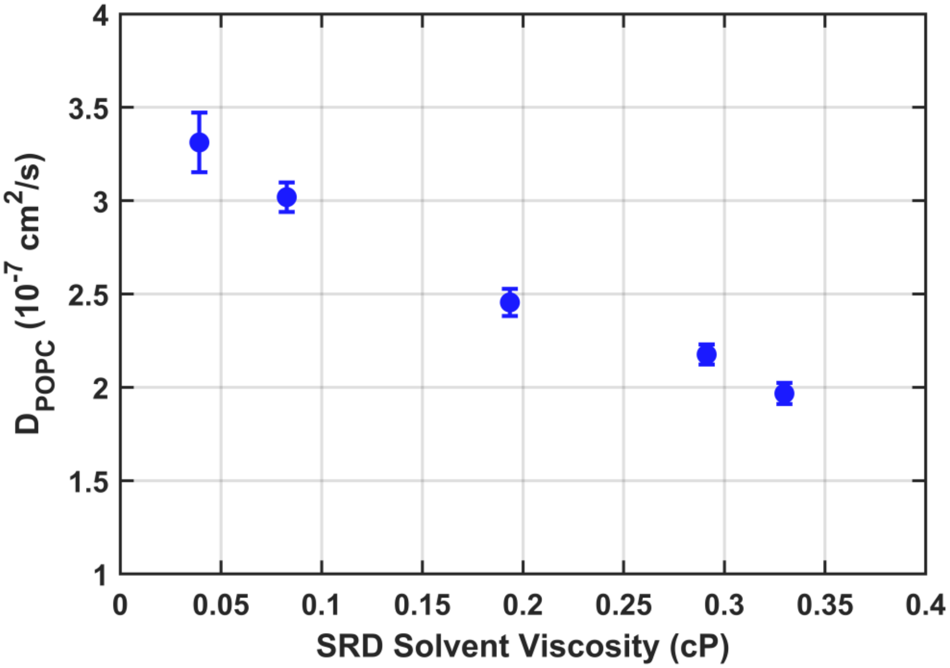

Figure 5.6: Self-diffusion of POPC lipids in STRD Martini using collisional coupling slows as the collision angle of the collision is increased. This data fits well to a linear model with the horizontal axis expressed in either solvent viscosity or collision angle. It does not comport with a SD-like model given by $A + B\log(1/\eta_w)$. This result reflects the fact that this collision angle dependence does not arise from hydrodynamic considerations, as expressed by equation 3.47 or 3.49, but through an additional viscous friction imparted by the collision step.

5.3.3 Equilibrium Comparison

Under zero surface tension, the area per lipid (APL) of a conventional Dry Martini simulation is 63.7 Å^2 — this is the target for our STRD Martini simulations. As we have seen, coupling MPC particles to the membrane with repulsive interactions reduces the area per lipid in a manner dependent upon the MPC particles' effective size. The dependence on particle density is reported in **Table 5.4** using the standard repulsive interaction parameters $\sigma = 0.80$ nm and $\varepsilon = 0.001$ kJ/mol. Other collision parameters are $a = 1.0$ nm, $\Delta t = 20$ fsec, $\tau_c = 80$ fs, and $\alpha = 180°$. APL is reduced from the conventional Dry Martini result to 62.9 Å^2 at a density of 1 nm^{-3}, a decrease of about 1.3%. This is an acceptable deviation which leaves the bilayer in a fluid state. Alternatively, one could perform simulations under constant surface tension of approximately 10 dyne/cm (= 100 bar-nm) to enforce the exact Dry Martini APL. Higher particle densities significantly perturb the membrane properties when repulsive coupling is employed.

Even closer agreement with the Dry Martini area per lipid is available when the collisional coupling method is used, even at the "maximum" density for STRD simulations (= 2.5 nm^{-3}). Membrane properties for this coupling scheme are also reported in **Table 5.4** using collision parameters $a = 2.0$ nm, $\Delta t = 20$ fs, $\tau_c = 200$ fs, and $\alpha = 180°$. The collisional coupling method also does a better job at reproducing the Dry Martini area compressibility modulus.

Equilibrium area per lipid is the lateral area at which surface tension is zero and area compressibility is the first derivative of surface tension with respect to area. The bilayer surface tension (computed via equation 5.3) as a function of lateral area is shown in **Fig. 5.7** for the membranes presented in **Table 5.4**. Each data point reports the average surface tension in a 1 μs simulation performed at fixed volume with the

MPC fluid coupled to the cell thermostat at 310 K (NVT). Area compressibilities reported in **Table 5.4**. are obtained from linear fits to this data in the vicinity of zero surface tension (i.e. their equilibrium area).

Thus, we have identified simulation protocol for both coupling methods which yield reasonable membrane properties. Nonetheless, the collisional coupling scheme is used throughout the remainder of this work because it is thought to more accurately model stick boundary conditions, such as those assumed in the Saffman-Delbrück theory. This choice should not be regarded as an endorsement of the collisional coupling scheme over the repulsive scheme; both constitute valid approaches and both are supported in STRD Martini.

Table 5.4: Equilibrium system size and area compressibility for Dry Martini and STRD Martini as a function of the SRD particle density. Data for both the collisional coupling (CC) and repulsive coupling (RC) methods are shown. Area compressibility is calculated from a linear fit of the data in **Fig. 5.7** at zero surface tension.

Simulation	**APL (Å^2)**	**Area Compressibility (mN/m)**
RC, $N = 1$	63.02 ± 0.10	384 ± 17
RC, $N = 5$	59.942 ± 0.042	428 ± 13
RC, $N = 10$	56.459 ± 0.082	378 ± 11
CC, $N = 2.5$	63.56 ± 0.18	320 ± 33
Dry Martini	65.084 ± 0.054	333 ± 12

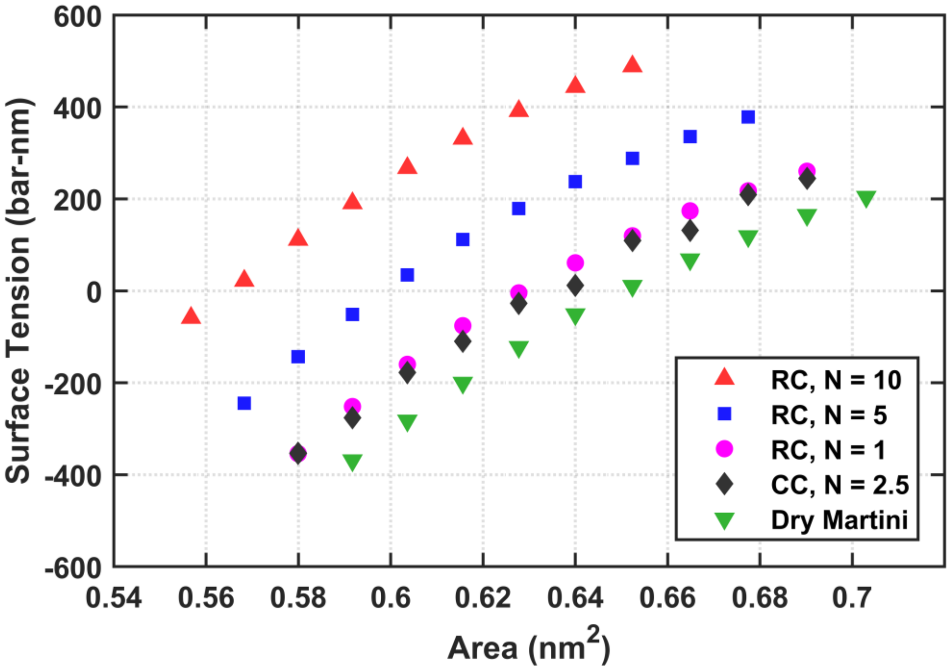

Figure 5.7: Surface tension as function of area per lipid (at fixed volume) for STRD POPC membrane are shown for three densities using repulsive coupling (RC), at standard density using collisional coupling (CC), and for Dry Martini without an MPC solvent.

5.3.4 Alternative Coupling Schemes

Further work on the STRD project might go toward improving the coupling scheme and verifying that the fluid-fluid boundary conditions are being modeled in a realistic way. For instance, GROMACS allows for the definition of "virtual sites" on molecules – abstract interaction sites which can be used to further "coarse-grain" them for specific purposes. For the purposes of coupling to the MPC fluid, each lipid may be assigned a single massive virtual site to represent its entire headgroup located at the headgroup center of mass. Forces applied to these sites are distributed among their underlying constituent particles. The collisional coupling scheme would benefit from including such interaction sites for each lipid in the collision step rather than a single headgroup bead from each lipid. Doing so would ensure flows in the solvent are more effectively distributed to the lipid center of mass, leading to more reliable stick boundary conditions. Coupling to the average lateral motion of lipids rather than noisy individual particle motions would also improve the response of the MPC fluid to lipid movement. The increased mass (from 72 to 216 or 288 amu depending on the number of headgroup particles included) would also improve the solvent-solute momentum balance within interfacial collision cells, providing a firmer justification for using the "standard" MPC particle density $N = 2.5$ nm^{-3}, which would correspond to a state of balance, while plausibly permitting even higher densities.

Implementing other coupling schemes from the MPC literature in GROMACS may prove a daunting task. A potentially promising scheme would create dynamically triangulated meshes for the bilayer surfaces with vertices at the lipid headgroups [177], [178]. The faces of this surface could be coupled to the SRD fluid through ordinary collisions for during the MPC streaming step. A second degree of coupling is then required to translate forces between the Dry Martini lipids and the triangulated surface (perhaps using forces acting on the aforementioned virtual sites or simply using the collisional coupling method as in [178]).

This method would allow different boundary conditions on the full slip vs stick spectrum to be implemented using different collision types between MPC particles and the triangle faces (specular, bounce-back, or some stochastic hybrid as described in the literature for fluid-solid boundaries [189]). It would also properly exclude MPC particles from entering or traversing the membrane. This method was not implemented in this work due to its relative complexity, especially once parallelization is taken into account. Nonetheless, it presents an attractive option for future work.

5.4 Conservation of Energy

Theoretically, simulations of only SRD particles should exactly conserve energy (see section 4.3.2). However, since all computer calculations suffer from numeric artifacts, the total energy in NVE STRD simulations containing only SRD particles slowly declines due to rounding errors in the collision step. These rounding errors produce a rotation matrix with a determinant slightly lower than unity. Different methods for performing the rotation were implemented: a quaternion-based approach, rotation matrix multiplication, and the axis-angle calculation given by equation 4.4. The quaternion approach was quickly disqualified due to the number of floating point multiplications required. The other two rotation methods were evaluated for numerical stability by repeatedly applying random rotations to a unit vector. After 100,000 rotations, the axis-angle rotations shrink the unit vector by 10% while rotation matrix rotations shrink it by 36%. Thus, the axis-angle method was selected.

During the development of STRD, we encountered puzzling violations of energy conservation which caused STRD membrane simulations to heat over time. Depending on the choice of integration and collision parameters, the rate of heating could be enormous or relatively minor (see **Fig. 5.8**). Heating only occurred when SRD particles were coupled to Dry Martini lipids; simulations of only SRD particles did not exhibit this problem. We originally attributed it to interplay between the repulsive coupling interaction potential and the discontinuities in SRD particle trajectories induced by the collision step. We argued that since these SRD particles have a position-dependent potential energy, the post-collision velocities may preferentially propel SRD particles into positions with higher potential energy. In other words, energy conservation was no longer guaranteed since the forces acting on SRD particles were not derived from the gradient of a potential.

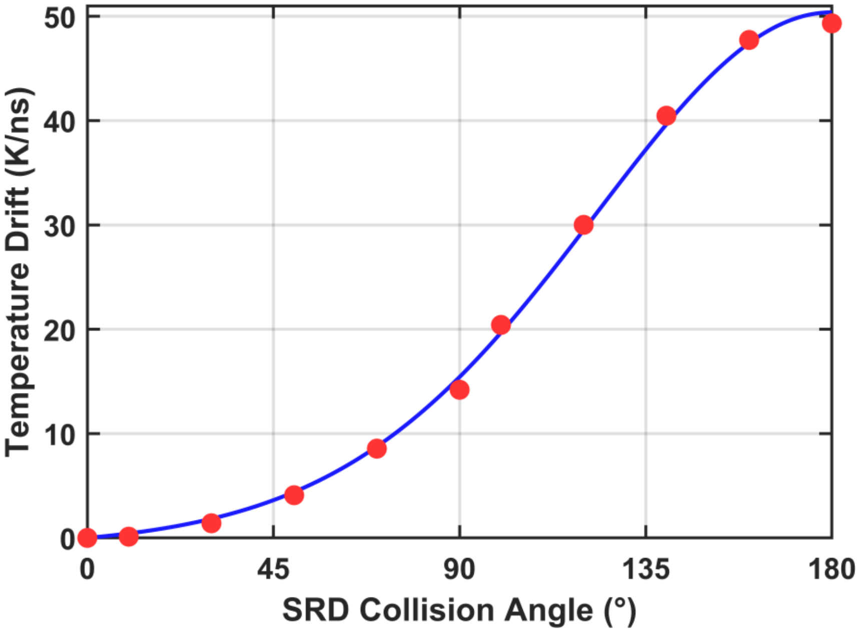

Figure 5.8: Temperature increase per nanosecond over 50,000 SRD collision steps for various collision angles. The fit function is $A\exp\left(B\sin(\alpha/2)\right)-1$, which was discovered to closely corresponded with the data when plotted as a function of $\sin(\alpha/2)$ on an ad hoc basis.

Our remedy for this issue was to employ a momentum-preserving cell thermostat (outlined in section 4.3.5) to keep the simulation stable. This thermostat conserves hydrodynamic flows by only operating on the relative velocities during MPC collision steps. For this reason, it can only act upon solvent particles and lipid headgroups that are participating in the collisions. Under conditions which violate energy conservation, selectively thermostating this part of the system drew energy from the membrane, leading to a steady state temperature gap between the SRD fluid and the membrane. The magnitude of this temperature gap depends on both the MD timestep and the collision parameters (see **Fig. 5.9**). These parameters can be selected to minimize the gap but doing so restricts MD timesteps to a maximum of 10 fs and prohibits frequent collisions with shorter timesteps.

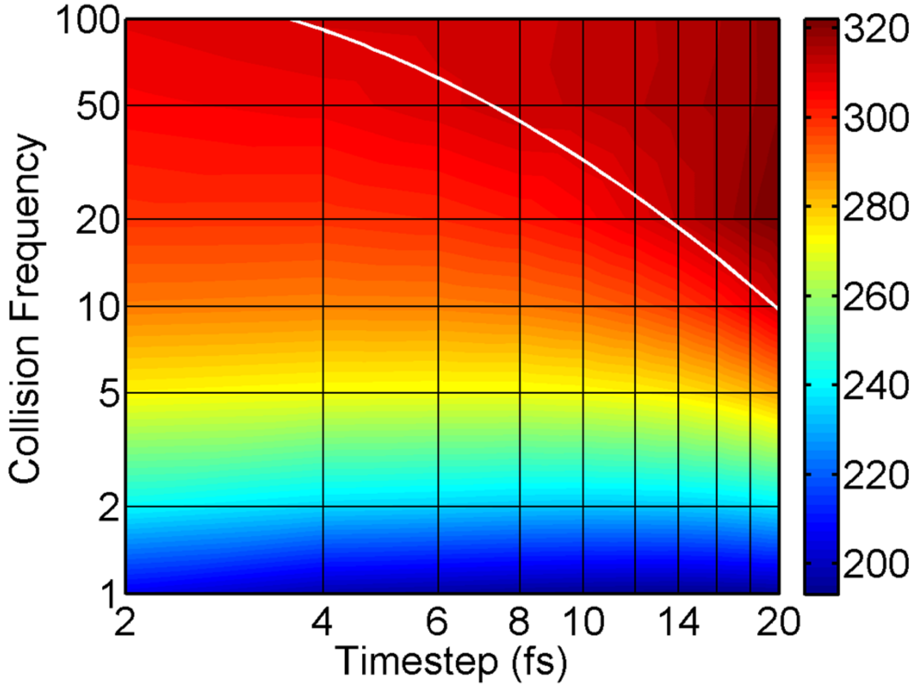

Figure 5.9: Steady state membrane temperature with α = 180° as a function of MPC collision frequency and timestep with the SRD thermostat set to 310 K. The white contour indicates combinations of these parameters that achieve 310 K for the membrane.

Only later did we realize that our implementation of STRD contained a subtle, but serious integration bug; the leapfrog equations of motion were not properly modified to include the MPC collision step. The half-step (midpoint) velocities were perturbed during collisions based on the on-step position of MPC particles according to the following procedure

$$\vec{v}_{n+1/2} = \vec{v}_{n-1/2} + \frac{\Delta t}{m}\,\vec{F}_n(\vec{x}_n) \tag{5.5}$$

$$\vec{v}'_{n+\frac{1}{2}} = \vec{u} + \mathrm{R}\left(\vec{v}_{n-\frac{1}{2}} - \vec{u}\right) \tag{5.6}$$

$$\vec{x}_{n+1} = \vec{x}_n + \Delta t\,\vec{v}'_{n+\frac{1}{2}} \tag{5.7}$$

This modification does not impact simulations containing only MPC particles but is important for simulations which include particles coupled to MPC when those particles also have forces acting on them. This is because applied forces are calculated based on the on-step particle positions, while impulse forces imparted by MPC collisions are applied during the half-step (i.e. the half-step velocities are modified). This inconsistent treatment violates conservation of energy, leading to a substantial heating rate in STRD membrane simulations with a magnitude dependent on both collision angle and integration timestep. The proper way to incorporate MPC collisions is to perform the collisions using on-step velocities. This is done with a modified velocity Verlet integrator

$$\vec{v}_{n+1/2} = \vec{v}'_n + \frac{\Delta t}{2m}\,\vec{F}_n(\vec{x}_n) \tag{5.8}$$

$$\vec{x}_{n+1} = \vec{x}_n + \Delta t\,\vec{v}_{n+\frac{1}{2}} \tag{5.9}$$

$$\vec{v}_{n+1} = \vec{v}_{n+\frac{1}{2}} + \frac{\Delta t}{2m}\,\vec{F}_{n+1}(\vec{x}_{n+1}) \tag{5.10}$$

$$\vec{v}'_{n+1} = \vec{u} + \mathrm{R}(\vec{v}_{n+1} - \vec{u}) \tag{5.11}$$

After this change, STRD simulations no longer heat over time. They do not perfectly conserve energy – simulations still gradually cool – but the rate of temperature change when using 20 fs timesteps is reduced by three orders of magnitude. The cell thermostat is still employed to maintain constant temperature in the solvent without disrupting hydrodynamic flows. As shown in **Fig. 5.10**, the steady state temperature gap between the thermalized SRD fluid (maintained at the prescribed temperature) and the membrane (at a colder temperature when using the erroneous implementation) is now negligible. Since it is no longer necessary to tune integration and collision parameters to minimize the temperature gap, a wider range of accessible parameter space is now available. In particular, the erroneous implementation could not tolerate timesteps in excess of 10 fs. Thus, in addition to being correct, the ability to use a longer timestep – 20 or 30 fs – without driving the simulation far out of equilibrium yields a considerable performance improvement.

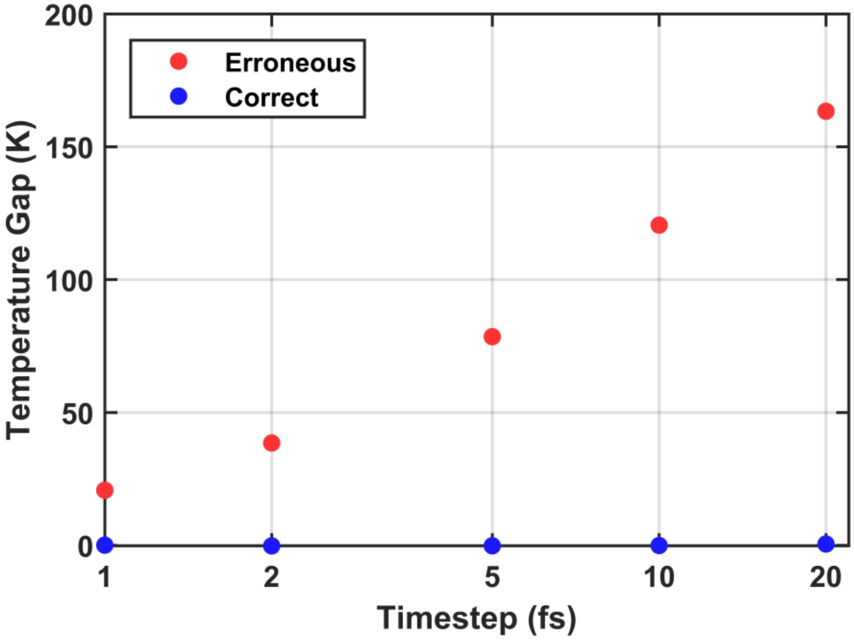

Figure 5.10: Steady state temperature difference in STRD simulations (T_{fluid} – $T_{membrane}$) as a function of timestep when a thermostat is applied to the solvent with correct and incorrect integration procedures.

5.5 Validation

After implementing STRD Martini in GROMACS and ensuring that the relevant conservation laws are followed in every collision cell during each SRD collision step, a series of simulations were run to assess whether our implementation reproduces various theoretical properties of the mesoscopic fluid. These tests are necessary to demonstrate the correctness of our implementation and to determine a valid range of collision parameters. In the following sections, we examine the equilibrium speed distribution of SRD particles under NVE conditions, the average temperature and fluctuations of the cell thermostat, two important dimensionless hydrodynamic numbers, the shear viscosity for both collision rules, and the effective hydrodynamic cutoff length determined through transverse velocity correlations.

5.5.1 Particle Velocity Distribution

It has been shown that the equilibrium velocity distribution of SRD particles is Maxwellian [136]. To verify that our dynamics are correct, we prepared a 30 nm box of SRD particles at moderate density ($N = 2.5$ nm^{-3}) with a uniform initial velocity of 327.7 m/s in random directions. This is simply the thermal velocity for 72 amu SRD particles at 310 K. This initial state corresponds to a delta velocity distribution. This system was simulated for 1000 SRD collisions using the standard SRD collision parameters and no thermostat (NVE ensemble). The resulting distribution after just 5 SRD collisions is shown in **Fig. 5.11**.

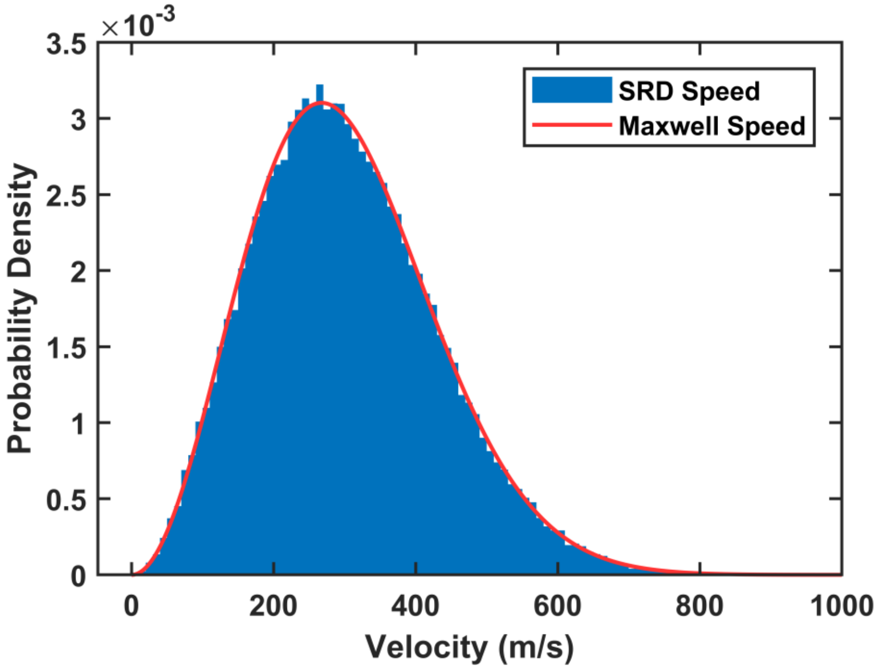

Figure 5.11: The microcanonical speed distribution at 310 K for 67500 SRD particles in a 30 nm box ($N = 2.5$ nm^{-3}) after just 5 collisions, starting from a delta speed distribution with all particles having $v = 327.7$ m/s. The Maxwell speed distribution for SRD particles at 310 K is shown in red.

5.5.2 Thermostat Properties

Two simple tests are required to evaluate our implementation of the cell thermostat from section 4.3.5 using a homogenous SRD fluid. First, a 10 ns simulation is run with the thermostat using a prescribed temperature of 310 K and a strength parameter $\varepsilon = 0.1$. The resulting temperature time series is shown in **Fig. 5.12**. This shows that our thermostat achieves the expected average temperature. Second, a well-known result from thermodynamics is that energy fluctuations for a monatomic gas in a canonical (NVE) ensemble are given by

$$\sigma_E{}^2 = \langle (E - \langle E \rangle)^2 \rangle = k_B T^2 C_v = \frac{3}{2} N (k_B T)^2 \tag{5.12}$$

Proper fluctuations are required to model the canonical ensemble, a property which many popular themostats do not achieve [60]. We can use this expression as a criterion in delineating the range of acceptable strength parameters for our thermostat. Fig 5.13 shows the ratio of kinetic energy fluctuations in STRD as a function of the strength parameter. From this figure, we see that the cell thermostat has proper energy fluctuations over the range $0.03 < \epsilon < 1.50$. Using smaller values results in inappropriately small fluctuations while larger values result in fluctuations that are too large, ultimately destabilizing the simulation.

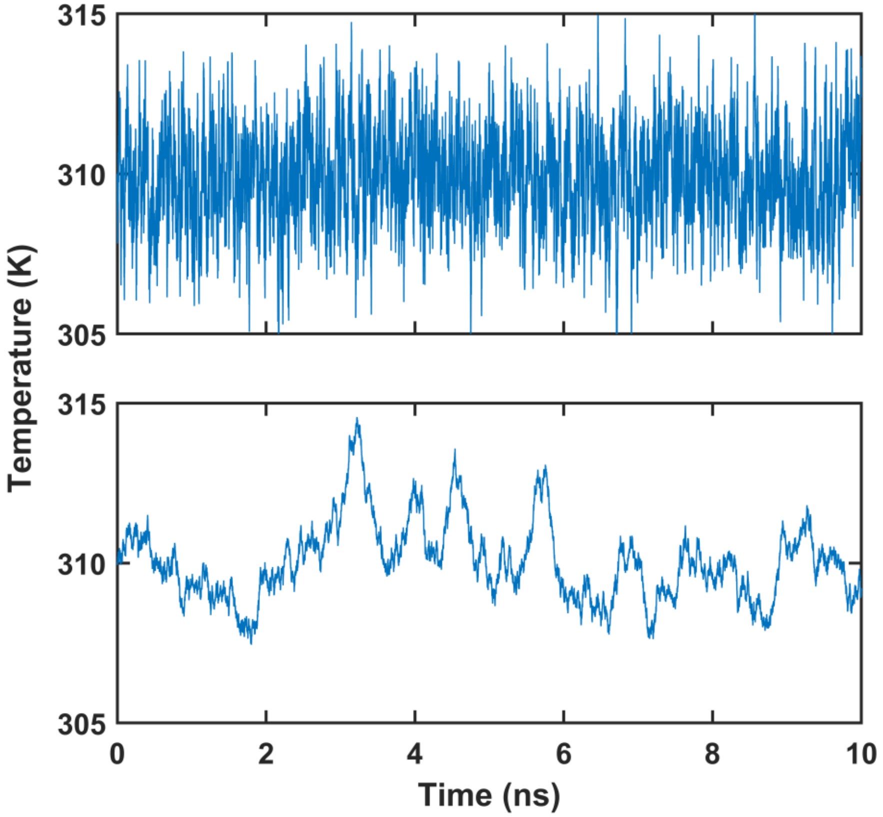

Figure 5.12: Temperature time series for two sample STRD simulations with different strength parameters for the cell thermostat: $\epsilon = 0.1$ (upper) and $\epsilon = 0.01$ (lower). The average temperature differs by only 0.18 K.

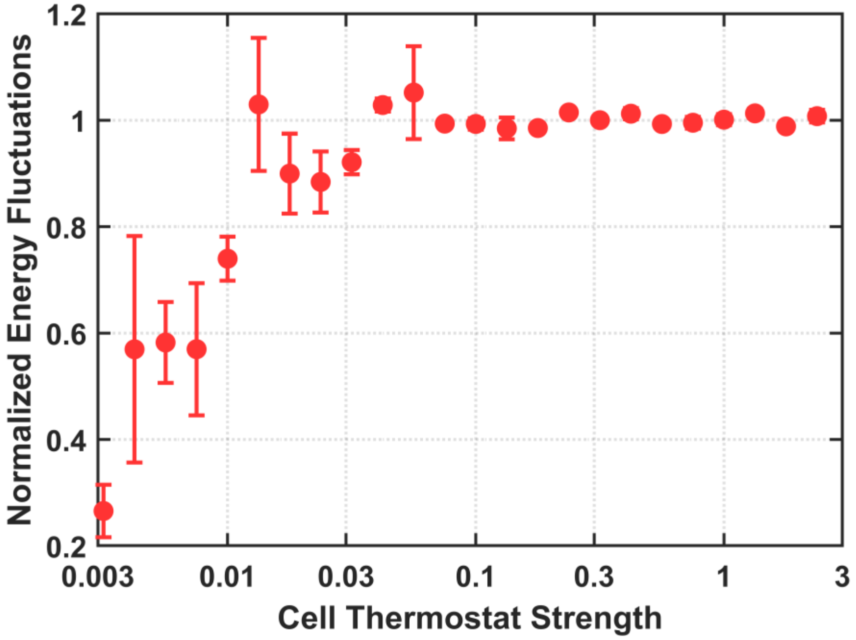

Figure 5.13: Variance in the kinetic energy for an SRD fluid using the cell thermostat at 310 K relative to a reference value calculated with equation 5.12. The energy fluctuations become inappropriately small for values of $\epsilon < 0.03$. Three simulations are run at each thermostat strength, with the error bars depicting standard error for each set.

5.5.3 Schmidt Number

Whether the dynamics of a fluid resemble a gas or liquid is characterized by the dimensionless Schmidt number, defined as the ratio of momentum transport to mass transport. For the SRD fluid, this is given by

$$\mathrm{Sc} \equiv \frac{\upsilon_{col} + \upsilon_{str}}{D_{SRD}} \tag{5.13}$$

Gas-like dynamics correspond to a Schmidt number near unity, since momentum transport in gases is dominated by particle diffusion. In contrast, liquid momentum transport is dominated by collisions between fluid particles, corresponding to large Schmidt numbers. Intuitively, the Schmidt number of an MPC fluid is connected to its mean free path. For an SRD fluid, the most crucial parameters for the Schmidt number are the collision interval and collision angle (see **Fig. 5.14**). Typical STRD collision parameters yield a Schmidt number ≈ 1000, well in the "liquid-like" regime.

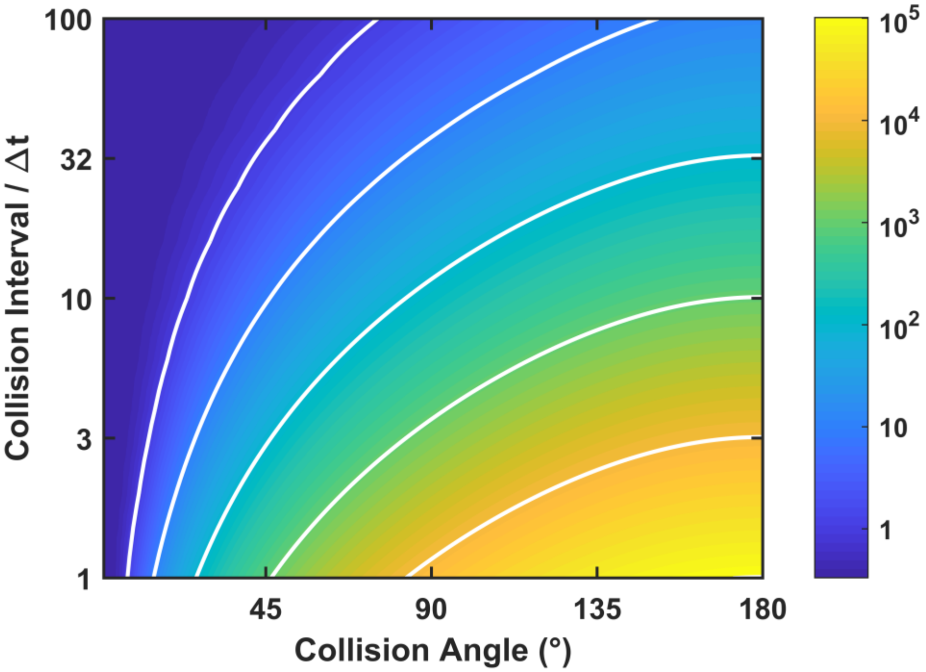

Figure 5.14: Schmidt number for an SRD fluid as a function of collision angle and collision interval in units of MD timesteps. White contours highlight powers of 10 ranging from 1 to 10^4 (as indicated on the color bar).

5.5.4 Mach Number

The compressibility of a fluid is characterized by the dimensionless Mach number, defined as the ratio between fluid flow velocity u and molecular thermal velocity v_t

$$\mathrm{Ma} \equiv u/v_t \qquad (5.14)$$

For monatomic fluids, thermal velocity given by

$$v_t = \sqrt{\frac{5k_B T}{3m}} \tag{5.15}$$

where m is the MPC particle mass. We may assume incompressibility in the limit Ma $\ll 1$, or equivalently $u \ll v_t$. Substituting 310 K and 72 amu gives a thermal velocity of 244 m/s for STRD. To satisfy the well-known incompressibility criterion Ma < 0.3, we are limited to flows with $u < 73$ m/s. Equilibrium studies are well below this limit. For context, the imposed flow fields from section 7.5.1 have a maximum of ~10 m/s. The relevant velocity for computing Mach numbers is the mean flow velocity for some collective motion larger than individual collision cells. In contrast, **Fig. 5.15** demonstrates the distribution of *instantaneous* cell velocities.

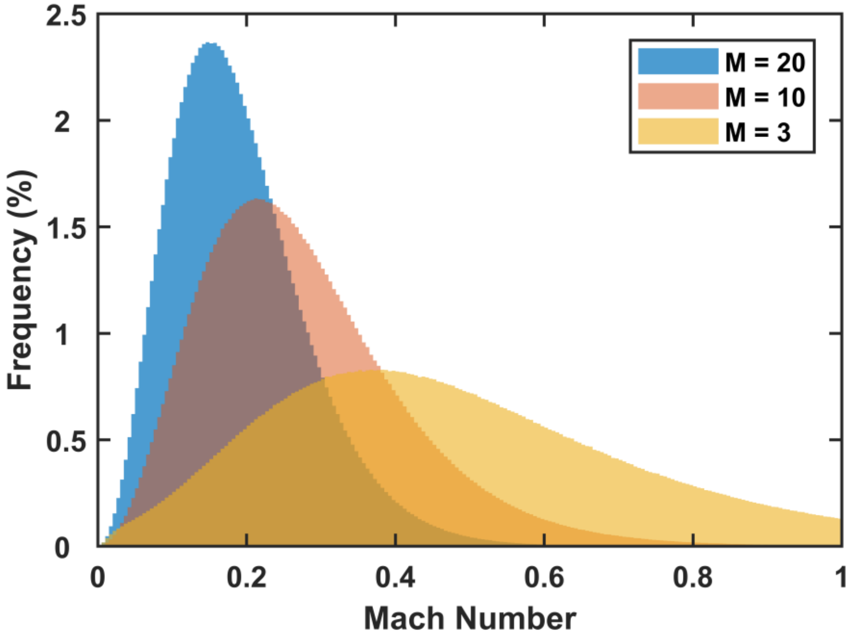

Figure 5.15: Distribution of instantaneous collision cell velocities of quiescent SRD fluids at 310 K at three particle-per-cell densities (M). Higher densities are less prone to wild fluctuations. 20 particles per cell is considered the upper limit on SRD density, while 3 is considered the lower limit.

5.5.5 Reynolds Number

Membrane hydrodynamics are firmly in the low Reynolds number, linearized regime. Thus, it is imperative that our MPC fluid is also modeling dynamics from this regime. The Reynolds number may be written

$$\mathrm{Re} \equiv \frac{\rho u L}{\eta} = \frac{uL}{\nu} \tag{5.16}$$

Where ν is the kinematic viscosity equal to η divided by the density ρ, u is a relevant flow velocity and L is a relevant length scale. For colloidal systems, L might be the radius of the colloid particles. For membranes, the size of extracellular protein domain or the wavelength of membrane undulations may be relevant. The low Reynolds number condition Re $\ll 1$, may also be written $uL \ll \nu$. For typical STRD simulations with water-like viscosity this demands $uL \ll 2 \times 10^6$ nm^2/ns, a condition safely satisfied in nearly any reasonable application. 200 nm/ns flows over 100 nm length scales or some similar combination would be required to realize even 1% of the limiting value for low Re.

5.5.6 Transverse Velocity Correlations

In reciprocal space, the velocity field of a molecular fluid associated with a particular wavevector may be expressed as a sum over particles indexed by α

$$\vec{v}(\vec{k},t) = \frac{1}{N}\sum_{\alpha} \vec{v}_\alpha(t)\exp(i\vec{k}\cdot\vec{r}_\alpha) \tag{5.17}$$

If we define $\vec{v}_T(\vec{k})$ as the transverse velocity (i.e. orthogonal to $\vec{k}$), then we may write normalized time correlations for a particular $\vec{v}_T(\vec{k})$ as

$$C_v^T(\vec{k},\tau) = \frac{\langle \vec{v}_T(\vec{k},t+\tau)\cdot\vec{v}_T(-\vec{k},t)\rangle_t}{\langle \vec{v}_T(\vec{k},t)\cdot\vec{v}_T(-\vec{k},t)\rangle_t} \tag{5.18}$$

For a low-Reynolds number fluid governed by the Stokes equation, these correlations decay exponentially in τ with time constant $= (\nu k^2)^{-1}$. Time integrals of C_v^T converge to values that depend only on the wavenumber and the viscosity [148]

$$T(\vec{k},\tau) = \int_0^\tau C_v^T(\vec{k},\tau')\,d\tau' = \frac{1}{\nu k^2}(1-\exp(-\nu k^2 t)) \tag{5.19}$$

With limiting values given by $(\nu k^2)^{-1}$ as $t \to \infty$. Note this is the same dependence as the magnitude of the Oseen tensor in reciprocal space

$$\mathbb{T}_\infty^{3D}(\vec{k}) = \frac{1}{\eta k^2}\left(\mathbb{I} - \frac{\vec{k}\vec{k}}{k^2}\right) \tag{5.20}$$

For a continuum fluid, equation 5.19 holds for arbitrary k. This is not the case for molecular or mesoscopic models, where it only holds over the length scales that support continuum hydrodynamic transport. Huang et al [148] found this length scale for MPC fluid to be the hydrodynamic cutoff length discussed in section 4.3.4, reproduced here for convenience

$$\lambda_c = \pi\sqrt{2\tau_c v_{SRD}} \qquad (5.21)$$

We simulated a 60 nm box of 216000 SRD particles (N = 1 nm^{-3}) for 5 ns with a 2 fs MD timestep, sampling velocities every 10 fs using collision parameters a = 2.0 nm, τ_c = 200 fs, and α = 180°. Wavevector-dependent transverse velocities for each frame were computed according to equation 5.17 and used to compute correlation time series as defined by equation 5.18. Plateaus for the integrated time series $T(k,\tau)$ are plotted in **Fig. 5.16**. The simulation results are in excellent agreement with the theory presented by Huang et al [148], demonstrating low-Reynolds number hydrodynamics above the cutoff length scale λ_c and loss of correlation on shorter length scales.

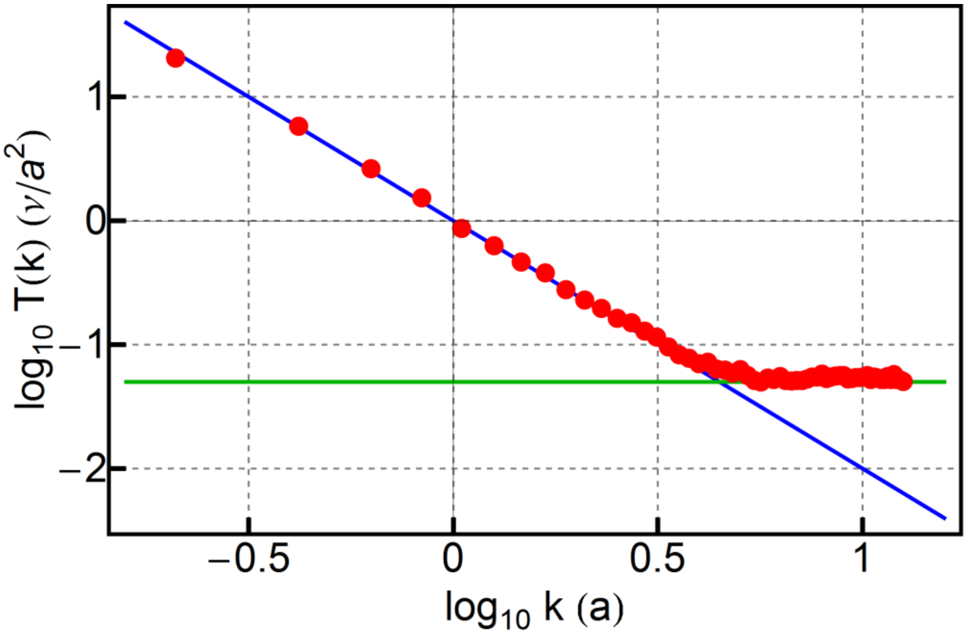

Figure 5.16: Limiting values for integrated time correlation functions $T(k,t)$ of the transverse velocity are depicted as a function of wave number for a 60 nm cube of SRD particles (N = 1 nm^{-3}). Over long length scales (small k), the data follows a $(\nu k^2)^{-1}$ relationship, indicating adherence to low Reynold number hydrodynamic theory (blue). Below the cutoff length scale, $T(k)$ reaches a plateau equal to half the collision interval (green). For the selected collision parameters, the cutoff length scale is 2.77 nm, in close agreement with the simulation data.

5.5.7 Shear Viscosity

As described in sections 4.3 and 4.4, the shear viscosity of an MPC fluid can be expressed analytically for both SRD and MPCA collision rules as a function of their collision parameters. This provides an excellent test for validating STRD Martini while simultaneously demonstrating its useful viscosity-tuning capabilities. There are many ways to calculate viscosity from MD simulations [190], some at equilibrium and some in driven systems. Equilibrium methods are based on evaluating time correlation functions of either the pressure tensor or particle velocities.

Since the STRD Martini fluid does not have a properly defined pressure in GROMACS and our velocity correlations have been shown to agree with low-Reynolds number theory, calculating viscosity from transverse velocity correlations is the best approach [191]. This is done by fitting the decays in $C_v^T(\vec{k},\tau)$ as a function of the lag time τ (see section 5.5.6) and extracting viscosity from the fit. Wavevectors with equal magnitude are averaged over, with the largest wavevectors giving the best fit. GROMACS has a built-in tool for computing the viscosity in this manner from simulation trajectories called *tcaf*.

Viscosity for the SRD collision rule was tested using a 30 nm homogenous box of MPC particles (N = 2.5 nm^{-3}) simulated under NVE conditions at different collision angles with the other collision parameters held fixed at a = 2.0 nm and τ_c = 80 fs. The positions and velocities were sampled every 2 ps for 5 ns and the resulting trajectory was analyzed with *tcaf*. The results are presented in **Fig. 5.17**. Excellent agreement with theory (equation 4.12), with a slight divergence from theory at high collision angles. This whole procedure was repeated with the cell thermostat enabled, and comparably excellent agreement was found with theory (not shown). Using the cell thermostat does not change SRD shear viscosity.

Viscosity for the MPC-A collision rule was calculated in an analogous fashion using a higher density MPC fluid (N = 7 nm^{-3}). Both the collision cell size and collision interval were varied for simulations in the NVT ensemble (recall MPC-A is always inherently thermalized) for 5 ns, taking position and velocity samples every 2 ps. Results are shown in **Fig. 5.18**. Once more, excellent agreement is found with theory (equation 4.19). When the same particle density and equivalent collision parameters are used, MPC-A achieves a viscosity c.a. 75% of what SRD collisions achieve at $\alpha = 180°$ and c.a. 150% of what is achieved at 90°.

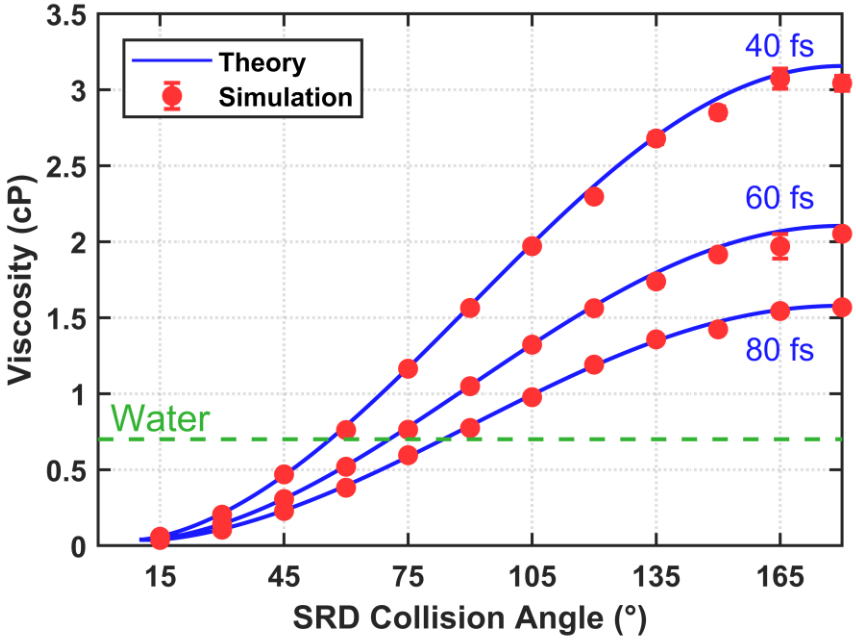

Figure 5.17: Shear viscosity of an MPC fluid ($N = 2.5$ nm^{-3}) using the SRD collision rule as a function of collision angle for three different collision intervals. Theoretical predictions given by equation 4.12. Viscosity of water at 310 K is shown for reference. The collision cell size was 2.0 nm. Simulation values are calculated from of fits to transverse velocity autocorrelation decays using the GROMACS *tcaf* tool.

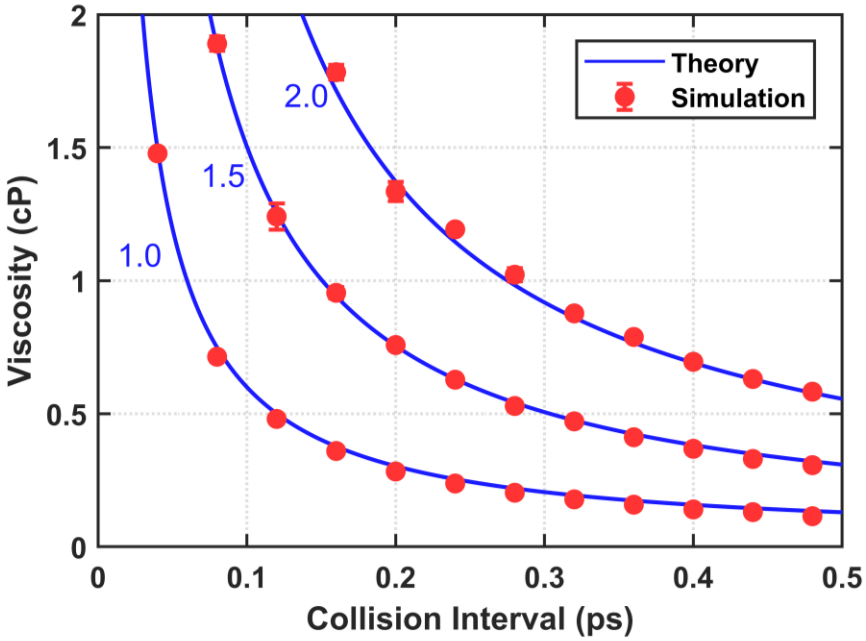

Figure 5.18: Shear viscosity of an MPC fluid ($N = 7$ nm^{-3}) using the MPC-Andersen collision rule as a function of collision interval for three different collision cell sizes. Theoretical predictions are given by equation 4.19. Simulation values are calculated from of fits to transverse velocity autocorrelation decays using the GROMACS *tcaf* tool.

5.6 PSD Finite Size Effect on Lipid Diffusion

With the dynamics of the SRD fluid validated and the assurance we are adequately modeling Stokes hydrodynamics, we may return to the problem of diffusion in a periodic membrane. To illustrate the importance of hydrodynamic interactions for membrane simulations, we sought correspondence with the periodic Saffman-Delbrück theory [44] described in section 3.5. Proteins have not yet been incorporated into STRD Martini, so we investigated self-diffusion of lipids as the periodic box size is varied. Whether lipids can be adequately modeled as cylinders spanning a single leaflet is an open question. Regardless, looking for this finite size effect is a useful test for STRD Martini.

Seven POPC membranes were prepared for both STRD and Dry Martini with a fixed vertical size of 25 nm and lateral sizes ranging from 10 to 40 nm. Each system was simulated for 2 μs at fixed volume with COM motion removed at every step. All used the velocity-Verlet integrator. This affords a direct comparison between similar membranes with and without the hydrodynamics modeled by MPC. STRD Martini membranes were thermalized by coupling MPC particles to the cell thermostat, while Dry Martini simulations used the Bussi v-rescale thermostat. STRD simulations used collision coupling, including POPC phosphate beads in the collision step, and the SRD collision rule with $N = 2.35$ nm^{-3}, $a = 2.0$ nm, $\alpha = 180°$, and $\tau_c = 200$ fs, yielding a solvent viscosity of 0.593 cP.

Diffusion coefficients were calculated according to the method described in section 2.7. Each simulation trajectory was divided into ten 200 ns subtrajectories. The MSD for each subtrajectory was independently calculated using the GROMACS *g_msd* program. Results are reported in **Table 5.5** with uncertainties given by the standard error of each ten calculations. Fitting these results to the PSD theory was accomplished by computing theoretical diffusion coefficients for the simulated system sizes with varying surface viscosity and interleaflet friction using equations 3.49 and 3.51. The “best fit” was found by minimizing the sum over all system sizes of the squared residuals. Parameter space was initially scanned logarithmically over ± 2 orders of magnitude, starting from reasonable initial values, and rapidly narrowed down to the values reported in **Table 5.6**.

The initially assumed size of the lipids ($R = 0.448$) is the radius of a circle with an area equal to 0.63 – the APL for POPC in STRD Martini. Since this is a rather crude model of a lipid, other effective radii were considered: half, double and triple the initial size. One concern arises from the fact that these simulations were performed before the integrator bug was fixed. The temperature gap between solvent and membrane was small in the STRD simulations (less than 2 K). With the bug in place, STRD still passes the validation tests in section 5.5 which suggests it will not have a significant influence on the dynamics if it can be kept under control.

Results for these fits using the half-cylinder model are presented in **Fig. 5.19** and using the membrane-spanning cylinder in **Fig. 5.20**. Results for Dry Martini are presented in **Fig. 5.21**. Qualitatively, the STRD data differs from the Dry Martini data, with self-diffusion reaching a plateau around 25 nm for STRD and continuing to increase for Dry Martini. In theory, Dry Martini membranes are subject to 2D hydrodynamic theory and their self-diffusion coefficients should diverge at large system size due to the Stokes paradox. In simulation, a limit will be reached when the underlying assumptions of Stokes hydrodynamics begin to break down (e.g. the lack of explicit time dependence).

Table 5.5: Lateral self-diffusion coefficients for POPC lipids calculated for Dry and STRD Martini as a function of lateral system size.

L (nm)	STRD Martini D_{POPC} (10^{-7} cm^2/s)	Dry Martini D_{POPC} (10^{-7} cm^2/s)
10	2.060 ± 0.041	3.069 ± 0.108
15	2.211 ± 0.063	3.367 ± 0.078
20	2.350 ± 0.067	3.410 ± 0.100
25	2.397 ± 0.075	3.591 ± 0.155
30	2.516 ± 0.083	3.836 ± 0.077
35	2.471 ± 0.113	3.751 ± 0.136
40	2.441 ± 0.053	3.871 ± 0.091

Table 5.6: Periodic Saffman-Delbrück fit parameters for STRD Martini POPC using various effective lipid radii using both the full-cylinder and half-cylinder expressions for D_{PBC}.

	Half Cylinder		Full Cylinder
R (nm)	η_m (10^{-8} P-cm)	b (10^5 P/cm)	η_m (10^{-8} P-cm)
0.224	7.99	8.39	4.84
0.448	6.23	5.82	3.82
0.896	4.32	3.27	2.80
1.344	3.32	2.30	2.21

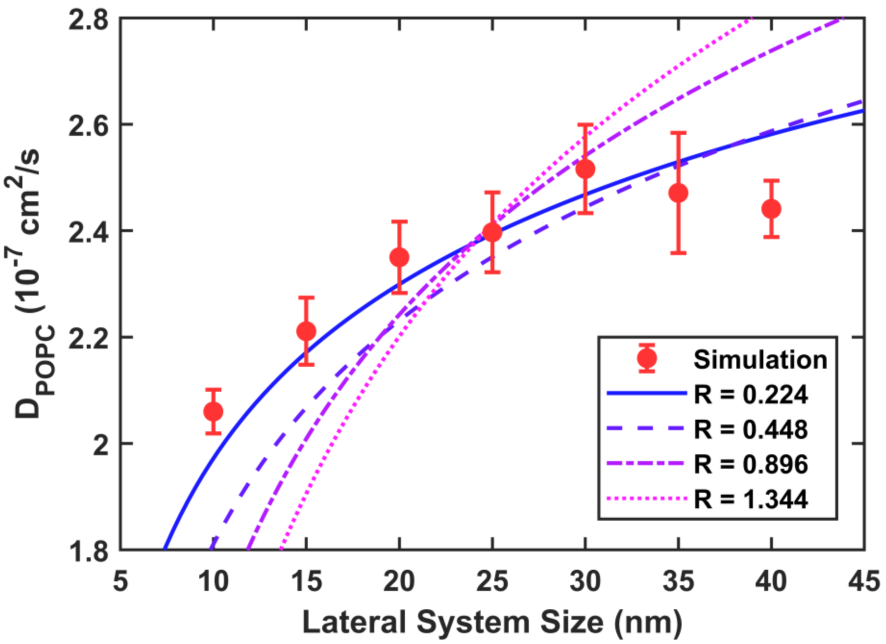

Figure 5.19: Self-diffusion of POPC lipids as the lateral box size is varied for STRD Martini. Fits from PSD theory modeling the lipids as half-membrane-spanning cylinders are shown for four different effective lipid radii. The best fit is obtained when using a radius half the size suggested by the APL. Clean fits are not possible using larger radii.

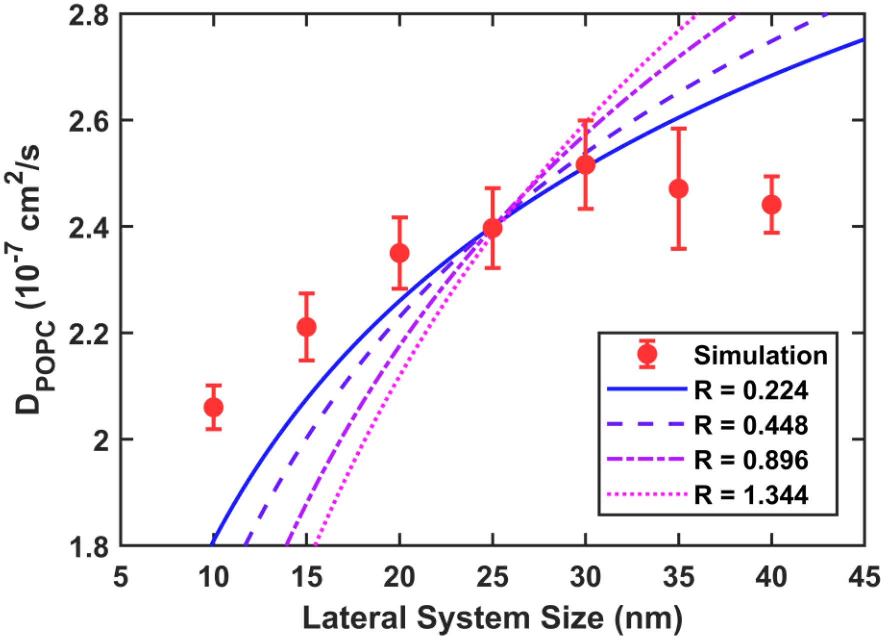

Figure 5.20: Self-diffusion of POPC lipids as the lateral box size is varied for STRD Martini. Fits from PSD theory modeling the lipids as full membrane-spanning cylinders are shown for four different effective lipid radii. Better fits are obtained for smaller effective lipid radii, but none of the fits are satisfactory.

An acceptable fit to the half-cylinder model is available for STRD Martini when the effective lipid radius is half the initially assumed radius. Parameters for this fit are $\eta_m = 7.99 \times 10^{-8}$ P-cm and $b = 8.39 \times 10^5$ P/cm. The best comparison for these parameters in the literature comes from a calculation for DPPC using an older version of Martini with non-equilibrium molecular dynamics [192]. They found $\eta_m = 1.2 \times 10^{-8}$ P-cm and $b = 2.4 \times 10^5$ P/cm. There are several plausible reasons for why these parameters ought to be higher in STRD Martini. (1) Experimentally, the viscosity of POPC membranes is higher than DPPC membranes due to the unsaturated tail of POPC. (2) Interactions between lipids were significantly less attractive in the older version of Martini used in their study. This is even more pronounced for Dry Martini, which requires extra attractive interactions due to the removal of the solvent. Higher attraction between lipids is expected to increase both the surface viscosity and the interleaflet friction. (3) Collisional coupling to the SRD fluid increase the surface viscosity through the introduction of noise (as seen in figure 5.6).

No acceptable fits are available using the full cylinder model for STRD Martini or with either model for Dry Martini. Extending the range of system sizes would be a useful follow-up on this analysis. During the initial investigation of this problem, STRD suffered from an inefficient implementation which prohibited the simulation of very large systems. The code has been greatly improved in the interim, rendering this problem a prime candidate for further study.

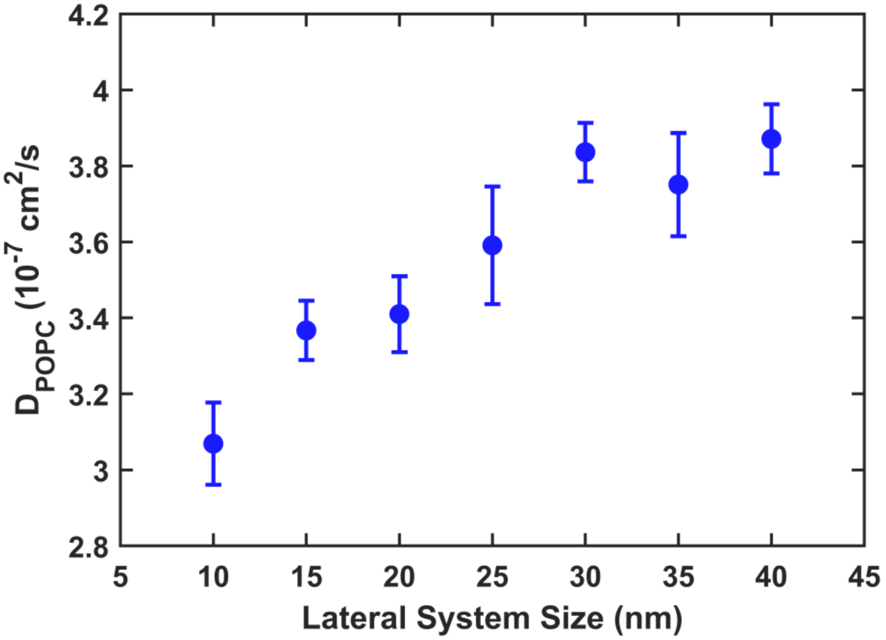

Figure 5.21: Self-diffusion of POPC lipids as the lateral box size is varied for Dry Martini. Note that the diffusion coefficient does not plateau.

Chapter 6

STRD MARTINI PARALLELIZATION

6.1 GROMACS Parallelization

In parallel environments, GROMACS uses domain decomposition (DD) to partition the simulation volume into mutually exclusive triclinic regions called domains [193]. Each domain is assigned to a single processor, which is responsible for integrating the equations of motion for the particles within. Particles are only ‘visible’ to the processor they are assigned to; no one process has access to the complete state of the system. The domains are arranged in a staggered 3D grid of cells along the *x*, *y*, and *z* axes, as depicted in two dimensions by **Fig. 6.1**. This method of decomposing the simulation applies naturally to MD simulations since most of the interactions are short-ranged when cutoffs are used (apart from electrostatics, which are efficiently treated in parallel with PME).

Inter-process communication is required to compute forces between particles residing within different domains. GROMACS uses the eighth-shell method [194] to determine which domain is responsible for resolving these interactions. Using this approach, forces between particles in different domains are calculated on the domain with the lowest index for each coordinate. For example, interactions between particles on domains (3,2,0) and (2,2,1) are computed by the process responsible for domain (2,2,0). This requires atom coordinates be communicated “downward” to domains with lower indices and the resulting forces communicated back “upward” to their original domains. GROMACS accomplishes this with a series of unidirectional “pulses” that send data along each coordinate axis in sequence. In some cases, multiple pulses are required for one or more axes. **Fig. 6.2** demonstrates this procedure with red arrows for pulses along x and green arrows for pulses along y. Note that the pulses must wrap around the boundaries to respect periodic boundary conditions. In this figure, short-range interactions between the particles in domain (1,2) and (2,1) will be handled by the process responsible for domain (1,1) after a single pulse along x followed by a single pulse along y. Resulting forces are communicated back to the particles through a second set return pulses in reverse order.

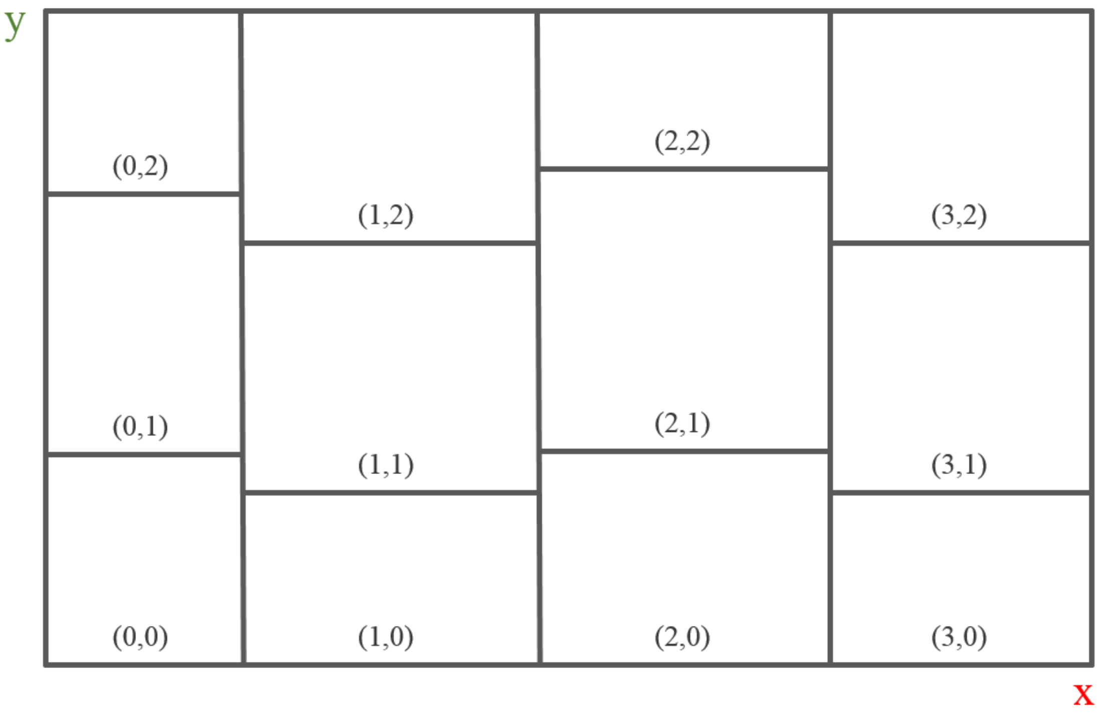

Figure 6.1: A 2D domain decomposition grid staggered in the *y* direction. Each domain is labeled with an ordered pair of grid indices. Note that the domains are generally staggered in the third dimension as well.

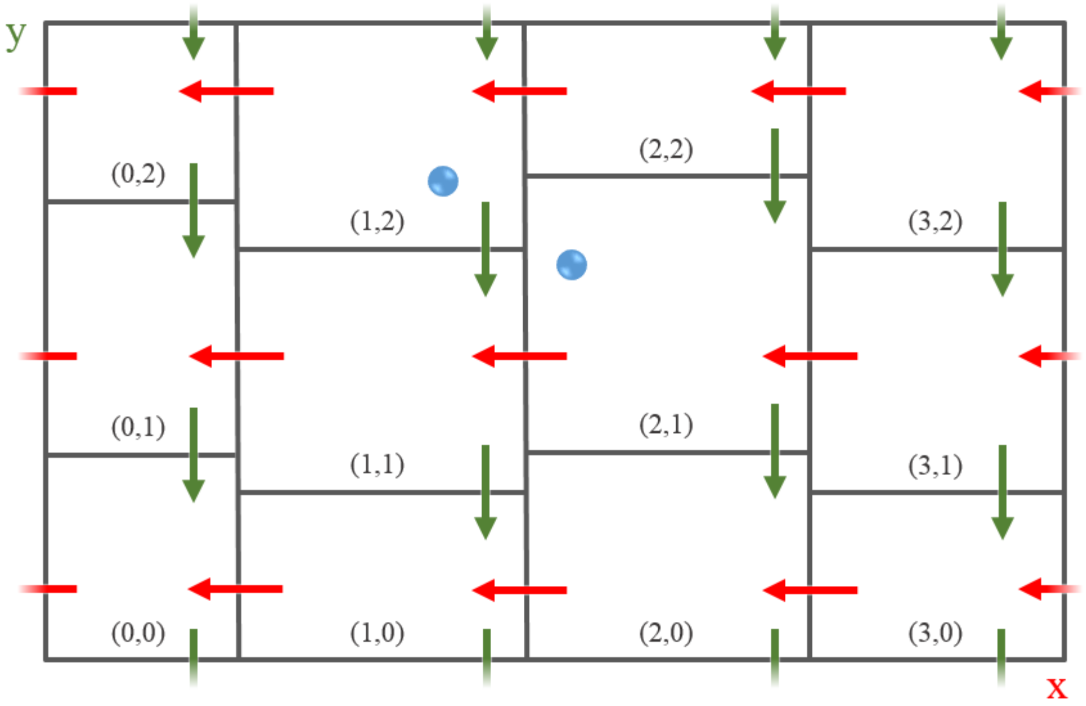

Figure 6.2: Domain decomposition grid overlaid with arrows demonstrating the flow of data in pulsed communication calls.

GROMACS generates a suggested domain decomposition grid when a simulation is initiated. The chosen domains all have the same volume under the assumption that the simulation has a roughly uniform density of interaction sites. Since the required communication scales with the domain surface area, the optimal domain grid is a lattice of cubic domains. As the simulation runs, each process tracks how much time it spends idle and the domains may expand or contract to most efficiently utilize the available computational resources. Overworked domains contract, giving a portion of their workload to their neighbors. This is called dynamic load balancing and it results in a more efficient allocation of computational resources for inhomogeneous systems or those with inhomogeneous interaction densities, such as simulations with united atom force fields. Particles must be periodically reassigned to new domains as they diffuse away from their initially assigned domains. This step is called repartitioning. By default, it is a done whenever the neighbor lists are updated.

GROMACS implements inter-process communication by linking to a Message Passing Interface (MPI) library (the specific library depends on the computing resource). MPI defines a set of operations for communicating between processes including point-to-point and collective operations. Point-to-point operations (such as MPI_Send) transfer data between two particular processes. Collective operations transfer data to a group of processes from either a single process (such as MPI_Broadcast) or among all processing in the group (such as MPI_AllReduce). The latter is called all-to-all communication. GROMACS uses point-to-point operations to implement the communication pulses required for force computation. Collective operations are used when building neighbor lists, calculating global quantities (energy, pressure, etc.), and outputting data. However, they must be sparingly if the goal is to optimize parallel efficiency.

6.2 MPC Communication Requirements

Inter-process communication is required for MPC collisions because some collision cells are necessarily split among different domains (**Fig. 6.3**). The collision cell grid cannot be aligned with the domain decomposition grid over multiple collision steps due to the random grid offset used for each collision step. In theory, one could avoid inter-process communication during the MPC collisions by performing domain repartitioning before every collision step with the new domains constrained to the boundaries of the collision cells. However, the frequency of collision steps is an important parameter of the hydrodynamic model which should remain independent of the frequency of domain repartitioning.

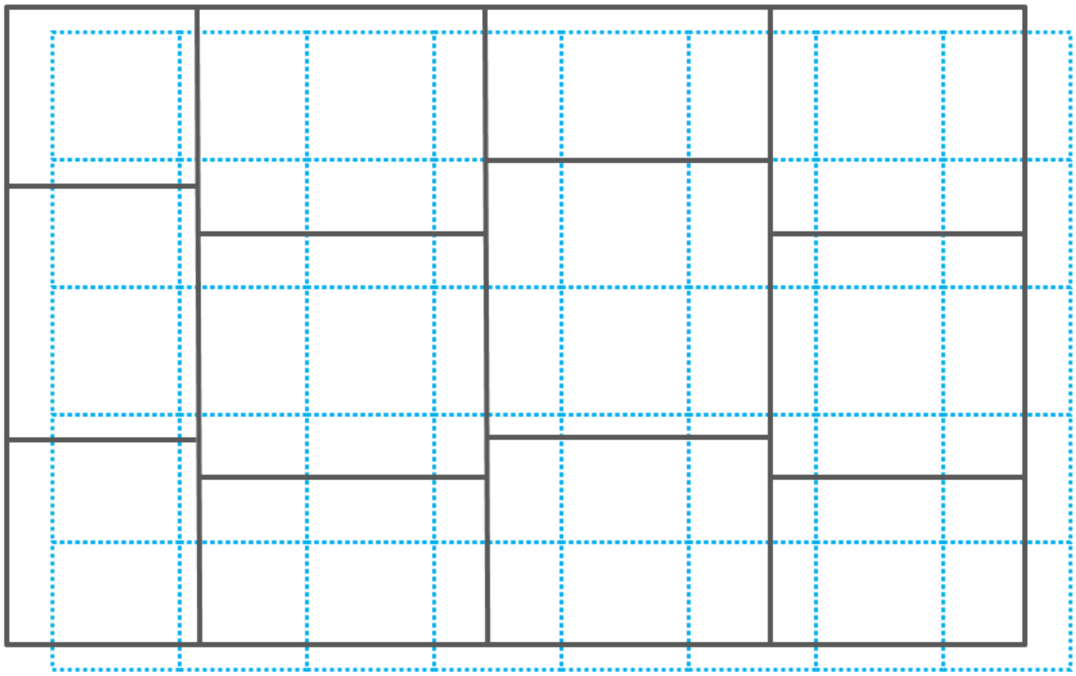

Figure 6.3: The MPC collision cell grid (dashed blue) is shown overlaid with a 3x4 domain decomposition cell grid (grey) in 2D.

Since MPC collisions require communication, two important choices need to be made. First, one must decide which processor is responsible for which collision cells. This must be done in a consistent manner on each processor to ensure there are no conflicts or overlapping claims of responsibility over cells. Every cell must be assigned to a processor. For optimal performance, this should be done in such a way as to minimize collective communication. Second, one must choose what data to communicate between which processors. Simple collision rules involve only the mean velocity of the collision cells, but additional information is required for momentum-preserving thermostats and rules conserving angular momentum. Our communication protocol must support the more general case.

6.3 STRD Communication Protocol

As development of STRD Martini progressed, the communication protocol was iteratively refined. Several approaches were explored before satisfactory parallel performance was achieved for large systems (>100 nm). The performance penalty associated with STRD Martini has three components: (1) book-keeping overhead required by GROMACS to accommodate the MPC tracer particles, (2) the additional computation required for the MPC collision step, and (3) communication required before and after the collision step. Little can be done to optimize the first two components. Book-keeping overhead arises primarily from the need to include the new tracer particles in domain repartitioning and the calculation of 'global' system properties like energy, pressure, and temperature. The upper limit on performance achievable by STRD Martini (henceforth referred to as the ***performance limit***) is set by this book-keeping overhead. This limit is identified by disabling the collision step and its associated communication. Therefore, the performance achieved by STRD Martini hinges crucially on the design of the communication protocol.

6.3.1 Particle Gather and Scatter

The original communication protocol implemented for STRD Martini was designed for simplicity rather than efficiency. The object was to establish a bug-free, rudimentary protocol to use while implementing and debugging other portions of the STRD Martini code. It later proved to be a useful reference for validating the outcome of collision steps involving more sophisticated communication protocols. This method uses collective communication to collect all particle position and velocity data to the master node at the start of each collision step, giving the master process a complete global snapshot of the simulation state. This is the gather operation. The master process is then responsible for identifying all MPC particles, assigning them to collision cells, performing the collisions for each cell, and assigning new velocities to each MPC particle. Once the collision step is complete, the updated MPC particle velocities are communicated back to their original nodes according to the domain decomposition. This is the scatter operation.

Beside the simplicity of its implementation, this protocol has the advantage of being authoritative. Only one process is responsible for assigning MPC particles to collision cells and updating their velocities. There is no potential for conflict or disagreement regarding the outcome of the collision step between different processors because they all receive the same output from the master node. More complicated schemes must devise rules for assigning responsibility over collision cells to different processors and communicating the requisite particle or cell data. Moreover, this protocol does not need to coordinate random number generation across multiple nodes.

Both phases of this communication protocol (gather and scatter) require collective communication involving large transfers of data to and from the master node. In all, nine floating point numbers must be communicated for each particle. The required communication bandwidth scales with the number of particles in the simulation, which is effectively proportional to the total fluid volume for large STRD Martini simulations. Increasing the number of nodes does not decrease the bandwidth required because all particle data must pass through the master node. Furthermore, increasing the number of processors will not reduce the time required for MPC collisions as they are not parallelized. Together, these impose a rather strict bottleneck on the parallel scalability of STRD Martini simulations using this communication protocol.

6.3.2 Partial Cell Sum Reduction

An early alternative approach significantly improved performance and parallel scalability using a very simple communication protocol based on collective MPI operations. This approach exploits the fact that basic MPC collisions only require communication to determine the mean cell velocity vectors $\vec{u}_c$. Once the mean velocity vector for a cell is known, the new velocities of any MPC particle within that cell can be computed without further information about other MPC particles, even those residing on other nodes. Thus, only cell velocity and occupancy sums for each cell need to be communicated among processes. The collisions can then be performed independently on each process. For cells that are split over multiple domains, partial sums are computed on each domain. These partial sums are communicated to other domains and corresponding partial sums are added together. This is the reduce operation, which handles both the communication and the addition. It can be applied to the whole set of MPC collision cells with one call to MPI_AllReduce for the velocity sums and another for the occupancy sums. The mean velocity in each cell is its velocity sum divided by its occupancy sum.

This protocol still requires collective communication, but the bandwidth required is much lower than the gather and scatter protocol. Only three floating point numbers and one integer are required *per cell* rather than nine floating point numbers *per particle*. Since the number of particles per cell is typically chosen to be between 3 and 30, the required bandwidth is at least an order of magnitude smaller with this method. Unfortunately, the use of collective communication still imposes a significant performance bottleneck when dealing with large systems and high levels of parallelization. A local communication protocol based on limited communication between neighboring domains is preferred.

6.3.3 Pulsed Particle Data

Much better performance is available from communication protocols that are based on “pulsed” point-to-point communication, sending and receiving MPC particle data only among neighboring domains in a manner similar to the GROMACS force computation as described in section 6.1. In such arrangements, the communication bandwidth decreases as the number of domains increase since fewer collision cells are shared between any two particular domains. A key objective is the elimination of all collective communication. The particle data that must be communicated depends on the collision rule – just the velocity for simple collisions and the position for those which conserve angular momentum. Additional “routing data” identifying the origin and destination nodes and particle ID is also communicated.

Each collision cell is assigned a “home” domain associated with a particular process that is responsible for handling the collision step of the cell. The set of all collision cells assigned to a domain are called its “local cells”. By convention, the range of local cells for each domain is chosen by rounding the domain decomposition boundaries to the nearest grid-shifted collision cell along each axis. Since some cells are split across the domain decomposition boundaries, some particles from the boundary cells must be communicated among their domain decomposition neighbors to reach their home domain. These particles are said to be “orphans” that are “adopted” by their home domains. After computing the collisions, the resulting velocities of these orphans must be sent back to the domain where they were originally adopted from.

The communication code is inspired by the pulsed communication scheme used by GROMACS during force computation. Data is sent to neighboring domains in pulses, first along +X and -X, then +Y, -Y, +Z and -Z. In this manner, SRD data can be sent or forwarded to any of the 26 surrounding domains.

The communication procedure is depicted in **Fig. 6.4**. At the start of a collision step, each process assigns its local SRD particles to collision cells and determines the home domain of those cells based on the current domain decomposition boundaries and randomized collision cell grid shift. Data for the orphaned SRD particles are copied to a “dispatch buffer” along with routing instructions specifying in which pulse each particle should be sent. At the start of each pulse, the dispatch buffer is read. Any particles that need to be sent in that pulse are removed from the dispatch buffer and placed in the send buffer. The particles are sent to the recipient domain’s receive buffer. Each particle’s routing information is read again. If they belong to the recipient domain, they are copied to the “adoption buffer”, where they will be included in the local collisions. If they belong elsewhere, they are copied to the dispatch buffer of that process and forwarded along. These steps are repeated for each pulse. Once all the communication is done, each process performs the collision step for their local collision cells and the resulting particle velocities are communicated back to their original domains by reversing the process described above.

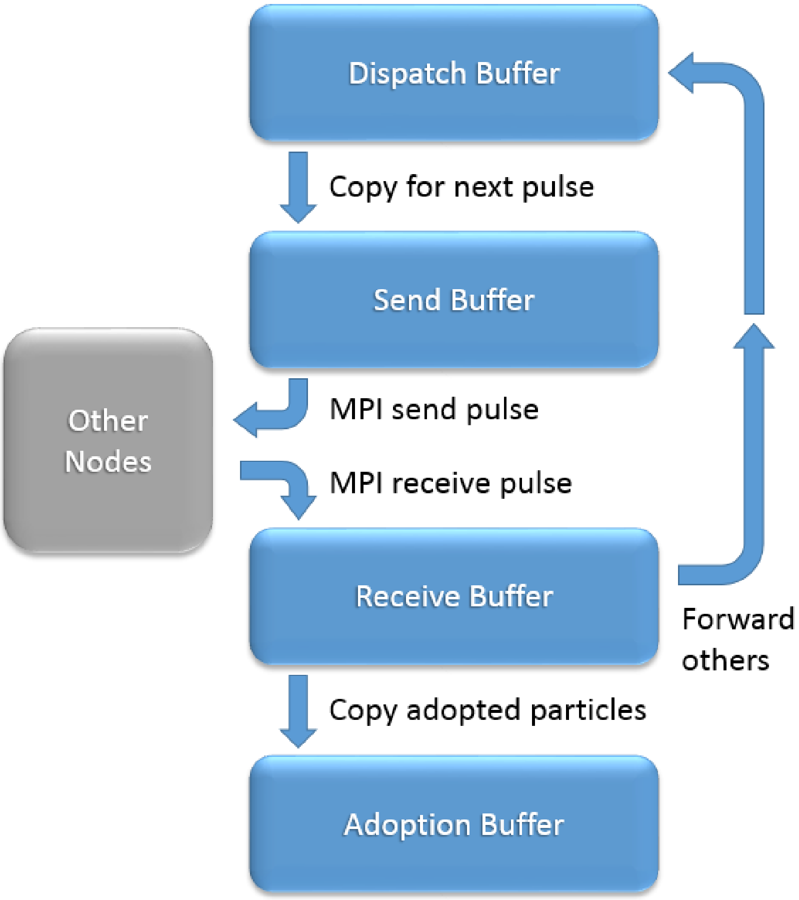

Figure 6.4: Data flowchart for a single pulse in the pulsed particle communication protocol. This process is performed once forward and backward along each dimension.

Since only a single pulse is performed in each direction, this implementation requires all domain decomposition cells to be larger than individual SRD cells, which are typically chosen to be 1-2 nm. This ensures that no SRD particle has a collision cell whose home domain is located more than one domain away in any particular direction. This restriction also requires dynamic load balancing to be disabled when the size of the domains approaches the size of a collision cell because it does not account for staggered domains. Neither of these restrictions are important for STRD Martini simulations because the interaction density is relatively homogenous and the large number of processes required to reach the size limit (625 processes for a 50×50 nm system using 2 nm collision cells).

6.3.4 Pulsed Cell Data

A related scheme based on pulsed point-to-point communication was also implemented which sought to reduce the required communication bandwidth by only sending and receiving partial sums for collision cell data rather than particle data. This is similar to the approach taken by the partial cell sum reduction protocol. Once more, each process may perform SRD collisions locally on its own MPC particles using only the total velocity and number of MPC particles for each cell. This approach can be adapted to other collision rules with the addition of partial sums for mean position, momentum of inertia, angular momentum, etc. The requisite partial sums are combined with their corresponding partial sums from other domains using the same pulsed communication protocol described for the pulsed particle data protocol. After the communication step, each node will have a complete set of sums for all collision cells containing its local atoms. Certain quantities, such as kinetic energy or angular momentum, require position and velocity sums before they can be calculated. Cell sums of these quantities must be computed and communicated in a second pass.

A naïve implementation would require a six separate MPI calls for reduction on each sum: both backward and forward in all three dimensions. In order to minimize the number of MPI calls used by this approach, the partial sums are packed into integer and floating-point buffers which the reduction operates on. The resulting sums are unpacked from the buffers afterward. Sums calculated in the second pass are likewise packed and unpacked in this manner.

This new communication scheme greatly reduces the bandwidth required for the SRD collision rule. Only three floating-point numbers (the velocity sum vector) and one integer (the cell's unique ID) are communicated per cell, compared to the three floating-point numbers and four integers *per particle* in the particle-based communication scheme. A second round of communication is required when the thermostat is enabled or when using a collision rule that conserves angular momentum. As a result, the total number of MPI calls is the same for the particle-based and cell-based local communication schemes. Despite the bandwidth reduction, the performance difference between them is negligible in most circumstances.

The cell-based scheme has one significant drawback: it lacks a reliable method for error detection. Communication errors can occur if dynamic load balancing is enabled and the domain size approaches the size of the collision cells. In such cases, the particle-based communication scheme is able to detect when a particle cannot be properly routed to its adoptive home and report the error. The cell-based method has no analogous mechanism and will happily continue the simulation, occasionally producing subtle errors without alerting the user. Particles for some boundary cells may be missing from the cell sums. For this reason, it is disabled by default in favor of the particle-based scheme.

6.4 Importance of the Domain Decomposition

When domain decomposition is used to partition a system over N processors, an initial grid of domains must be selected with n_i cells in each dimension such that the product $n_x n_y n_z$ equals N. Since there are multiple ways of choosing this grid for any composite-valued N, selecting an optimal configuration is crucial for maximizing performance. The computational workload should be evenly divided among all processors to minimize idle processor time. As this workload predominately involves the computation of pairwise forces between nearby interaction sites, an optimal grid is typically one that evenly divides the number of interaction sites.

For homogenous systems, this implies domains of equal volume. However, parallel performance is also impacted by communication between network nodes, which scales with the surface area of the domains. All else being equal, long skinny domains will require more communication than square domains of an equal volume. In fact, the triclinic geometry which minimizes surface area for a given volume is a cube. This is not the case for inhomogeneous systems, where the relationship between volume and interaction density (and hence computational cost) breaks down. Dynamic load balancing can mitigate this problem by allowing the domains to grow and shrink until an equilibrium is established.

Even this may not be sufficient for particularly inhomogeneous systems, such as the membranes simulated in Dry Martini and STRD Martini. In both cases, the region enclosing the membrane requires far more computation resources than the surrounding vacuum or MPC fluid. Any processing power assigned to the fluid region will sit idle during force computation and be squandered, regardless of how much the domains expand during dynamic load balancing. User-specified domain grids with only a single layer of vertical domains (n_z = 1) are warranted in these cases. For large systems, these domains become long and narrow, increasing the communication required during the MPC communication. A performance trade-off is likely to occur for very large systems, but it has not been identified for systems with z ≤ 100 nm.

6.5 Additional Optimizations

6.5.1 Generating Random Numbers in Parallel

Performing the collision steps locally permits some serial SRD collision code to be parallelized for an additional performance improvement over the original implementation. The most substantial benefit comes from generating random numbers in parallel. SRD collisions with a thermostat require six random numbers per collision cell – three for the randomly chosen axis and three for the cell thermostat. The original implementation generated these random numbers for the entire system at the start of each collision step. This task requires a significant amount of time for large systems since it is proportional to the total number of SRD cells, and hence the simulation volume.

This task has been completely parallelized. Random numbers are now generated locally – only for the local collision cells and their colliding particles. The new implementation uses GROMACS' built-in ThreeFry2x64 random number generator. This random number generator is stateless and counter-based. It takes two seeds and two counters to generate sequences of random numbers. The first seed is chosen to be a constant and the second is available as an input parameter. By supplying the current timestep and collision cell ID as the counters, it is ensured that the same random numbers are generated for each collision independent of the number of MPI ranks and the particular domain partitioning of the system. Thus, SRD dynamics for a particular seed will not change if the same system is run with a different number of processes.

6.5.2 Improved Cell Iteration

Another substantial performance improvement resulted from optimizing the method of collision cell iteration. At various points in the code, the collision cells must be iterated over in order to do such things as assign random rotation axes, compute thermostat rescaling factors, or calculate velocity vectors from velocity and occupancy sums. The original implementation would simply iterate over all collision cells, skipping all non-local cells with a conditional in the body of the loop. Careful performance profiling revealed an inordinate time being spent whenever such iteration was employed. A more efficient method of cell iteration was introduced, iterating over only the range of cells local to each processor. The cell range is saved during the cell assignment step and saved.

6.6 Performance Benchmarks

The cumulative results of these optimizations are presented in **Figs. 6.5** through **6.9**. Performance benchmarks were run on a 168-node cluster composed of 42 Intel Xeon E5630 processors networked with InfiniBand interconnects. One fluid system and three membrane systems were benchmarked. The fluid system was simply a 30 nm cubic box of MPC fluid containing 189000 fluid particles ($N = 7$ nm^{-3}). The membrane systems were all roughly 40 x 40 x 25 nm POPC membranes prepared for Martini, Dry Martini, and STRD Martini. The bilayer in each membrane system contains 2704 lipids per leaflet. In addition, the STRD Martini system is filled with a bath of 100000 MPC fluid particles ($N = 2.5$ nm^{-3}), while the equivalent Martini membrane is solvated with 290448 Martini water beads. All benchmarks presented here use a conservative 10 fsec timestep, reflecting a limitation of the original STRD implementation. All other benchmarks could be run with a 20 fsec timestep for an additional 2x speedup.

Fig. 6.5 compares benchmarks for different versions of STRD Martini simulating a 30 nm cubic box containing 189000 fluid particles ($N = 7$ nm^{-3}) and no membrane. MPC collisions occur every 20 timesteps using the SRD collision rule. Since none of the particles in these simulations have pairwise interactions, there is no force calculation required. The primary performance bottleneck for STRD becomes the communication required during MPC collision steps. The originally published code (labeled “Original”) scales poorly due to the massive communication overhead involved, reaching a plateau around 32 processes. The optimized communication code (labeled “Optimized”) is over three times faster than the original and has yet to reach a performance plateau at 168 processes. The performance limit (labeled “Limit”) imposed by GROMACS indicates a set of simulations which include MPC particles but disable the collision step entirely. The optimized communication code nearly achieves performance limit and scales along with it (see **Fig. 6.6**).

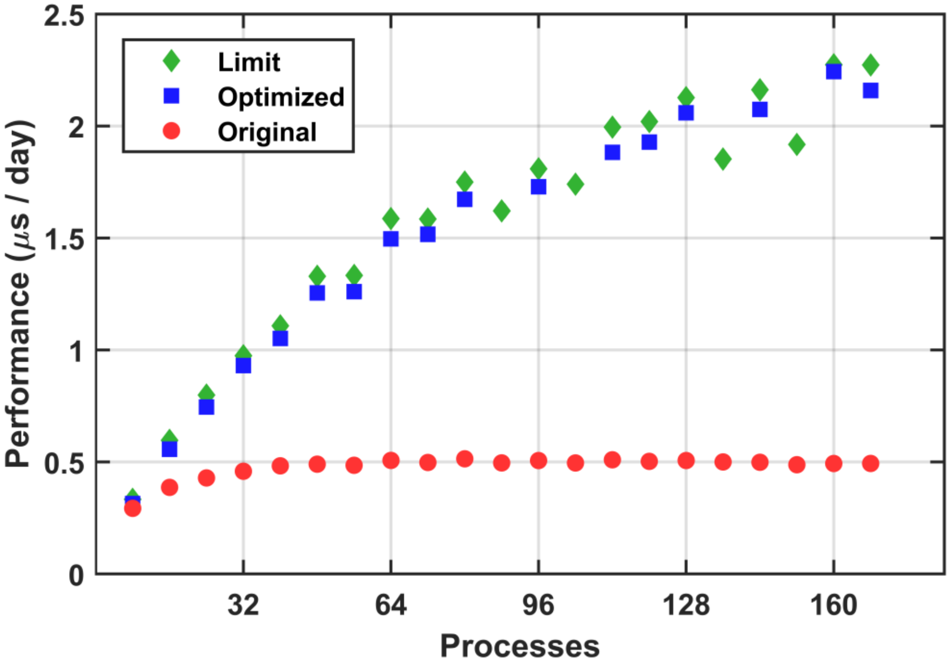

Figure 6.5: Performance benchmarks for a 30 nm cube of STRD fluid.

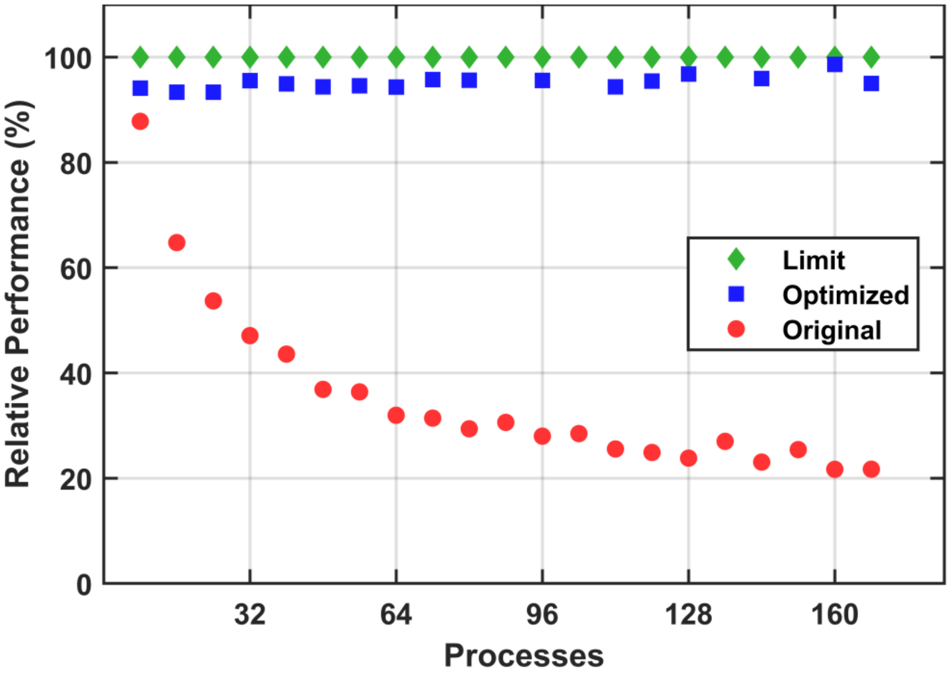

Figure 6.6: Relative performance for a 30 nm cube of STRD fluid.

A broader comparison of performance benchmarks is possible for simulations of membrane systems, since equivalently-sized benchmarks can also be run for ordinary Martini and Dry Martini. These provide context for interpreting the STRD Martini results, setting the upper and lower bounds for expected performance. No matter how much effort is devoted to optimizing STRD Martini, its performance will never exceed that of Dry Martini. On the other hand, comparisons to ordinary Martini are relevant to determine the performance benefit over including hydrodynamic transport via an explicit solvent.

Benchmarks for the aforementioned 40 x 40 x 25 nm POPC membrane systems are presented in **Figs. 6.7** and **6.8** and summarized in **Table 6.1**. Using a 168-node cluster, Dry Martini runs 4.5x faster than ordinary Martini, as a result of removing the Martini solvent particles. When 100,000 SRD fluid particles ($N = 2.5$ nm^{-3}) are added to the Dry Martini system and SRD collisions are disabled, the simulation runs 3.45x faster than ordinary Martini. Once more, this is the maximum performance that can be achieved by optimizing the communication code. Even though STRD particles do not have pairwise interactions between them, they impose additional bookkeeping requirements for GROMACS that result in some degraded performance relative to Dry Martini. In particular, GROMACS must consider the additional particles during domain decomposition, when integrating the equations of motion, and when calculating global quantities of the simulation such as total kinetic energy and pressure. With collisions enabled the optimized communication code runs 3.40x faster than an equivalently sized Martini system. The original communication code was only 0.44x faster than ordinary Martini.

Table 6.1: Performance summary for a 40 x 40 x 25 nm POPC membranes on a 168-processor compute cluster.

Benchmark Simulation	**Absolute Performance (ns/day)**	**Relative Performance (vs Martini)**	**Relative Performance (vs Dry Martini)**
Martini	256.566	1.000	0.181
Original STRD	368.143	1.435	0.260
Optimized STRD	1129.370	4.402	0.796
STRD Limit	1140.336	4.445	0.804
Dry Martini	1418.289	5.528	1.000

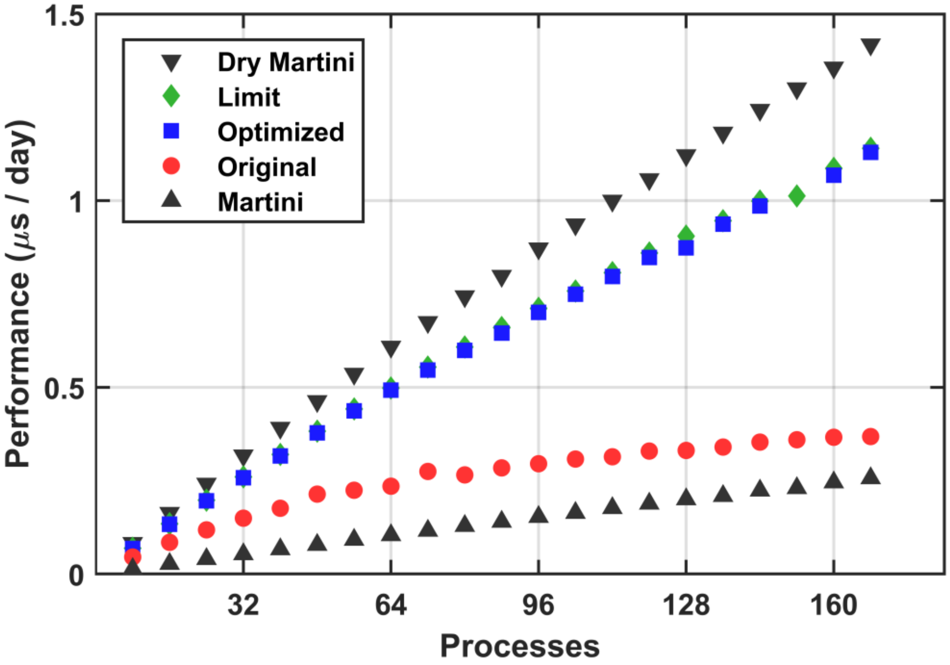

Figure 6.7: Performance benchmarks for 40 x 40 x 25 nm POPC membranes.

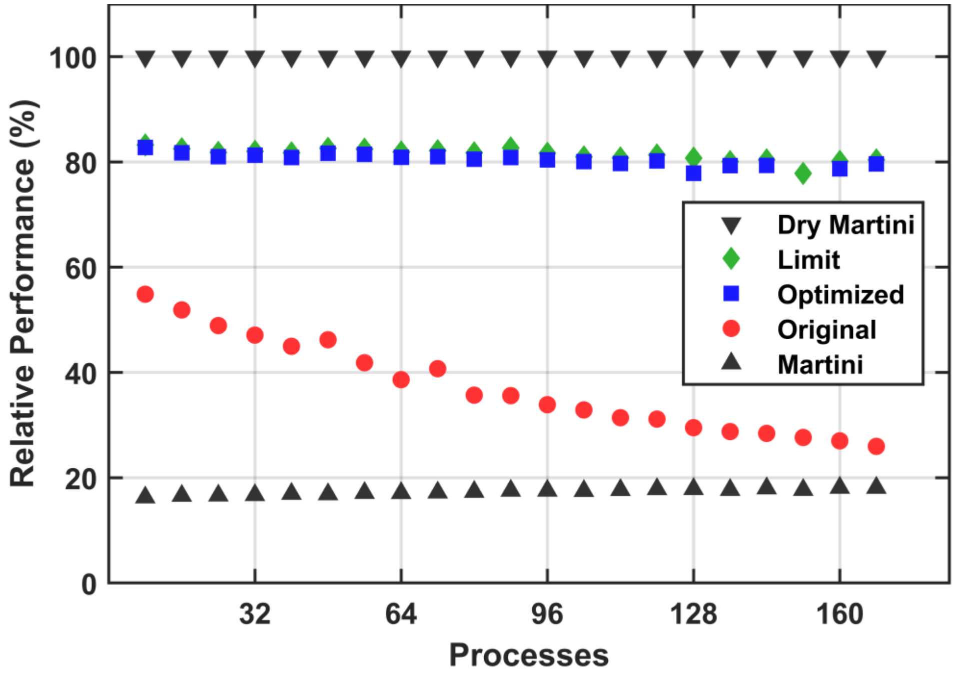

Figure 6.8: Relative performance for 40 x 40 x 25 nm POPC membranes.

As discussed in section 6.4, special care must be taken in choosing the domain decomposition grid for STRD and Dry Martini simulations. In both cases, the absence of a solvent requiring pairwise force calculation introduces significant inhomogeneity in computational cost. This is demonstrated in **Fig. 6.9** by varying the number of cells in the direction normal to the membrane. All STRD and Dry Martini benchmarks are carried out with $n_z = 1$ to maximize performance.

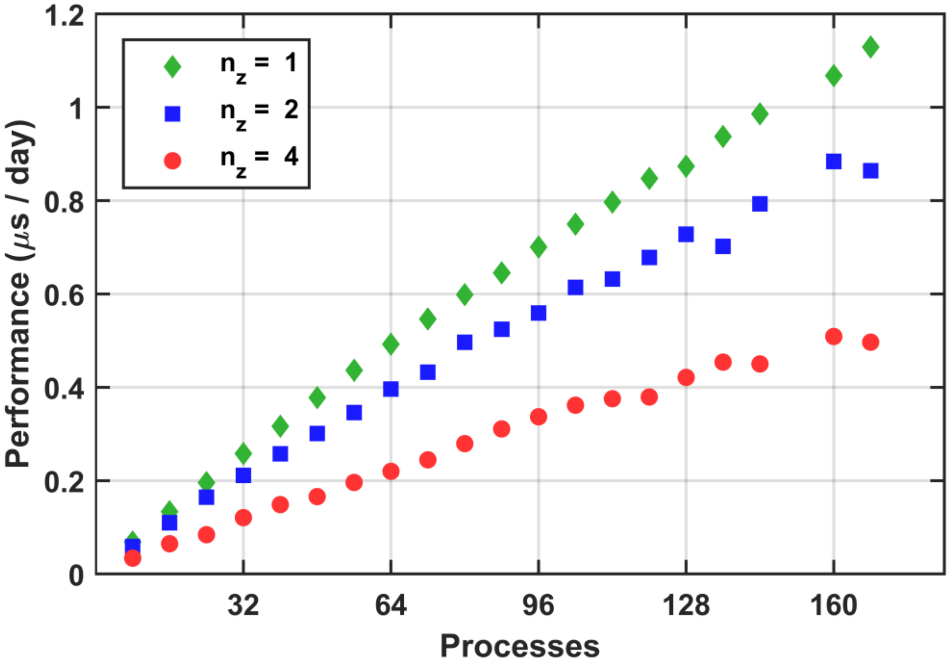

Figure 6.9: STRD performance as number of domain decomposition cells along the direction normal to the membrane is varied.

Additional benchmarks were run on the Stampede2 supercomputer located at the Texas Advanced Computing Center of the University of Texas at Austin. These tests involved much larger systems and higher levels of parallelism. 100 x 100 x 100 nm POPC membranes were prepared for Dry Martini (0.43 million atoms), ordinary Martini (8.47 million atoms), and STRD Martini with $N = 2.6$ nm^{-3} (3.18 million atoms). Each ran on 1024 “Knight’s Landing” (Intel Xeon Phi 7250) cores using the same simulation parameters as the smaller performance tests. The results are presented in **Table 6.2**.

Under these circumstances, ordinary Martini runs at 50 ns/day while Dry Martini runs 16.6x faster at 880 ns/day. In comparison, the optimized STRD Martini code achieves 383 ns/day, which is 7.6x as fast as ordinary Martini or 44% as fast as

Dry Martini despite the addition of 2.75 million SRD fluid particles. The remaining performance gap between STRD and Dry Martini is almost entirely due to the accounting overhead associated with these particles. Running without collisions increases the performance to 393 ns/day, a mere 2.6% increase. The aforementioned overhead stems from the additional time required for domain decomposition, integration, and global energy communication involving the additional particles.

Table 6.2: Performance summary for a 100 x 100 x 100 nm POPC membranes on the Stampede2 supercomputer.

Benchmark Simulation	**Absolute Performance (ns/day)**	**Relative Performance (vs Martini)**	**Relative Performance (vs Dry Martini)**
Martini	50.34	1.000	0.057
Pulsed Cell	380.56	7.560	0.432
Pulsed Particle	382.85	7.605	0.435
STRD Limit	392.83	7.804	0.446
Dry Martini	880.57	17.492	1.000

Chapter 7

CALCULATION OF MEMBRANE FLUID PARAMETERS FOR THE MARTINI AND CHARMM36 FORCE FIELDS

7.1 Introduction

Membrane surface shear viscosity η_m and interleaflet friction b have been measured experimentally for membranes [195]–[199], but few studies have calculated these properties from lipids in simulation [113], [192], [200], [201]. The authors of popular membrane force fields parameterize their models using thermodynamic quantities such as area per lipid or the solvation free energy of small molecules rather than comparatively more difficult-to-measure hydrodynamic properties [52], [54], [55], [103], [106]. As a result, these popular force fields obtain reasonable results for such properties as tail order parameters and area compressibility, but their fluid properties and consequently Saffman-Delbrück lengths for the models are not well-characterized.

Both surface shear viscosity (η_m) and interleaflet friction (b) are key parameters in the periodic Saffman-Delbrück (PSD) theory for lipid diffusion (see section 3.5) [44]. Although these two parameters can be determined through fits to PSD theory [1], [46], an independent means of calculating them is required for the theory to be predictive. The third parameter that enters the PSD theory is the effective hydrodynamic radius of a lipid or membrane protein. This is especially problematic for lipid diffusion, where the effective hydrodynamic radius of a lipid may be significantly larger than its lateral footprint due to correlated motion with its nearest neighbors, and for which it is not even clear that a continuum theory makes sense.

A pair of papers by den Otter and Shkulipa demonstrates a method for calculating η_m and b for Martini DPPC using non-equilibrium simulations which impose either a lateral or tangential shear on the membrane [192], [200]. This chapter utilizes their method to calculate these parameters for different lipids in both the Martini coarse-grain force field and the CHARMM36 all-atom force field. Surface viscosities for CHARMM36 Ld and Lo mixtures are also reported. Interleaflet friction may also be calculated from undulation relaxation rates [120], [201], [202], while surface viscosity has been calculated from simulated tether pulling [113]. Neither approach has been pursued here, as values calculated from shearing simulations show good agreement with equilibrium approaches [192].

Before using non-equilibrium simulations to calculate previously unknown PSD parameters (η_m and b) for a membrane system, we must ensure that our method matches published results for a well-characterized system. Water is an excellent choice, as the viscosity of water in both force fields has been well-established using equilibrium methods [203], [204]. Once a set of best practices is established for calculating the viscosity of water, we turn to more complex systems, starting with Martini DPPC. The results for Martini DPPC will be compared to the values published by den Otter and Shkulipa before moving on to all-atom membranes and mixtures.

7.2 Theory and Methods for Nonequilibrium Calculations

For convenience, symbols used throughout this section are listed in **Table 7.1**. Each are reintroduced in context when they first appear in the text.

Table 7.1: Symbols used in calculations of solvent viscosity, membrane surface viscosity, and interleaflet friction.

$\sigma_{\alpha\beta}$	Viscous stress tensor
$P_{\alpha\beta}$	Virial pressure tensor
η_w	Solvent viscosity
η_m	Membrane surface viscosity
η_m^b	Effective 3D membrane viscosity
h	Membrane thickness
H	Vertical simulation box size
L	Lateral simulation box size
$\dot{\varepsilon}$	Deformation strain rate
$\dot{\gamma}_w$	Solvent shear rate
b	Membrane interleaflet friction
Δv	Relative velocity between leaflets

7.2.1 Viscosity of a Homogenous Newtonian Fluid

Viscosity characterizes the degree to which a fluid resists a steady shearing deformation. As discussed in section 3.2.3, the stress exerted between adjacent layers of a Newtonian fluid is proportional to the transverse velocity gradient. Thus, one can measure the viscosity with a non-equilibrium technique by either (1) applying a steady, modest shear stress and measuring the resultant velocity gradient (called the "shear rate"), or (2) imposing a steady, modest deformation (called "strain rate") and measuring the resultant shear stress.

Consider a fluid under constant, steady shear rate $\dot{\varepsilon}$ with a flow field given by

$$\vec{u}(\vec{r}) = \dot{\varepsilon} y \hat{x} \qquad (7.1)$$

$$\frac{\partial u_x}{\partial y} = \dot{\varepsilon} \qquad (7.2)$$

The Newtonian definition of viscosity (equation 3.17) relates this to the off-diagonal element of the shear stress tensor σ using the shear viscosity η

$$\sigma_{xy} = \eta\left(\frac{\partial u_x}{\partial y} + \frac{\partial u_y}{\partial x}\right) = \eta\dot{\varepsilon} \qquad (7.3)$$

This stress is equal to the average pressure tensor element P_{xy} after a conventional sign change (pressure is positive and stress is negative under compression)

$$\eta\dot{\varepsilon} = \sigma_{xy} = -\langle P_{xy}\rangle \qquad (7.4)$$

This pressure element may be interpreted as the average force per unit area exerted perpendicular to the shearing axis. We can calculate the viscosity by shearing the simulated fluid at a constant rate and measuring the mean shear pressure.

For simplicity, our shear deformation is applied by changing the shape of the entire simulation box at a constant rate. A general triclinic box may be defined by the lattice vectors $\vec{a}$, $\vec{b}$, and $\vec{c}$ corresponding to its edges. We can achieve a steady, volume-preserving shear in the xy plane by gradually increasing the x coordinate of the box vector $\vec{b}$ as a function of time and transforming the atomic coordinates in the box accordingly, as shown in **Fig. 7.1**. To prevent the box from becoming unduly deformed, we may subtract twice the box width from b_x when the angle between $\vec{b}$ and the y-axis reaches 45°. Under periodic boundary conditions, this results in an equivalent box which we can continue shearing along x. Since this procedure does work on the system, temperature coupling is required to dissipate the added energy. Throughout this chapter, the Bussi [60] thermostat is used for Martini simulations while Nosé-Hoover [58], [59] is used for CHARMM36.

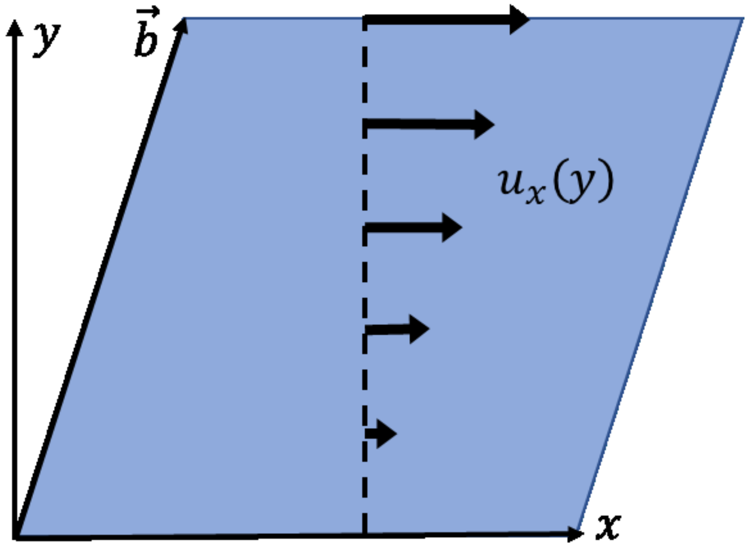

Figure 7.1: The average velocity field resulting from the deformation of a periodic box with lattice vector $b_x = tL_y\dot{\varepsilon}$.

This deformation-based shearing approach differs from the method employed by den Otter and Shkulipa. They imposed a shear flow using modified boundary conditions along the top and bottom of the box called Lees-Edwards boundary conditions [205] (see **Fig. 7.2**). These special boundary conditions slowly translate the upper and lower periodic images at constant velocity $\pm\dot{\varepsilon}L$ relative to the simulation box. This relative velocity is added to particles crossing the periodic boundary, generating a transverse velocity gradient which is linear at steady state. A shear flow may also be imposed by modifying the Hamiltonian to explicitly include the prescribed velocity field [74], [206], [207]. This third approach was attempted, but ultimately discarded in favor of the simpler box deformation method.

Fortunately, the analysis is unaffected so long as a linear transverse velocity gradient is achieved. At sufficiently low strain rates, the fluid is expected to be Newtonian and we expect the viscosity to be constant. At higher strains rates, shear thinning effects appear, decreasing the viscosity. This becomes evident for the membrane simulations but not the solvent simulations.

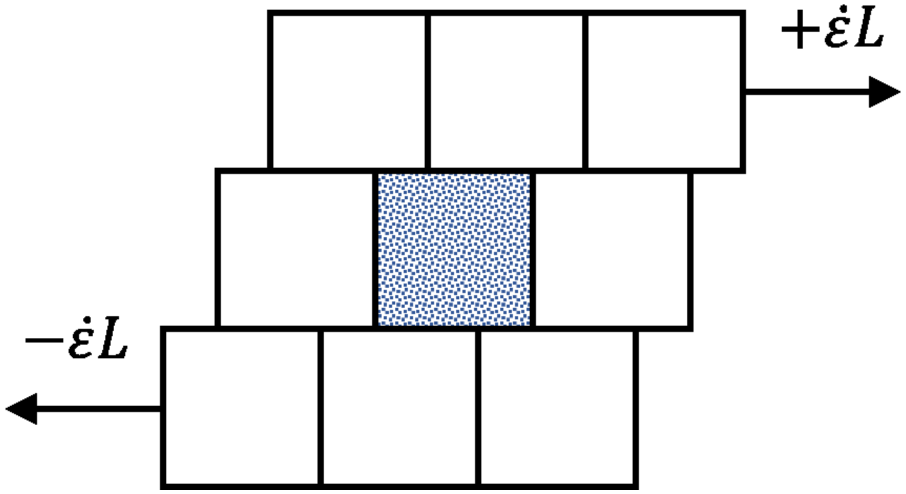

Figure 7.2: Lees-Edwards boundary conditions exert a shearing force on fluid in the simulation box by steadily moving its periodic images at a rate $\pm\dot{\varepsilon}L$ and adding the difference in velocity to all particles crossing the boundary.

7.2.2 Surface Shear Viscosity

Membrane surface viscosity is an effective 2D viscosity resisting lateral shear. We can use momentum conservation and dimensional analysis to understand to understand the differences between 2D surface viscosity and 3D shear viscosity. The total change in momentum within a region must equal the stress through its boundary. Since change in momentum has units of force, this means that the stress tensor must have units of force per area in 3D and force per length in 2D. It follows that surface viscosity must have units of viscosity × length. Keeping these units in mind, we can write an expression for surface viscosity

$$\sigma_{xy} = \eta_m \dot{\varepsilon} \tag{7.5}$$

$$\eta_m = h\eta_m^b \tag{7.6}$$

Where $\sigma_{\alpha\beta}$ is the viscous stress tensor and η_m is the surface viscosity of a membrane with thickness h and an effective 3D viscosity η_m^b.

Once more, we can calculate this from simulation by imposing a lateral shear (see **Fig. 7.1**) and recording the average value of the pressure tensor. The membrane stress is equal to the total stress minus the contribution from the solvent

$$\eta_m = \frac{\sigma_{xy}^{TOT} - \sigma_{xy}^{SOL}}{\dot{\varepsilon}} = \frac{\langle P_{xy}\rangle H - \eta_w \dot{\varepsilon}(H-h)}{\dot{\varepsilon}} \tag{7.7}$$

Where H is the height of the box and η_w is the viscosity of the solvent. Just as we did for the homogenous fluid, we must ensure our surface viscosity results are taken from systems near equilibrium. Our first safeguard is to vary the strain rates and look for a regime where the surface viscosity does not depend on the strain rate, i.e. the Newtonian regime. We can also check for correspondence with equilibrium simulations by calculating the total viscosity of the entire system and comparing it with the first term on the RHS of equation 7.7. This is possible with either a Green-Kubo relation [208]

$$\eta_{TOT} = \frac{V}{k_B T}\int_0^\infty \left\langle P_{xy}(t_0)\, P_{xy}(t_0+\tau)\right\rangle_{t_0} d\tau \tag{7.8}$$

or an equivalent Einstein relation

$$\eta_{TOT} = \frac{V}{2k_B T}\lim_{\tau\to\infty}\frac{d}{d\tau}\left\langle\left(\int_{t_0}^{t_0+\tau} P_{xy}(t)\, dt\right)^2\right\rangle_{t_0} \tag{7.9}$$

Equation 7.8 converges slowly due to pressure fluctuations which occur in MD simulations, and which are large relative to the average pressure due to the small sizes of typical MD simulation boxes. So long as enough samples are available, a plot of this equation as the upper limit of integration is increased will rise and reach a plateau.

At long times, fewer samples are available and this calculation becomes unreliable due to statistical error. This restriction is compounded by the fact the η_{TOT} calculation considers only one element of the pressure tensor, while calculations for homogenous fluids may average over all off-diagonal $P_{\alpha\beta}$. Better results are usually available from equation 7.9 by performing a linear fit to the time-averaged value of the integral vs the lag time τ.

7.2.3 Interleaflet Friction

Membrane interleaflet friction may also be calculated from non-equilibrium simulations. Recall the definition of interleaflet friction b from equation 3.52

$$F = b\Delta v \tag{7.10}$$

Where F is the friction force as the leaflets slide past one another and Δv is their relative velocity. We may calculate b from simulation by applying steady traction forces to each leaflet and recording the resulting mean relative velocity. In principle, this can be achieved through any traction force acting on the leaflets. Assuming stick boundary conditions, a parallel shear flow in the solvent with shear rate $\dot{\gamma}_w$ will exert a traction force on each leaflet given by

$$F = \eta_w \dot{\gamma}_w \tag{7.11}$$

Adding the traction forces on each leaflet, we may calculate the interleaflet friction

$$\eta_w\left(\dot{\gamma}_{upper} + \dot{\gamma}_{lower}\right) = b\Delta v \tag{7.12}$$

$$b = \eta_w\left(\frac{\dot{\gamma}_{upper} + \dot{\gamma}_{lower}}{\Delta v}\right) \tag{7.13}$$

The solvent shear rate must be calculated from the simulation trajectory by fitting to an average velocity profile along the shearing directions. Relative velocity is easily obtained by fitting the difference between leaflets as a function of time, where the leaflet position for each frame is given by its center of mass along the direction of the traction forces.

Shearing the solvent through box deformation imposes a differential force throughout each leaflet, complicating the task of computing the force exerted at the interface. A different method is preferred. As a simpler alternative to Lees-Edwards boundary conditions, **Fig. 7.3** depicts a method whereby a constant force is applied to all particles near the vertical boundaries. This method creates a nonlinear velocity profile in the vicinity of the forcing which rapidly transitions to a linear velocity gradient toward the center of the box. A linear fit to this part of the velocity profile gives the solvent shear rate $\dot{\gamma}_w$.

Obtaining a well-resolved velocity profile requires both spatial and temporal averaging. For each frame of the trajectory, a series of evenly-spaced points are selected along the box dimension normal to the membrane surface, which we can take to be the z axis without loss of generality. The mean velocity for each point may be calculated through a Gaussian-weighted spatial average of all nearby atoms (positions z_i) within a certain cutoff. The weights for the jth point located at $z = p_j$ are given by

$$w_{ij} = \exp\left[-\frac{1}{2}\left(\frac{z_i - p_j}{\sigma}\right)^2\right] \qquad (7.14)$$

with standard deviation σ controlling the degree of smoothing. The jth point of the velocity profile has velocity u_j given by

$$u_j = {\sum_i w_{ij}\, v_i}\Big/{\sum_i w_{ij}} \qquad (7.15)$$

Where v_i are the atom velocities. These profiles are averaged over all frames to obtain the final transverse velocity profile $\langle u_x(z)\rangle$ and the solvent shear rate $\dot{\gamma}_w$. Fits to the profile are performed above and below the membrane separately.

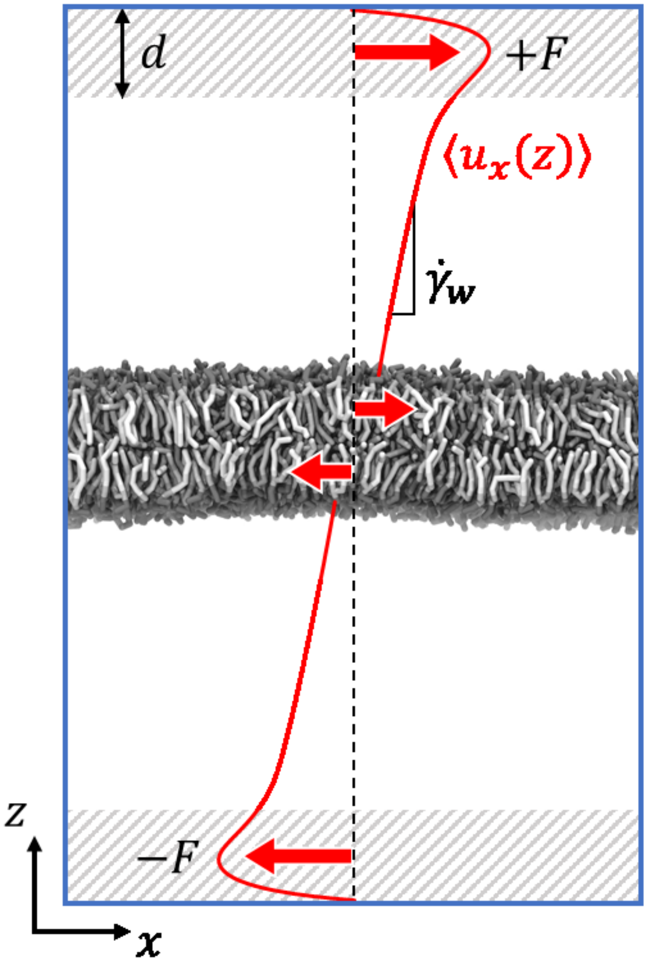

Figure 7.3: Diagram of the Martini normal shear protocol. A uniform force $\pm F$ is applied to all particles in the shaded regions of thickness d near the vertical periodic boundaries. Not far from these regions, the x-velocity profile becomes a linear function of z with shear rate $\dot{\gamma}_w$. The resulting traction forces on the leaflets are depicted with smaller red arrows.

7.3 Results: Solvent Viscosity

7.3.1 Martini Water

Before running membrane shearing simulations, the box-deformation protocol was tested on a 10 nm water cube containing c.a. 6100 martini water beads. This homogenous cube of water particles was equilibrated to 1 bar pressure in the NPT ensemble using the parameters from **Table 7.2** and sheared in the xy plane at different shear rates using the parameters from **Table 7.3**. The viscosity, calculated using the time average of the pressure tensor element P_{xy} and equation 7.4, was compared to published values of calculated with equilibrium methods. This protocol was carried out over a range of strain rates, so that a Newtonian shear regime could be identified where the viscosity does not depend on strain rate.

Table 7.2: Selected simulation parameters for equilibration of Martini water. Other parameters use defaults listed in **Table 2.1**.

Integration	Velocity-Verlet with $\Delta t = 20$ fs Center of mass motion removed every timestep
Thermostat	Bussi velocity rescaling; $T_0 = 323$ K, $\tau_T = 1$ ps
Barostat	Berendsen, semiisotropic coupling; $P_0 = 1$ bar, $\tau_P = 12$ ps compressibility $= 3.0 \times 10^{-4}$ bar^{-1}

This analysis is complicated by the fact that time series of P_{xy} feature large fluctuations (see **Fig. 7.4** for an example featuring a Martini membrane). As a result, obtaining a precise average requires significant simulation time, even for a system at steady-state. Several 400 ns shearing simulations were carried out to characterize this uncertainty and inform the necessary length of our production simulations. As shown in **Fig. 7.5**, diminishing returns are expected for simulations after 200 ns, where statistical uncertainty prohibits calculation of viscosity involving off-diagonal stress less than 0.1 bar. In our analysis, this uncertainty is further mitigated by running an ensemble of independent shearing simulations starting from identical initial states but with randomly assigned initial velocities and different seeds for the thermostat's random number generator.

Production simulations used three independent replicas at each strain rate for 420 ns with the first 20 ns discarded to account for the transient response to the deformation starting. They were run in the NVT ensemble, with the Bussi velocity rescaling thermostat at 323 K and no pressure coupling (fixed volume). **Fig. 7.6** depicts the resulting viscosities as the strain rate is varied. Viscosity is uniform over this range, with an inverse-variance weighted average of 0.6920 ± 0.0082, in close agreement with the 323 K value reported in the literature (0.69) [203] and our own independent calculations using equilibrium methods.

Table 7.3: Selected simulation parameters for shearing simulations of Martini water. Other parameters use defaults listed in **Table 2.1**.

Integration	Velocity-Verlet with $\Delta t = 20$ fs Center of mass motion removed every timestep
Thermostat	Bussi velocity rescaling; $T_0 = 323$ K, $\tau_T = 1$ ps
Barostat	None (fixed volume)

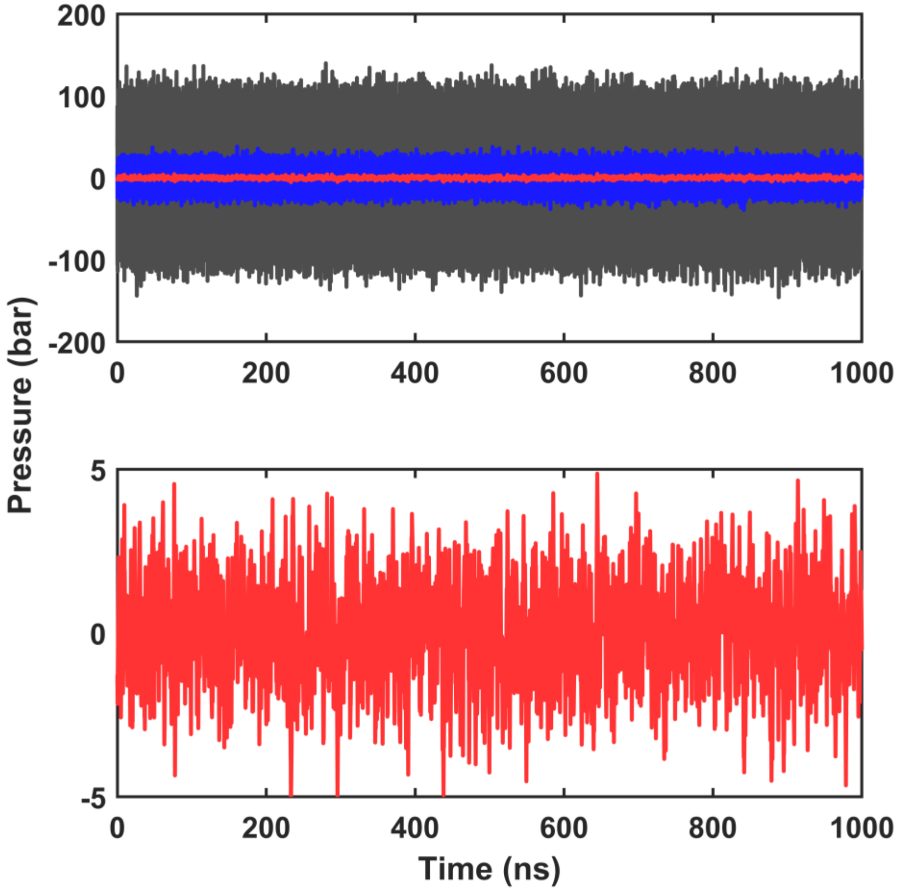

Figure 7.4: Example off-diagonal pressure for a relaxed Martini DPPC membrane sampled every 100 fs for 1000 ns. The upper panel depicts the raw time series in gray along with moving averages with windows of 10 ps (blue) and 1 ns (red). A zoomed in view of the 1 ns moving average is shown in the lower panel. The autocorrelation of this time series decays by 90% in under 1 ps.

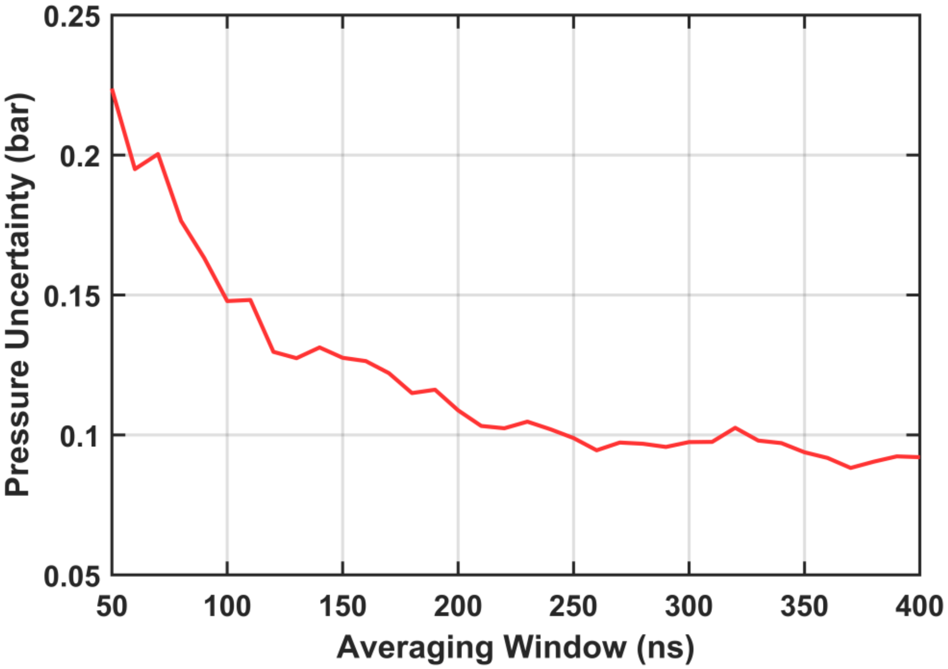

Figure 7.5: Uncertainty in average off-diagonal pressure P_{xy} for Martini coarse-grain water as the simulation time is increased.

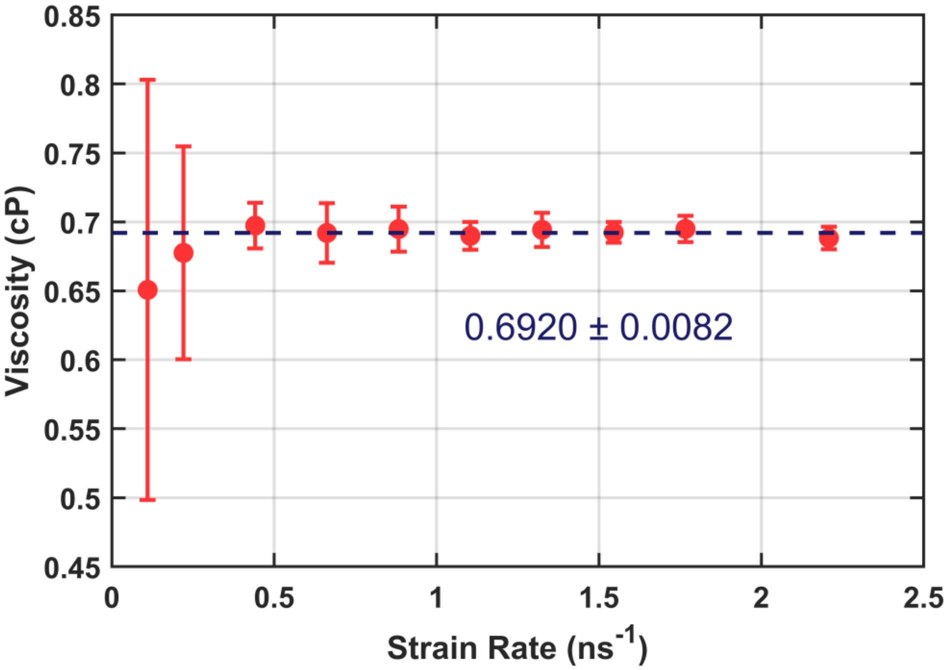

Figure 7.6: Martini water viscosity as a function of strain rate. Each data point is the average of three 400 ns trials and the dashed line indicates an inverse-variance weighted average of the data. Error bars combine the standard error computed from each set of three trials with error propagation.

7.3.2 TIP3P Water

The CHARMM36 lipid force field was parameterized using a modified version of the TIP3P water model. The original TIP3P [91] represents each molecule with a single charged van der Waals interaction site for the oxygen atom and two charged interaction sites for the hydrogens with no van der Waals interactions. The TIP3P hydrogens interact with other molecules solely through electrostatics. The modified version [209] used by CHARMM36 adds weak van der Waals interactions to the hydrogens. Both versions adequately model thermodynamic properties such as density and heat of vaporization, but suffer inaccurate transport coefficients, e.g. self-diffusion is too fast [210] and viscosity is too low [204].

Running shearing simulations of TIP3P water serves three purposes: (1) to evaluate the efficacy of our shearing protocol for calculating viscosity in the context of an all-atom force field, (2) to validate the results of that calculation using published values calculated from equilibrium simulations, and (3) to identify the relevant time scales and uncertainty in advance of the more extensive series of simulations involving all-atom membranes. The TIP3P shear protocol closely resembles the protocol used for Martini water in the previous section. 5 nm cubic systems containing 4074 TIP3P molecules were equilibrated to 1 bar pressure at various temperatures ranging from 293 to 323 K using the parameters in **Table 7.4**. Two systems were prepared at each temperature using different treatments of the cutoff for van der Waals interactions, one switching the force to zero over 8-10 Å and the other over 8-12 Å. Both are required in order to test the sensitivity of the solvent viscosity to the cutoff treatment and in anticipation of an issue described in a later section on the surface viscosity CHARMM36 DPPC membranes.

Table 7.4: Selected simulation parameters for equilibration of CHARMM36 water. Other parameters use defaults listed in **Table 2.3**.

Integration	Velocity-Verlet with $\Delta t = 2$ fs Center of mass motion removed every 100 timesteps
Van der Waals	Force switched to zero over 8-10 Å or 8-12 Å
Thermostat	Nosé-Hoover; Various T_0, $\tau_T = 1$ ps

Barostat	Berendsen, isotropic coupling;
	$P_0 = 1$ bar, $\tau_P = 5$ ps
	compressibility = 3.0×10^{-5} bar^{-1}

Each production series was run over a range of shear rates from 0.2 to 4.0 ns^{-1} with five independent replicas used for each datum. Each individual simulation ran at fixed volume with a shear deformation imposed in the xy plane using the parameters in **Table 7.5**. Based on the pressure uncertainty vs time curve depicted in **Fig 7.7**, the length of each production simulation was chosen to be 12 ns with the first 2 ns discarded. Results for a particular simulation series (323 K with 8-10 Å force switching) are presented in **Fig. 7.8**. In this figure, viscosity is uniform over the whole range of strain rates. The inverse-variance weighted average is 0.242 ± 0.013. The full data set is presented in **Fig. 7.9**, showing our data compared with published results from equilibrium viscosity calculations [204]. We find excellent agreement over the whole temperature range for both cutoff treatments. Viscosity is slightly lower when using 8-12 Å cutoffs.

Table 7.5: Selected simulation parameters for shearing simulations of CHARMM36 water. Other parameters use defaults listed in **Table 2.3**.

Integration	Velocity-Verlet with $\Delta t = 2$ fs Center of mass motion removed every 100 timesteps
Van der Waals	Force switched to zero over 8-10 Å or 8-12 Å
Thermostat	Bussi velocity rescaling; Various T_0, $\tau_T = 1$ ps
Barostat	None (fixed volume)

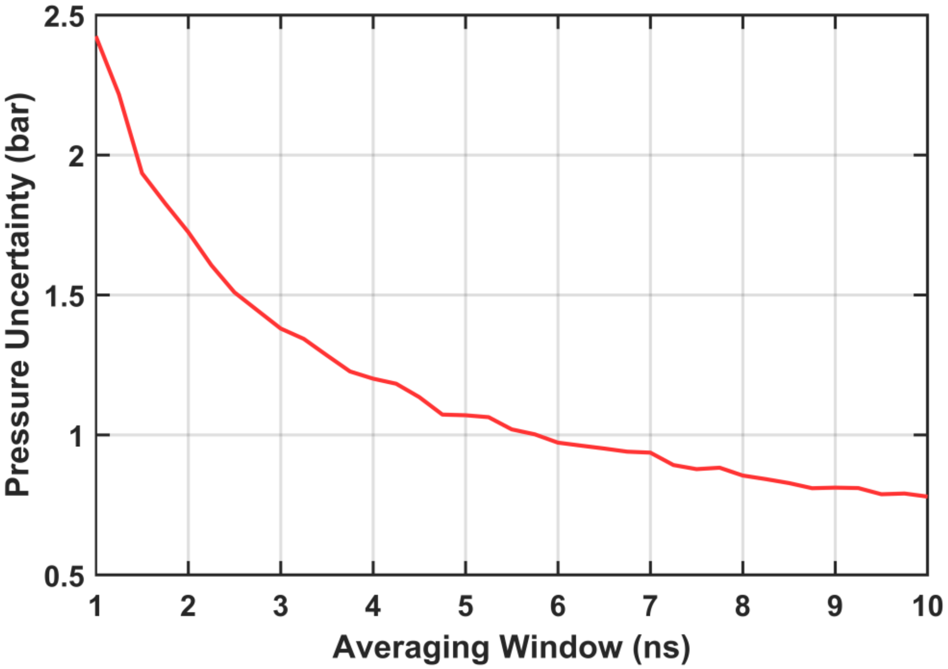

Figure 7.7: Uncertainty in average off-diagonal pressure P_{xy} for TIP3P as the averaging window is adjusted.

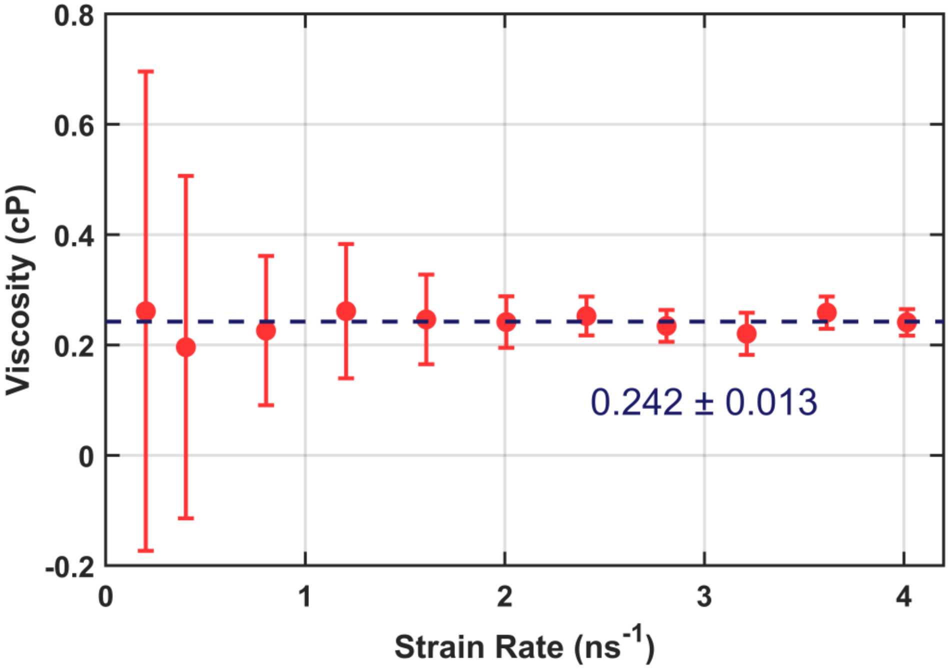

Figure 7.8: Example TIP3P viscosity as a function of shear rate using 8-10 Å force switching at 323 K. Each data point is the average of five 10 ns trials. The dashed line indicates the inverse-variance weighted average. Error bars combine the standard error computed from each set of five trials with error propagation.

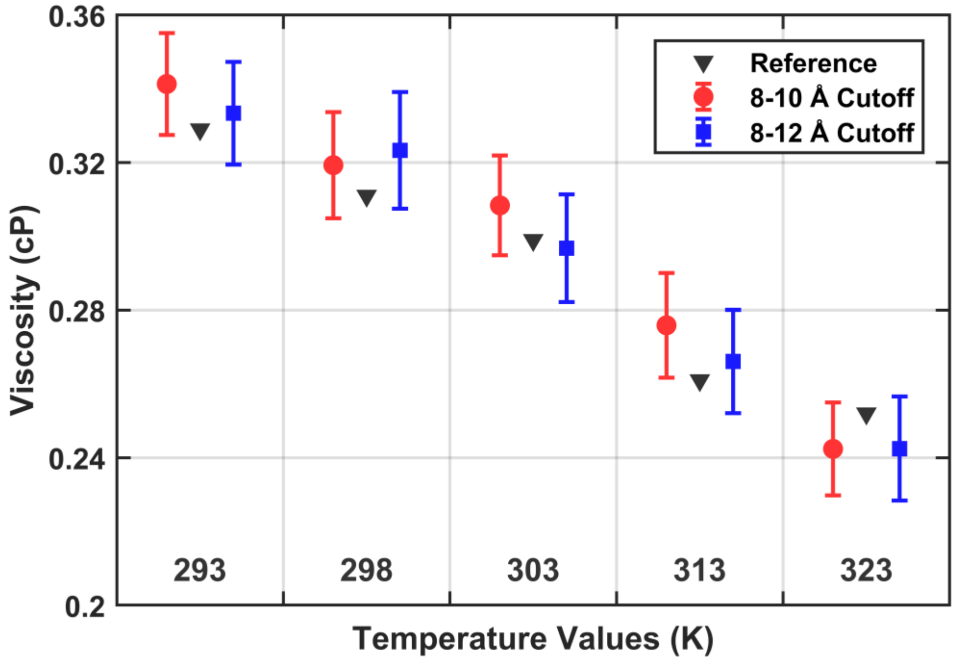

Figure 7.9: CHARMM36 TIP3P viscosity determined through shearing simulations compared to reference equilibrium calculations. Error bars represent the uncertainty in the weighted fit at each temperature.

7.4 Surface Shear Viscosity

7.4.1 Martini DPPC

Parameters for the Martini forcefield have changed substantially since the initial publication [211] cited by den Otter and Shkulipa and used in their Martini shearing simulations [192]. Though the coarse-grain structure of DPPC lipids has not changed, newer versions of Martini [54], [109], [110] have revised the interactions between its constituent CG beads. Interactions have generally become more attractive both among headgroup beads and between the headgroup beads and water, while interactions between headgroup and tail beads have become more repulsive (see **Table 7.6**). Since the overall attraction between lipids has increased, we should expect greater surface shear viscosities from simulations using newer Martini parameters relative to the findings of den Otter and Shkulipa. We calculate surface viscosity for DPPC using two versions of Martini: the current version, Martini 2.2, and the original version used by den Otter and Shkulipa (henceforth “Martini 2004”).

Table 7.6: Comparison of Lennard-Jones parameters for the DPPC bead interactions in Martini 2004 and Martini 2.2. Solvent beads are highlighted in blue, headgroups in purple, and alkane tail beads in tan.

Interaction		σ (2004)	ε (2004)	σ (2.2)	ε (2.2)	Change
W	W	0.47	5.0	0.47	5.0	=
W	GL	0.47	3.4	0.47	4.0	More attractive (ε+0.6)
W	C	0.47	1.8	0.47	2.0	≈
W	PO4	0.47	5.0	0.47	5.6	More attractive (ε+0.6)
W	NC3	0.47	5.0	0.47	5.6	More attractive (ε+0.6)
GL	GL	0.47	4.2	0.47	4.0	≈
GL	C	0.47	2.6	0.47	2.7	≈
GL	PO4	0.47	3.4	0.47	4.0	More attractive (ε+0.6)
GL	NC3	0.47	3.4	0.47	4.0	More attractive (ε+0.6)
C	C	0.47	3.4	0.47	3.5	≈
C	PO4	0.47	1.8	0.62	2.0	More repulsive (σ+0.15)
C	NC3	0.47	1.8	0.62	2.0	More repulsive (σ+0.15)
PO4	PO4	0.47	3.4	0.47	5.0	More attractive (ε+0.6)
PO4	NC3	0.47	3.4	0.47	4.5	More attractive (ε+1.1)
NC3	NC3	0.47	3.4	0.47	3.5	≈

For both versions of Martini, 10 nm membranes consisting of 169 DPPC lipids per leaflet were created and solvated with a 4 nm layer of Martini water. To reach a state of zero surface tension, ten replicas with different initial velocities were created for both parameter sets and equilibrated over 1 μs using the parameters in **Table 7.7**. The final systems were rescaled to equal the inverse-variance weighted average of the box sizes from these equilibration simulations (discarding the first 100 ns of each) and run for a further 100 ns in the NVT ensemble to relax disturbances from the velocity rescaling.

Production simulations were 820 ns long with the first 20 ns discarded. They were run at constant volume using the parameters from **Table 7.8** with a lateral shear applied in the xy plane. Since the uncertainty in P_{xy} from a single simulation plateaus around 250 ns (see **Fig. 7.10**), further reductions were obtained by running three replicas for each strain rate with different initial conditions and random seeds. A wide range of strain rates were explored. Results from the Newtonian shear regime shown in **Fig. 7.11**. As expected, surface viscosity in Martini 2.2 is higher than Martini 2004, likely a consequence of the general increase in attraction between lipid beads. The result for Martini 2004 ($= 1.457 \pm 0.043$) compares favorably with the den Otter and Shkulipa result [192] ($= 1.2$) considering the implementation differences between MD programs.

Results for Martini 2.2 over a wider range of shear rates are depicted on a log scale in **Fig. 7.12**, where both the Newtonian plateau and the shear thinning regime are evident. The danger of simply extrapolating results from the "linear" shear thinning regime is shown in **Fig. 7.13**. Without knowing how close one is to the plateau, an extrapolation can significantly over- or underestimate the equilibrium surface viscosity. It is therefore prudent to locate and consider only those strain rates which are slow enough to remain in the Newtonian regime.

Table 7.7: Selected simulation parameters for equilibration of Martini DPPC. Other parameters use defaults listed in **Table 2.1**.

Integration	Velocity-Verlet with $\Delta t = 20$ fs Center of mass motion removed every timestep, treating membrane and solvent separately.
Thermostat	Bussi velocity rescaling, treating water and lipids separately; $T_0 = 323$ K, $\tau_T = 1$ ps
Barostat	Berendsen, semiisotropic coupling; $P_0 = 1$ bar, $\tau_P = 12$ ps compressibility $= 3.0 \times 10^{-4}$ bar^{-1}

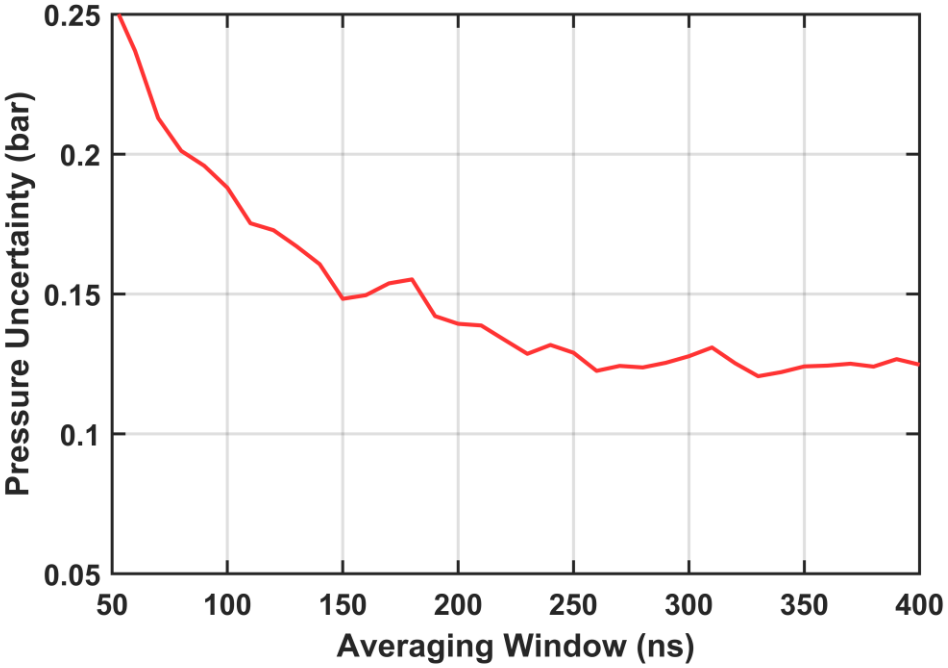

Figure 7.10: Uncertainty in average off-diagonal pressure P_{xy} for a Martini DPPC membrane as the simulation time is increased.

Table 7.8: Selected simulation parameters for shearing simulations of Martini DPPC. Other parameters use defaults listed in **Table 2.1**.

Integration	Velocity-Verlet with $\Delta t = 20$ fs Center of mass motion removed every timestep, treating membrane and solvent separately.
Thermostat	Bussi velocity rescaling, treating water and lipids separately; $T_0 = 323$ K, $\tau_T = 1$ ps
Barostat	None (fixed volume)

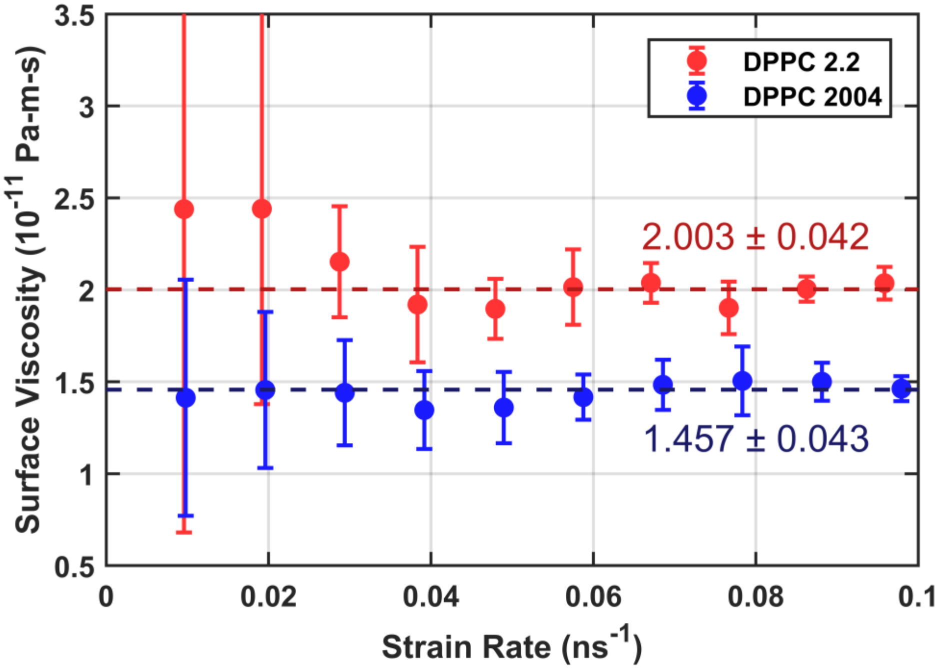

Figure 7.11: Surface viscosity as a function of shear rate for DPPC membranes using force field parameters for the 2004 version of Martini cited by den Otter and Shkulipa and the more recent Martini v2.2. Dotted lines show the result of taking inverse-variance weighted averages for both. Error bars combine the standard error computed from each set of three trials with error propagation.

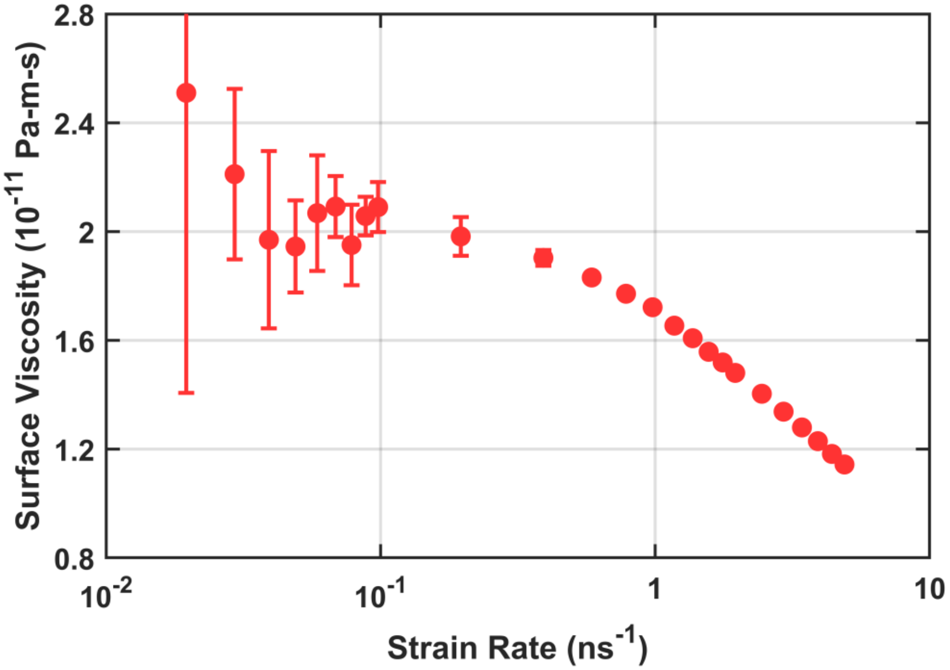

Figure 7.12: Extended range of strain rates demonstrating turnover from Newtonian viscosity to shear thinning at higher shear rates for Martini 2.2 DPPC (slowest strain rates omitted for clarity due to large uncertainty bars).

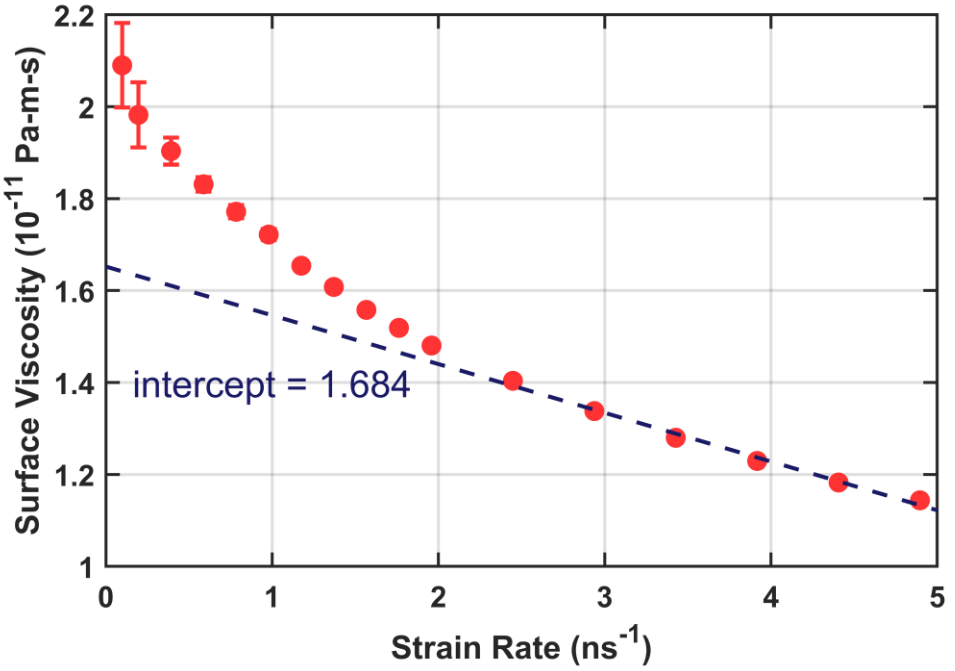

Figure 7.13: Surface viscosity of Martini DPPC showing only the higher strain rates (those covered in **Fig. 7.12** but not **Fig. 7.11**). An extrapolation from this linear regime (dotted line) significantly underestimates the equilibrium surface viscosity calculated from the low strain regime.

For additional validation, the total system viscosity (membrane + solvent) was calculated from a 1 μs simulation at equilibrium using both the Green-Kubo relation from Equation 7.8 (**Fig. 7.14**) and its equivalent Einstein relation from Equation 7.9 (**Fig. 7.15**). Extracting the total viscosity from the Green-Kubo relationship is difficult as it requires resolving and identifying the "infinite time" plateau, which itself requires precisely resolving long-time correlations. In theory, this plateau extends unbroken to infinite lag times. However, the lack of sampling at long times and the accumulated noise between distant times in the simulation cause the long-time correlations to randomly drift. This leads to a subjective choice over where the plateau is located: where it begins and where it ends. This is avoided when using the Einstein relation, which is calculated using much short correlation times. Note that Equation 7.9 takes the limit of long correlation times. This limit is relative to the autocorrelation time of the pressure time series, which decays by 90% in under 1 ps. The linear fit in **Fig. 7.15** is taken over sufficiently long lag times for the Einstein relation to hold.

The results are encouraging, with close agreement between the weighted average of total viscosity from shearing simulations 2.264 ± 0.042 and the average from Einstein relations 2.23 ± 0.21. The uncertainty on the latter figure is obtained by breaking the equilibrium P_{xy} time series into four equal parts before performing the Einstein relation analysis on each. **Fig. 7.15** shows the result for one of these four.

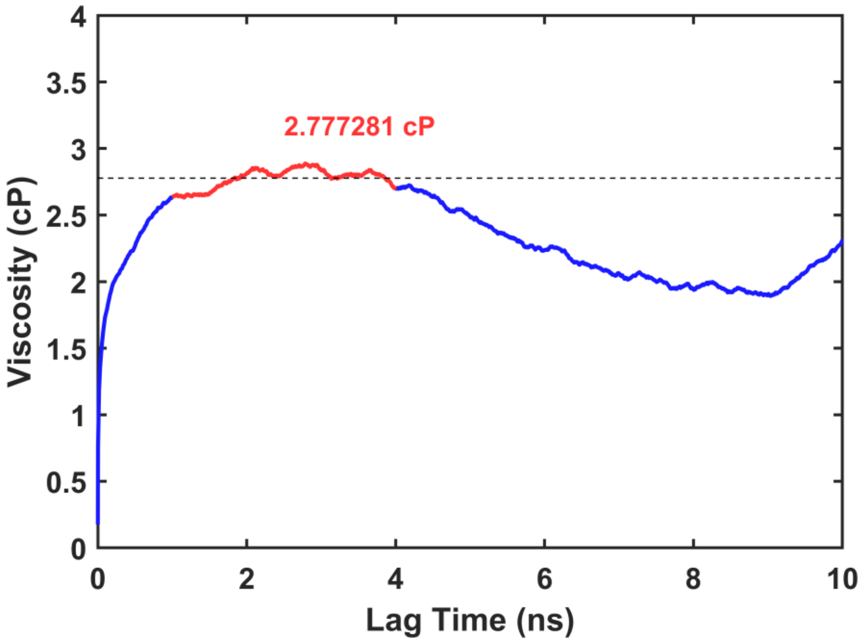

Figure 7.14: Sample depiction of total viscosity for Martini DPPC using the Green-Kubo relation (equation 7.8). The correlation builds until a plateau is reached. The plateau is not maintained as lag times increase due to the finite length of the simulation and the decrease in the number of samples between longer lag times.

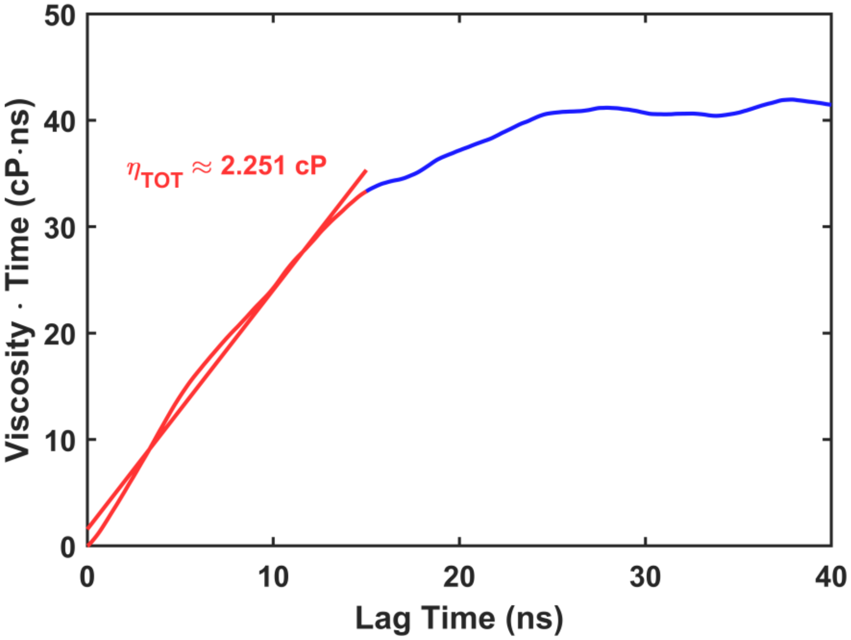

Figure 7.15: Sample depiction of total viscosity for Martini DPPC using the Green-Kubo equation recast as an Einstein relation (equation 7.9). The expression under the derivative is plotted as a function of time. A linear fit to the linear regime gives total viscosity. This occurs for times that are long compared to the pressure fluctuations but short compared to the length of the simulation.

7.4.2 CHARMM36 DPPC

DPPC was chosen as an initial test case for shearing simulation using the CHARMM36 force field due to its use in experiments and simulations as a standard case. In general, membranes modeled with all-atom force fields are expected to exhibit higher surface viscosities than their coarse-grain counterparts due to the additional conformational freedom of the all-atom fatty acid tails and stronger interactions among their constituent interaction sites. We began our investigation of DPPC by creating and equilibrating a 10 x 10 nm membrane patch with 160 lipids per leaflet solvated by a 4 nm layer of TIP3P water using the CHARMM-GUI membrane builder

[104]. Our initial equilibration simulations ran for 10 ns using the parameters listed in **Table 7.9**. During these simulations, the box size reached a steady lateral size by 2 ns.

Table 7.9: Selected simulation parameters for equilibration of CHARMM36 DPPC. Other parameters use defaults listed in **Table 2.3**.

Integration	Leapfrog with $\Delta t = 2$ fs Center of mass motion removed every 100 timesteps, treating membrane and solvent separately.
Van der Waals	Interaction forces switched off over 8-10 or 8-12 Å.
Thermostat	Nosé-Hoover, treating water and lipids separately; $T_0 = 323$ K, $\tau_T = 1$ ps

Barostat	Parrinello-Rahman, semiisotropic coupling;
	$P_0 = 1$ bar, $\tau_P = 5$ ps
	compressibility = 4.5×10^{-5} bar^{-1}

Despite using the standard set of CHARMM36 run parameters, our DPPC membranes relaxed to an area per lipid of c.a. 61 Å^2 rather than the expected experimental value of 63 Å^2 [212]. This is concerning as lipid packing density plays an influential role in observed diffusion [213] and hence surface viscosity. Fearing a bug, we reproduced this discrepancy in four ways: (1) with three different versions of GROMACS on the same cluster, (2) with the same version of GROMACS on two other clusters, (3) using CHARMM36 topology files from an alternative source, (4) and using a different MD program called NAMD with the same settings. The same anomalous area per lipid was found in every case.

Equilibrium area per lipid is quite sensitive to the treatment of nonbounded force calculations including the cutoff lengths [104] and the frequency of recalculating long-range interactions [214]. Whenever possible, the same simulation parameters and software should be used as were used to parameterize the force field. Discrepancies arise among different MD programs due to subtle differences in implementation (e.g., the exact form of the force switching). In this case, the “bug” we encountered arose from a discrepancy within the CHARMM force field [215]. The lipid force field was parameterized with nonbonded forces switching off over the range of 8-12 Å [103] while proteins were parameterized using the range 10-12 Å [51]. To ensure compatibility with membrane-associated proteins, the CHARMM-GUI membrane builder generates run files using the latter parameters. The expected area per lipid with these parameters is 60 - 61 Å^2 [104], [215], in agreement with our equilibration results. Strictly speaking, these parameters are incorrect for membrane simulations.

A study comparing DPPC area per lipid using different MD programs with various force switching ranges (8-10, 8-12, 10-12) found 8-10 Å to be the most accurate at 63.6 Å^2 [104]. Thus, we are left with three choices for our shearing simulations: (1) 8-10, which best agrees with experiment when using GROMACS, but was not used during force field parameterization; (2) 10-12, which was used for protein parameterization, and therefore is the most widely-used; and (3) 8-12, which was used for lipid parameterization and the simulations for the PSD fits from Venable et al. [46]. We decided to calculate surface viscosity using 8-10 and 8-12 switching for DPPC and only 8-12 switching for all other membranes.

Equilibrating the membrane to a truly relaxed state is necessary to obtain an accurate calculation of the equilibrium surface viscosity. Example results for a poorly-equilibrated membrane are depicted in **Fig. 7.16**. This membrane consists of a Lo mixture compressed 8% relative to its relaxed lateral size. Its nonzero surface tension results in a stress tensor that is not asymptotically zero as the strain rate is decreased.

This latent stress contribution becomes dominant as the shear contribution becomes smaller, which can be seen by solving equation 7.7 for the pressure and adding a constant term T_{xy}

$$\langle P_{xy} \rangle = \eta_m^b \dot{\varepsilon} \frac{h}{H} + \eta_w \dot{\varepsilon} \frac{(H-h)}{H} + T_{xy} \qquad (7.16)$$

If we apply the analysis developed for calculating η_m, assuming P_{xy} is proportional to the strain rate, this equation results in a divergence as $\dot{\varepsilon} \to 0$ which is clearly seen in **Fig. 7.16**. This latent stress also manifests at higher strain rates, where the surface viscosity is an order of magnitude higher than what is found in a relaxed membrane. The divergence at low shear rates allows us to identify poorly-equilibrated systems from their surface viscosity plots.

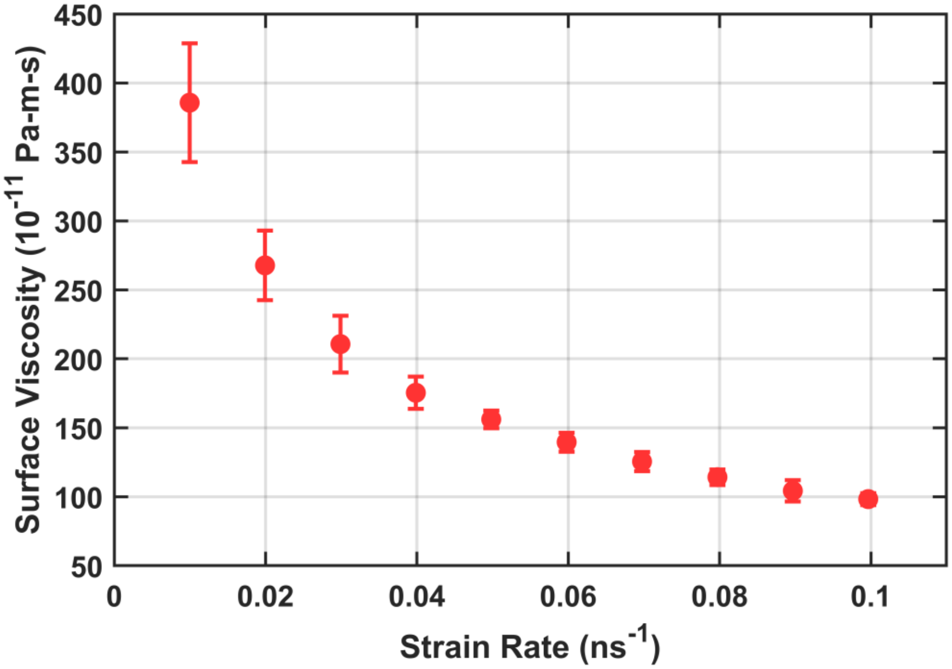

Figure 7.16: Surface viscosity as a function of strain rate for a poorly-equilibrated membrane (Lo Mixture with 8-12 Å force switching).

Since our shearing simulations run at constant volume, any initial stress cannot relax during the simulation. Careful equilibration is required to avoid this scenario. After creating and minimizing each membrane, nine replicas are created for each with random initial atom velocities and random seeds. The replicas are equilibrated in the NPT ensemble for 12 ns using the parameters from **Table 7.9**. The last 10 ns of each are used in **Fig. 7.17**, and the equilibrium area per lipid is computed from the inverse variance weighted average of this data. The membrane is then rescaled to this size and equilibrated at constant volume for an additional 10 ns to relax transient disturbances resulting from this rescaling.

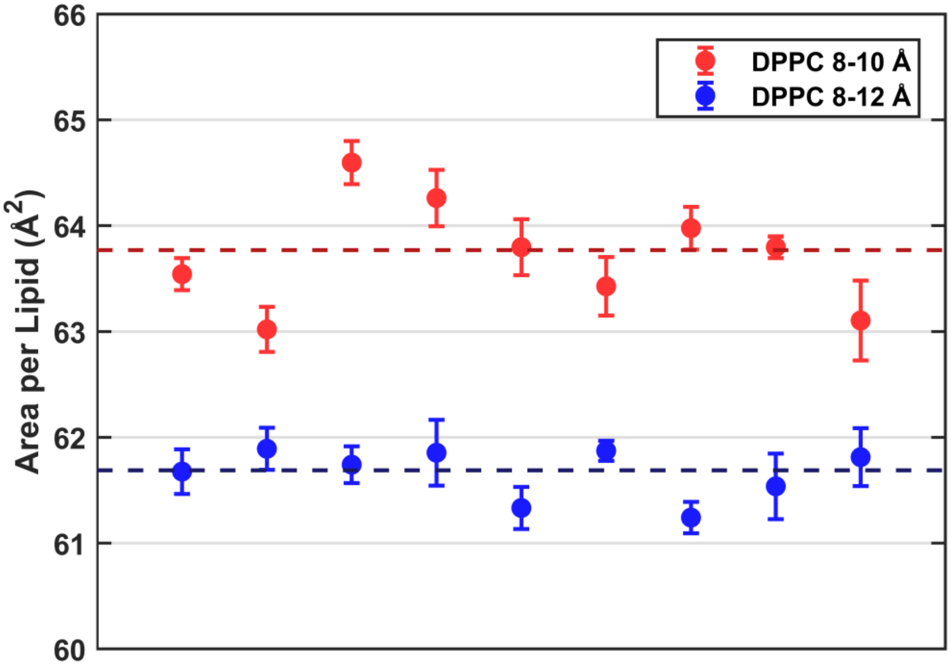

Figure 7.17: Sample equilibration results for CHARMM36 DPPC showing the average area per lipid of nine 10 ns trials for each cutoff treatment.

The additional computational cost associated with all-atom simulation requires us to use much shorter production simulations. Fortunately, we do not require the pressure uncertainty to be as low as coarse-grained membrane simulations due to the higher surface viscosity in CHARMM36. According to **Fig. 7.18**, 20 ns simulations are sufficient for an uncertainty < 1.5 bar with rapidly diminishing returns thereafter. Our computational resources are much better used on more replicas per strain rate. With this in mind, five 25 ns production simulations were run at each strain rate using the simulation parameters from **Table 7.10**. A wide range of shear rates was explored for DPPC with results at low shear rate presented in **Fig. 7.19** and all shear rates in **Fig. 7.20**. The error bars on these points combine the statistical uncertainty within each replicant with the variance of the replicants.

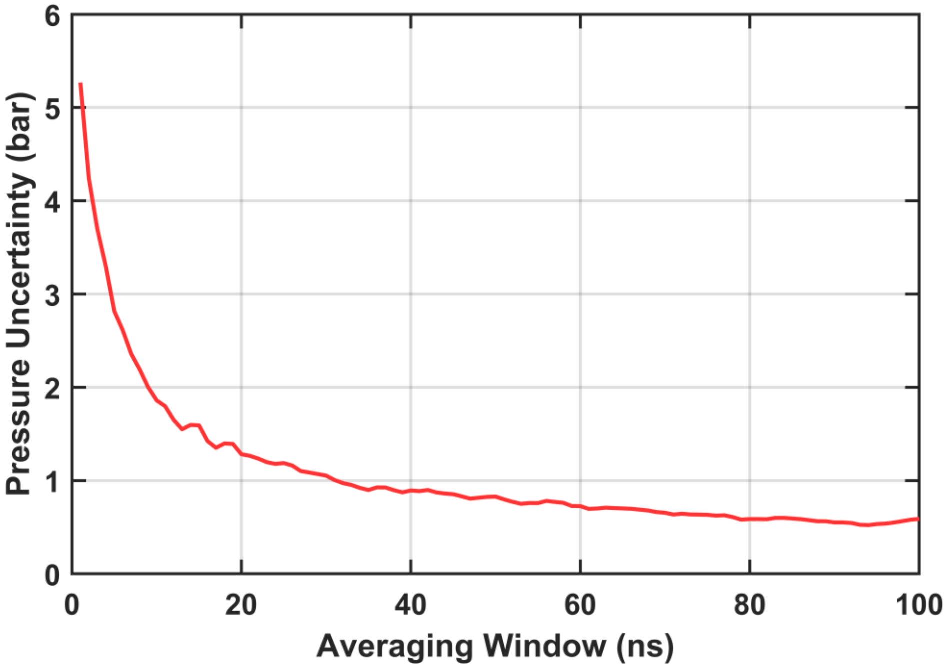

Figure 7.18: Uncertainty in average off-diagonal pressure P_{xy} for a CHARMM36 DPPC membrane as the simulation time is increased.

Table 7.10: Selected simulation parameters for shearing simulations of CHARMM36 DPPC. Other parameters use defaults listed in **Table 2.3**.

Integration	Leapfrog with $\Delta t = 2$ fs Center of mass motion removed every 100 timesteps, treating membrane and solvent separately.
Van der Waals	Interaction forces switched off over 8-10 or 8-12 Å.
Thermostat	Nosé-Hoover, treating water and lipids separately; $T_0 = 323$ K, $\tau_T = 1$ ps
Barostat	None (fixed volume)

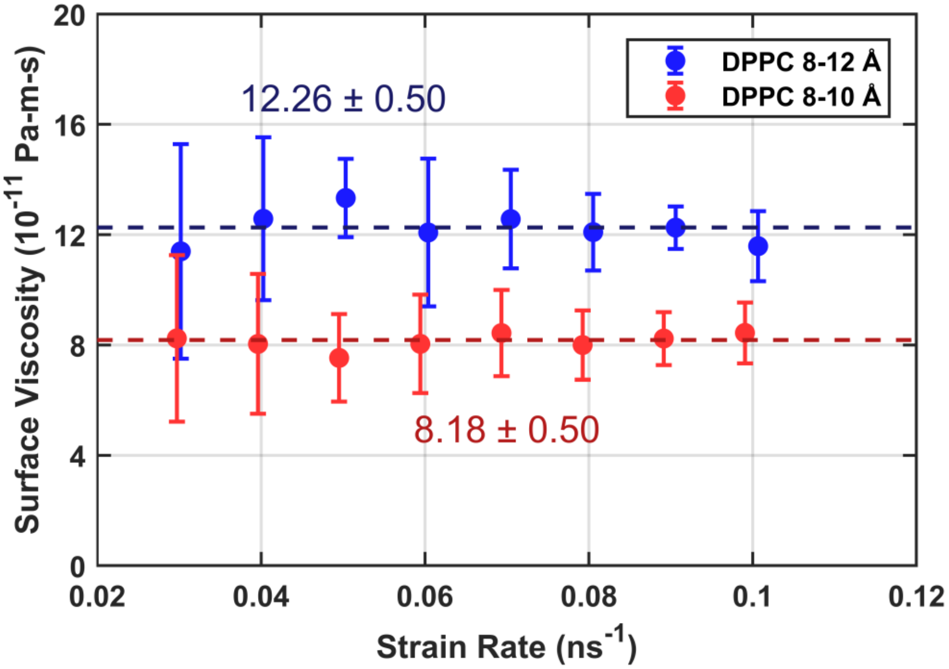

Figure 7.19: Surface viscosity as a function of shear rate for CHARMM36 DPPC using two cutoff treatments: standard 8-12 Å force switching and “best DPPC area per lipid” 8-10 Å force switching. Both series were equilibrated independently to zero surface tension before shearing. Dotted lines show the result of taking inverse-variance weighted averages for each series. Each data point is the average of five trials at the given strain rate.

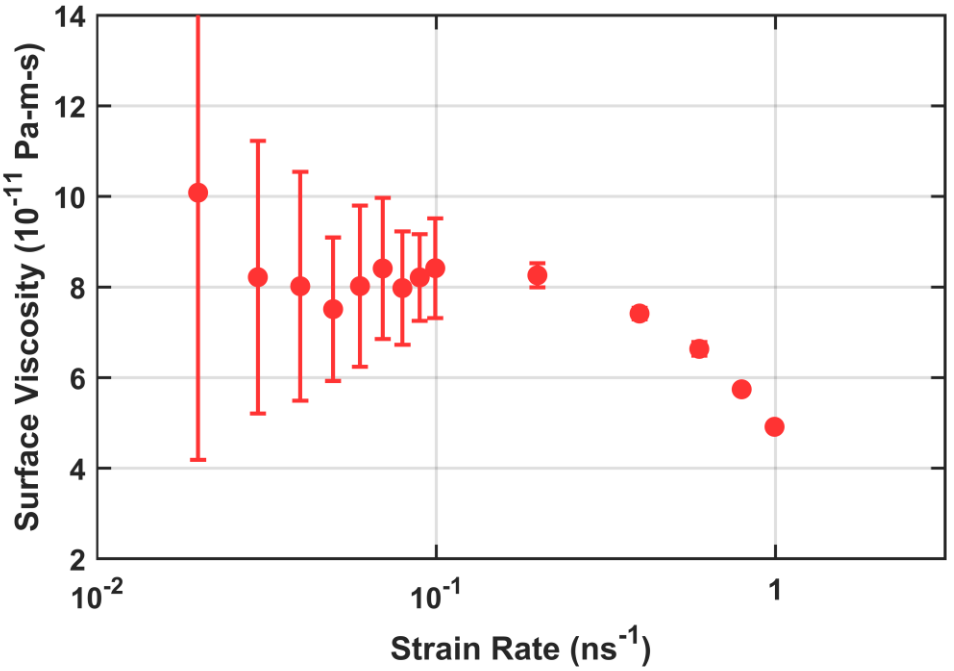

Figure 7.20: Extended range of strain rates demonstrating turnover from Newtonian viscosity to shear thinning at higher shear rates for CHARMM36 DPPC using 8-10 Å force switching (slowest strain rate omitted for clarity due to large uncertainty bars).

As expected, the cutoff treatment has a significant effect on the surface viscosity (~ 50%), even more so than its effect on equilibrium area per lipid. Like our Martini calculations, these surface viscosity results were independently corroborated with equilibrium calculations. Five 50 ns replicas with random initial atom velocities were simulated for both force switching ranges in the NPT ensemble using the simulation parameters from **Table 7.9**. Equilibrium surface viscosity was calculated using both the Green-Kubo relation from Equation 7.8 (**Fig. 7.14**) and its equivalent Einstein relation from Equation 7.9 (**Fig. 7.15**). Results for the average of each five trials are reported in **Table 7.11**, finding good agreement with values calculated from the shearing simulations, especially for 8-12 Å force switching.

Table 7.11: Total viscosity calculated for CHARMM36 DPPC membranes. Results from shearing simulations are presented along with average equilibrium calculations for both 8-10 and 8-12 Å force switching.

Method	Total Viscosity (10^{-11} Pa-m-s) = (10^{-8} P*cm)
Shearing (8-10)	8.27 ± 0.50
Einstein Relation (8-10)	11.1 ± 2.4
Shearing (8-12)	12.35 ± 0.50
Einstein Relation (8-12)	12.2 ± 1.4

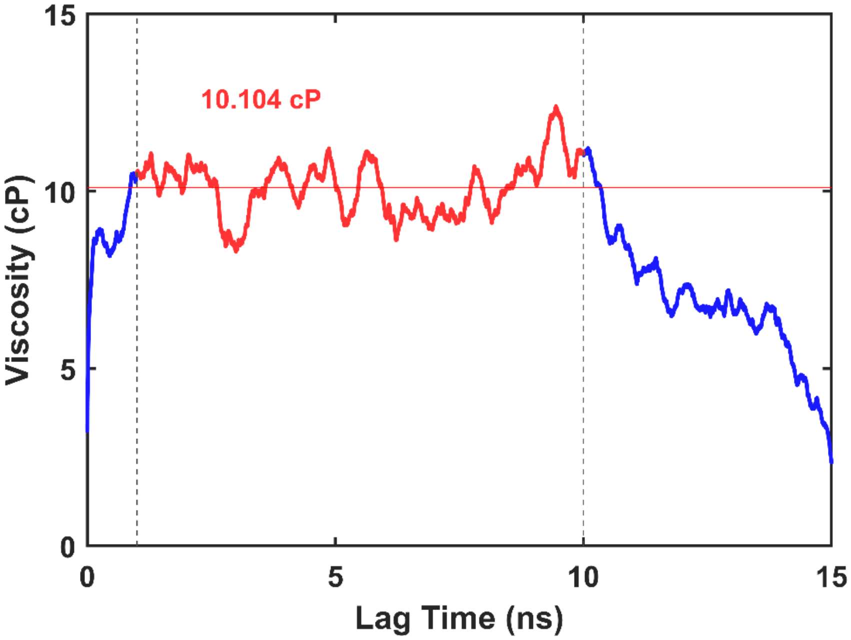

Figure 7.21: Sample depiction of total viscosity for CHARMM36 DPPC with 8-12 force switching using the Green-Kubo relation (equation 7.8). The correlation builds until a plateau is reached. The plateau is not maintained as lag times increase due to the finite length of the simulation and the decrease in the number of samples between longer lag times.

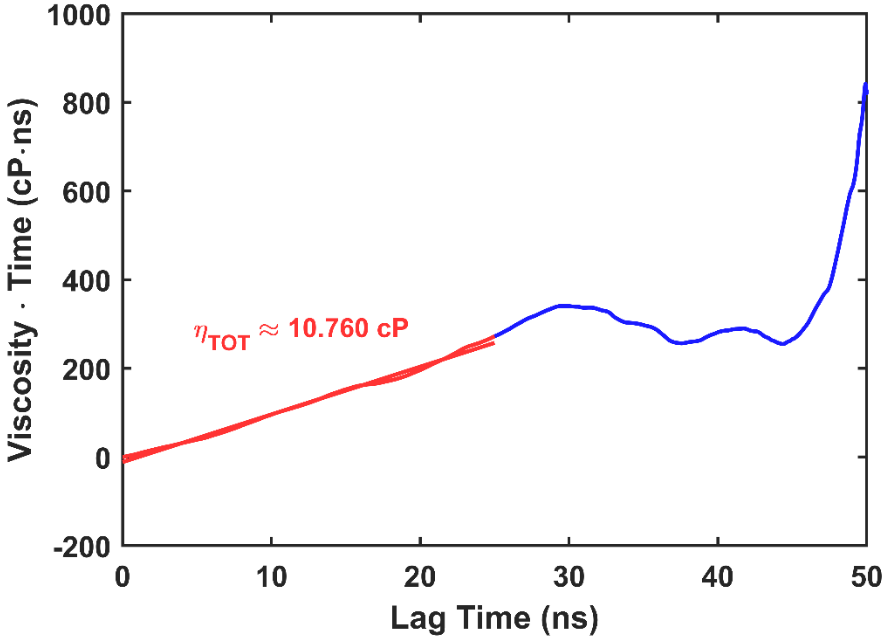

Figure 7.22: Sample depiction of total viscosity for CHARMM36 DPPC with 8-12 force switching using the Green-Kubo equation recast as an Einstein relation (equation 7.9). The expression under the derivative is plotted as a function of time. A linear fit to the linear regime gives total viscosity. This occurs for times that are long compared to the pressure fluctuations but short compared to the length of the simulation.

7.4.3 Other CHARMM36 Lipids

The same equilibration protocol, shearing simulations, and analysis used for DPPC were repeated for DOPC and PSM. The hydrocarbon tails of DOPC are two carbons longer than DPPC and contain an unsaturated bond in the middle of the chain, leading to more disordered tails and a larger area per lipid compared to DPPC. Even though it has a higher area per lipid, the surface viscosity of DOPC membranes are expected to exceed DPPC on the basis of diffusion experiments. This may be the result of enhanced entanglement among the oleyl tails due to kinks from the unsaturation. PSM has a sphingosine backbone rather than the glycerol in DPPC and DOPC, which forms extensive PSM-PSM hydrogen bonds at the polar-nonpolar interface[216]. PSM is therefore expected to have an even higher surface viscosity. Each membrane was simulated above their melting temperature (see **Table 7.12**).

Two mixtures were also tested, one in the liquid ordered (L_o) phase (55% DPPC, 15% DOPC, 30% cholesterol) and another in the liquid disordered (L_d) phase (30% DPPC, 60% DOPC, 10% cholesterol). Based on experimental measurements of diffusion, which is slower in L_o, the L_o phase is expected to have higher viscosity [198] [29]. This is likely due to the higher concentration of cholesterol in the L_o phase, which is known to slow diffusion by ordering the tails [217] and lowering the area per lipid. Both mixtures were run at 323 K.

Equilibration results are presented in **Table 7.12**. Minor differences from these reference values are expected due to implementation-specific details of the different MD programs (GROMACS vs CHARMM) and differences in timestep (2 vs 1 fs). Agreement with reference values are excellent for DPPC and DOPC, but PSM area per lipid is 3.3% too large. Its shear viscosity plot exhibits minor signs of latent stress, indicating incomplete equilibration. After an additional 100 ns equilibration, PSM area per lipid fell within 1% of the reference value. Curiously, the shear viscosity plot of this “well-equilibrated” PSM diverges significantly at low shear rates. Results reported for PSM use the original, “poorly-equilibrated” membrane. Although no reference sizes are available for the mixtures, we can infer adequate equilibration from the lack of divergence in their surface viscosity data (**Fig. 7.23**).

Surface viscosity results are reported in **Table 7.13**. As expected, DOPC has a higher surface viscosity than DPPC and PSM has the highest among the single-component membranes. The L_o mixture has a higher surface viscosity than the L_d (see

Fig. 7.23) mixture despite the higher abundance of DPPC which, given the single-component results, one may naïvely expect to lower viscosity. This instead reflects the tail ordering imposed by the additional cholesterol in the L_o mixture.

Table 7.12: Equilibration summary for CHARMM36 membranes using different cutoff treatments compared with values from reference simulations [215] and experiments for DPPC [218], DOPC [212], and PSM [219]. Reference simulations were carried out using the CHARMM program with 8-12 Å force switching and 1 fs timesteps. Errors are reported with respect to the reference simulations. References for melting temperatures are DPPC [13], DOPC [14], and PSM [16].

Lipid	T_{melt} (K)	T_{sim} (K)	Experiment APL (Å^2)	Reference APL (Å^2)	Equilibrium APL (Å^2)	Error %
DPPC (8-10)	314	323	63.1	62.9	63.71	+1.3
DPPC (8-12)					61.69	-1.9
DOPC (8-10)	256	303	67.4	68.9	69.72	+1.2
DOPC (8-12)					68.33	-0.8
PSM (8-10)	314	323	55.0	55.1	58.64	+6.4
PSM (8-12)					56.92	+3.3

Table 7.13: Surface viscosity calculations for various CHARMM36 lipids. As expected, unsaturated DOPC lipids are more viscous than saturated DPPC ones; the liquid order mixture is more viscous than the liquid disorder mixture; the polar PSM lipids are the most viscous overall; and the cutoff treatment has a significant effect on surface viscosity.

Membrane Composition	Surface Viscosity (10^{-11} Pa-m-s) = (10^{-8} P*cm)
DPPC (8-10)	8.18 ± 0.50
DPPC (8-12)	12.26 ± 0.50
DOPC (8-12)	19.68 ± 0.69
PSM (8-12)	48.8 ± 1.2
L_o (8-12)	23.83 ± 0.92
L_d (8-12)	9.39 ± 0.47

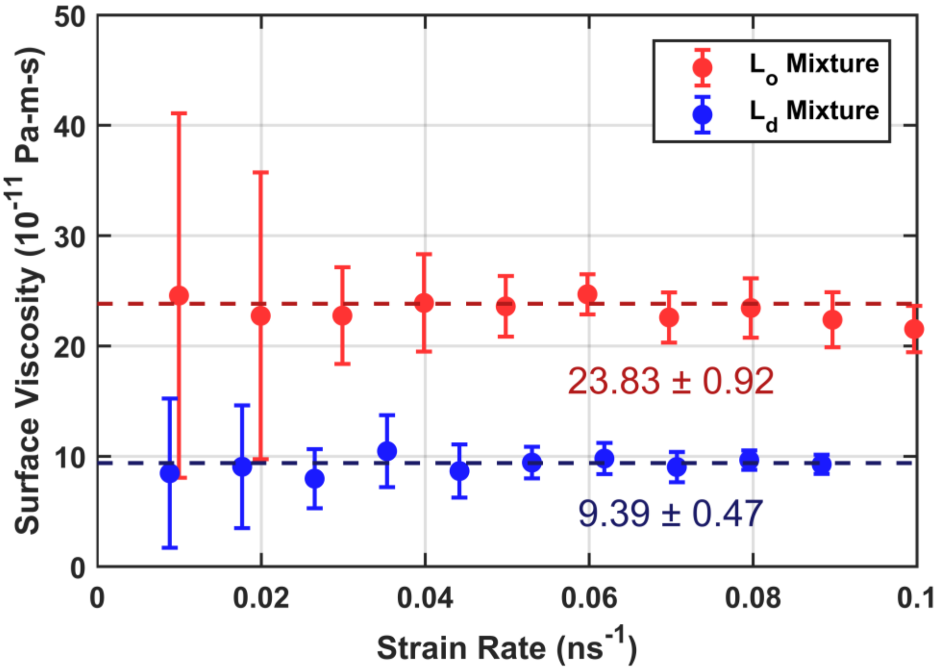

Figure 7.23: Surface viscosity as a function of shear rate for liquid ordered (L_o) and liquid disordered (L_d) lipid mixtures using the CHARMM36 force field. Dotted lines show the result of taking inverse-variance weighted averages for each series. Each data point is the average of five trials at the given strain rate.

7.5 Interleaflet Friction

7.5.1 Martini DPPC

Membranes equilibrated for surface viscosity were reused for interleaflet friction simulations. Each was padded with 20 nm of additional water followed by another 10 ns equilibration at fixed lateral size with the vertical box dimension coupled to a barostat at 1 bar pressure. The force applied to particles near the vertical boundaries was varied over several orders of magnitude to identify a regime where parallel shear flows appear in the solvent and the traction forces applied to the membrane leaflets cause them to smoothly slide past one another. The units of force used in the simulation (and reported below) are kJ/(mol-nm) ≅ 1.6605 pN. The force applied to the membrane must be directly proportional to the relative velocity between the leaflets Δv for Equation 7.13 to hold. With stick boundary conditions, this implies a direct relationship between Δv and the solvent shear rate. Thus, for our calculation of b to be accurate, it must be independent of the solvent shear rate we calculate from the velocity profile.

Three replicas with random initial particles velocities were used for each value of the boundary force magnitude. 1 nm was used for the thickness d of the boundary region in all simulations (recall **Fig. 7.3**). Production simulations ran for 1 μs in the NVT ensemble using the parameters from **Table 7.14**. Each trajectory was analyzed in three stages: first, verifying that the membrane does not drift vertically over the course of the simulation; second, calculating the relative velocity between leaflets from their separation as a function of time; and last, computing the average velocity profile along z was from the instantaneous particle velocities.

Table 7.14: Selected simulation parameters for friction simulations of Martini DPPC. Other parameters use defaults listed in **Table 2.1**.

Integration	Velocity-Verlet with $\Delta t = 20$ fs Center of mass motion removed every 500 timesteps, treating membrane and solvent separately.
Thermostat	Bussi velocity rescaling, treating water and lipids separately; $T_0 = 323$ K, $\tau_T = 1$ ps
Barostat	None (fixed volume)

Fig. 7.24 shows a typical plot of the vertical leaflet positions as a function of time. Since very little deviation and no overall drift is observed, we may regard its

vertical position as fixed; we do not need to be concerned about the possibility of the membrane contaminating the solvent velocity profile during the time averaging. Separation between leaflets and Δv fit are shown for simulations with insufficient forcing (**Fig. 7.25**) and sufficient forcing (**Fig. 7.26**) to generate steady traction forces. All simulations with boundary forces too weak to generate sufficient traction on the leaflets were discarded at this point. These include all simulations with boundary forces less than 0.05 kJ/(mol-nm). Those with values between 0.05 and 0.10 kJ/(mol-nm) were not discarded, though they generally feature unreliable traction forces leading to significantly higher uncertainties in their interleaflet friction calculations.

A typical velocity profile for a boundary force sufficient to generate a parallel shear flow in the solvent is demonstrated in **Fig. 7.27**. The location of the membrane, the linear shear flow in the solvent, and nonlinear solvent flows from forcing are all readily identifiable. The periodic boundaries were ignored in the spatial averaging as the velocity profile of the boundary region was not used in the analysis. The spatial averaging kernel (shown in the figure) obscures two features of the profile which are only seen during longer simulations with a narrower kernel. First, the linear velocity profile of the solvent does, despite the smooth appearance in **Fig. 7.27**, extend all the way to the solvent-membrane interface. Second, the average velocity does not smoothly vary over each leaflet; they are composed of solid molecules and move, on average, as monolithic slabs. Each leaflet actually moves at a fixed velocity $\pm v_L$ with an abrupt discontinuity at the membrane center equal in magnitude to $\Delta v = 2v_L$. Resolving the velocity difference Δv in this way is more error-prone than fitting the average separation as a function of time.

The results of this analysis are shown in **Fig. 7.28** for both sets of Martini DPPC parameters. The newer version of Martini has a significantly higher interleaflet friction (~25%), perhaps owing to the slight increase in attraction among tail beads. The result for Martini 2004 is the same order of magnitude as den Otter and Shkulipa results but differs almost by a factor of 2 [192]. They report 2.4×10^6 Pa-s/m from nonequilibrium simulations driven by Lees-Edwards boundary conditions and 3.0×10^6 Pa-s/m from equilibrium relaxation of membrane undulations. This is not a cause of great concern for application to the periodic Saffman-Delbrück theory, as the theory only weakly depends on this parameter [46].

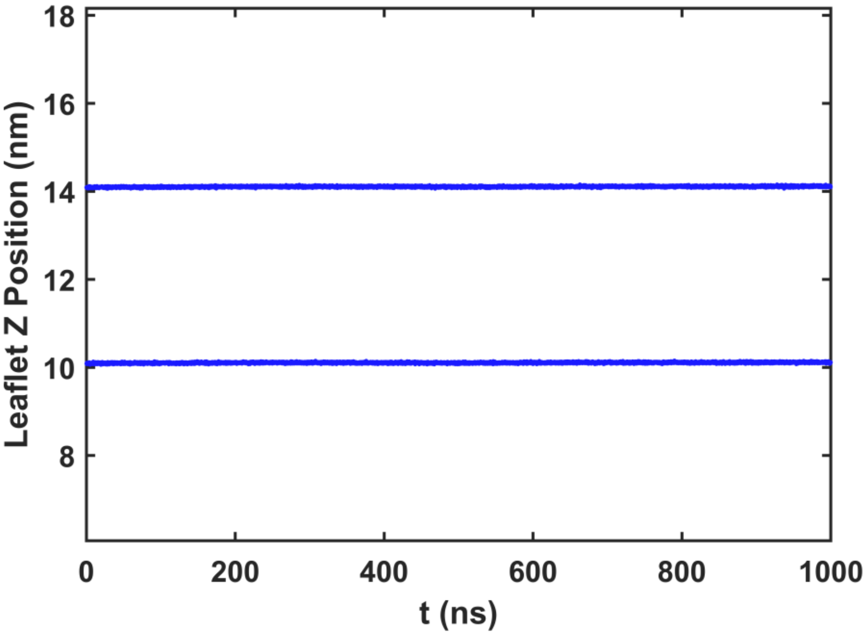

Figure 7.24: Sample plot of the average leaflet position along z as a function of time.

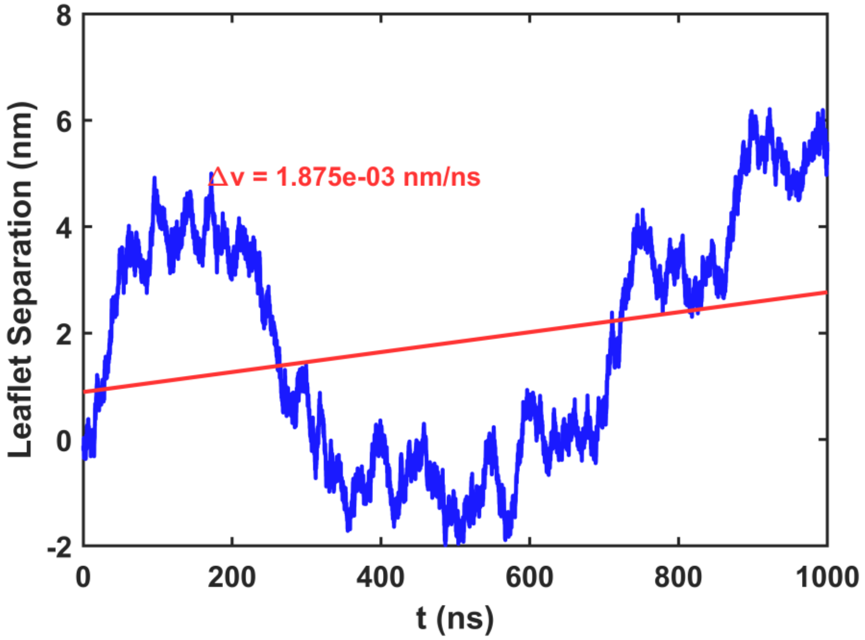

Figure 7.25: Distance between average leaflet positions as a function of time when the solvent exerts insufficient traction forces on the leaflets. In this particular case, the forcing at the boundaries was not strong enough to generate a consistent, steady parallel shear in the solvent.

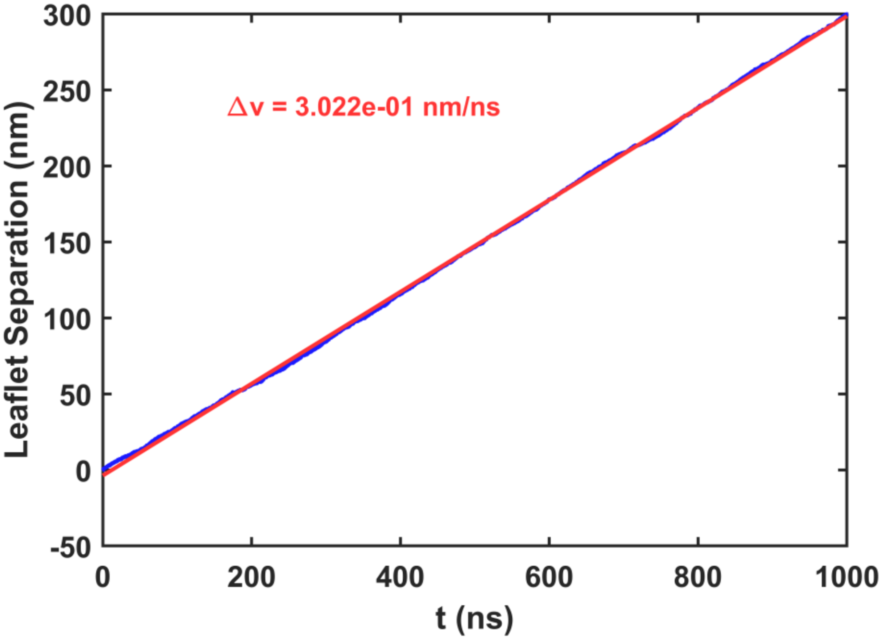

Figure 7.26: Distance between average leaflet positions as a function of time when the solvent exerts sufficient traction forces on the leaflets. Leaflets slide past one another at a relatively smooth, steady rate. The linear fit (red) gives their average relative velocity.

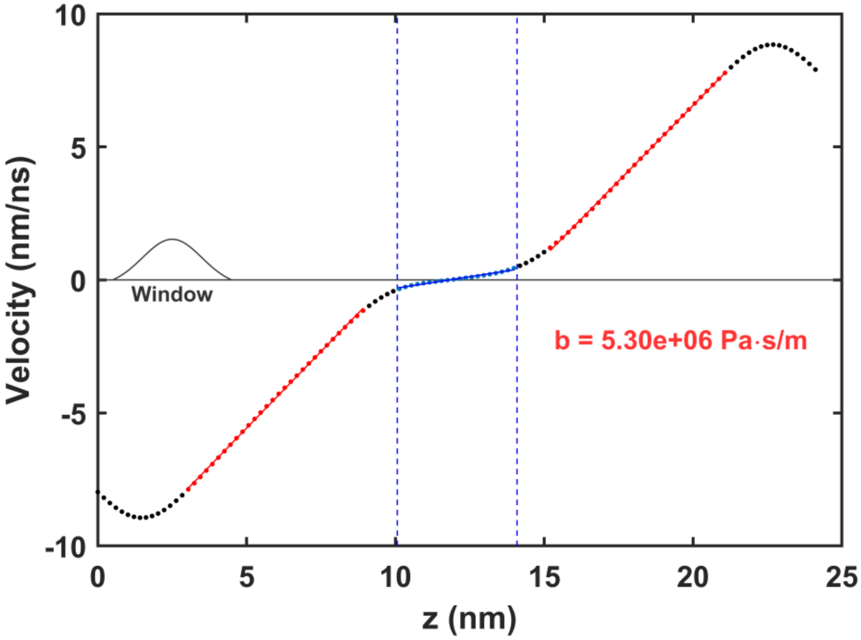

Figure 7.27: Average velocity profile for Martini DPPC with a 1.0 kJ/(mol-nm) force applied to all particles within 1 nm of the boundary at every timestep. The region corresponding to the membrane is highlighted in blue and the extent of linear parallel shear flow in the solvent is highlighted in red. Fits to these regions are used to determine the solvent shear rates. The spatial averaging window used to create the velocity profile is depicted on the left.

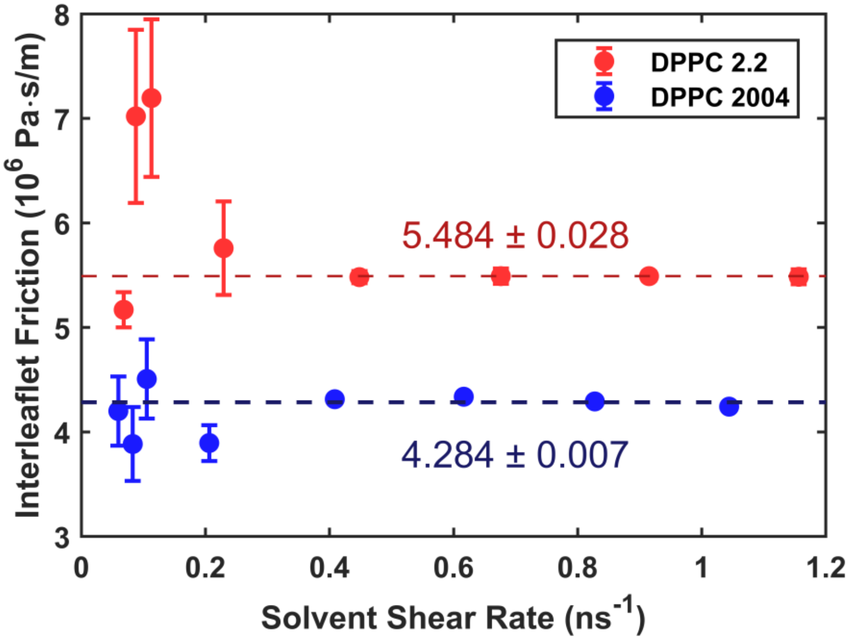

Figure 7.28: Interleaflet friction as a function of shear rate for DPPC membranes using force field parameters for the 2004 version of Martini cited by den Otter and Shkulipa and the more recent Martini v2.2. Dotted lines show the result of taking inverse-variance weighted averages for both. Error bars combine the standard error computed from each set of three trials with error propagation.

7.5.2 CHARMM36 DPPC

Interleaflet friction for the CHARMM36 force field was calculated using the same procedure as the Martini force field. The shorter timestep necessitated shorter simulations (only 100 ns) and the stronger attraction among CHARMM lipids required higher solvent shear rates to generate steady traction forces. A sample velocity profile is depicted in **Fig. 7.29** and the results for DPPC are shown in **Fig. 7.30**. The application of higher solvent shear rates (relative to Martini) leads to a rate dependence which complicates the interpretation of these results. Further work is required to characterize the rate-independent regime and extend the analysis to other

CHARMM36 lipid types, namely POPC and PSM. This work will be carried out during the remainder of the summer before being published together with the interleaflet friction results for Martini and the surface viscosity results for both CHARMM36 and Martini.

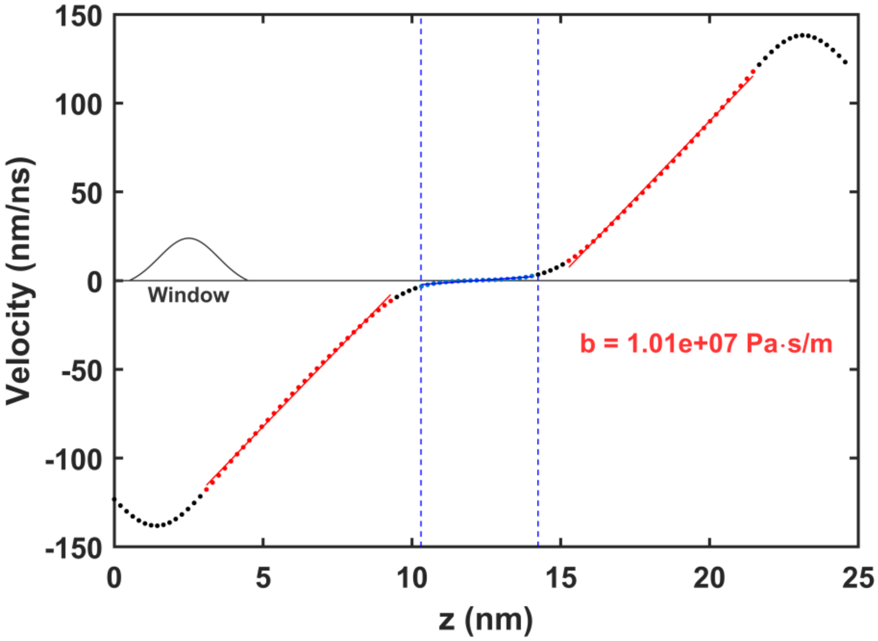

Figure 7.29: Average velocity profile for CHARMM36 DPPC with a 0.5 kJ/(mol-nm) force applied to all particles within 1 nm of the boundary at every timestep. The region corresponding to the membrane is highlighted in blue and the extent of linear parallel shear flow in the solvent is highlighted in red. Fits to these regions are used to determine the solvent shear rates. The spatial averaging window used to create the velocity profile is depicted on the left.

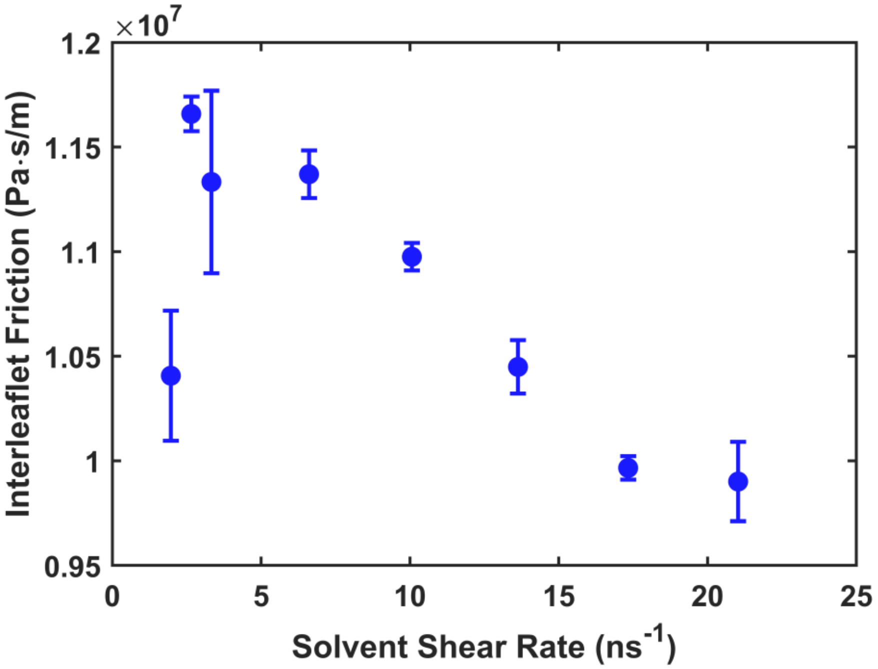

Figure 7.30: Interleaflet friction as a function of shear rate for CHARMM36 DPPC. Error bars combine the standard error computed from each set of three trials with error propagation. The use of higher shear rates leads to rate-dependent results. More simulations are required to resolve the rate-independent regime for shear rates $\lesssim 5$ ns^{-1}.

REFERENCES

[1] A. Zgorski and E. Lyman, “Toward Hydrodynamics with Solvent Free Lipid Models: STRD Martini,” *Biophys. J.*, vol. 111, no. 12, pp. 2689–2697, 2016.

[2] W. Humphrey, A. Dalke, and K. Schulten, “VMD: Visual molecular dynamics,” *J. Mol. Graph.*, vol. 14, no. 1, pp. 33–38, Feb. 1996.

[3] B. Alberts, A. Johnson, J. Lewis, M. Raff, K. Roberts, and P. Walter, *Molecular Biology of the Cell*, 5th ed. Garland Science, 2007.

[4] M. Almén, K. J. V. Nordström, R. Fredriksson, and H. B. Schiöth, “Mapping the human membrane proteome: a majority of the human membrane proteins can be classified according to function and evolutionary origin,” *BMC Biol.*, vol. 7, no. 1, p. 50, 2009.

[5] M. Rask-Andersen, M. S. Almén, and H. B. Schiöth, “Trends in the exploitation of novel drug targets,” *Nat. Rev. Drug Discov.*, vol. 10, no. 8, pp. 579–590, Aug. 2011.

[6] P. L. Yeagle, Ed., *The Structure of Biological Membranes*, 3rd ed. Boca Raton: CRC Press, 2011.

[7] P. G. Saffman and M. Delbrück, “Brownian motion in biological membranes,” *Proc Natl Acad Sci USA*, vol. 72, no. 8, pp. 3111–3113, 1975.

[8] M. Deserno, “Fluid lipid membranes: From differential geometry to curvature stresses,” *Chem. Phys. Lipids*, vol. 185, pp. 11–45, Jan. 2015.

[9] J. P. Overington, B. Al-Lazikani, and A. L. Hopkins, “How many drug targets are there?,” *Nat. Rev. Drug Discov.*, vol. 5, no. 12, pp. 993–996, 2006.

[10] M. J. Gerl *et al.*, “Quantitative analysis of the lipidomes of the influenza virus envelope and MDCK cell apical membrane,” *J. Cell Biol.*, vol. 196, no. 2, pp. 213–221, Jan. 2012.

[11] K. Simons and E. Ikonen, “Functional rafts in cell membranes,” *Nature*, vol. 387, no. 6633, pp. 569–572, Jun. 1997.

[12] G. van Meer, D. R. Voelker, and G. W. Feigenson, “Membrane lipids: where

they are and how they behave," *Nat. Rev. Mol. Cell Biol.*, vol. 9, no. 2, pp. 112–124, Feb. 2008.

[13] R. L. Biltonen and D. Lichtenberg, "The use of differential scanning calorimetry as a tool to characterize liposome preparations," *Chem. Phys. Lipids*, vol. 64, no. 1–3, pp. 129–142, Sep. 1993.

[14] A. S. Ulrich, M. Sami, and A. Watts, "Hydration of DOPC bilayers by differential scanning calorimetry," *Biochim. Biophys. Acta - Biomembr.*, vol. 1191, no. 1, pp. 225–230, Apr. 1994.

[15] B. J. Litman, E. N. Lewis, and I. W. Levin, "Packing characteristics of highly unsaturated bilayer lipids: Raman spectroscopic studies of multilamellar phosphatidylcholine dispersions," *Biochemistry*, vol. 30, no. 2, pp. 313–319, Jan. 1991.

[16] P. R. Maulik and G. G. Shipley, "N-Palmitoyl Sphingomyelin Bilayers: Structure and Interactions with Cholesterol and Dipalmitoylphosphatidylcholine," *Biochemistry*, vol. 35, no. 24, pp. 8025–8034, Jan. 1996.

[17] R. J. Lewis, *Hawley's Condensed Chemical Dictionary*, 13th ed. John Wiley & Sons, 1997.

[18] "ChemSketch." Advanced Chemistry Development, Inc., Toronto, 2015.

[19] S. A. Shelby, D. Holowka, B. Baird, and S. L. Veatch, "Distinct stages of stimulated fcεri receptor clustering and immobilization are identified through superresolution imaging," *Biophys. J.*, vol. 105, no. 10, pp. 2343–2354, 2013.

[20] K. Spendier, K. A. Lidke, D. S. Lidke, and J. L. Thomas, "Single-particle tracking of immunoglobulin e receptors (FcRI) in micron-sized clusters and receptor patches," *FEBS Lett.*, vol. 586, no. 4, pp. 416–421, 2012.

[21] A. Kusumi, H. Ike, C. Nakada, K. Murase, and T. Fujiwara, "Single-molecule tracking of membrane molecules: Plasma membrane compartmentalization and dynamic assembly of raft-philic signaling molecules," *Semin. Immunol.*, vol. 17, no. 1, pp. 3–21, 2005.

[22] C. Eggeling *et al.*, "Direct observation of the nanoscale dynamics of membrane lipids in a living cell," *Nature*, vol. 457, no. 7233, pp. 1159–1162, 2009.

[23] D. M. Andrade *et al.*, "Cortical actin networks induce spatio-temporal confinement of phospholipids in the plasma membrane – a minimally invasive

investigation by STED-FCS," *Sci. Rep.*, vol. 5, no. 1, p. 11454, 2015.

[24] S. K. Saka, A. Honigmann, C. Eggeling, S. W. Hell, T. Lang, and S. O. Rizzoli, "Multi-protein assemblies underlie the mesoscale organization of the plasma membrane," *Nat. Commun.*, vol. 5, no. May, pp. 1–14, 2014.

[25] F. A. Heberle *et al.*, "Bilayer thickness mismatch controls domain size in model membranes," *J. Am. Chem. Soc.*, vol. 135, no. 18, pp. 6853–6859, 2013.

[26] A. Honigmann, V. Mueller, S. W. Hell, and C. Eggeling, "STED microscopy detects and quantifies liquid phase separation in lipid membranes using a new far-red emitting fluorescent phosphoglycerolipid analogue," *Faraday Discuss.*, vol. 161, pp. 77–89, 2013.

[27] A. J. Sodt, M. L. Sandar, K. Gawrisch, R. W. Pastor, and E. Lyman, "The Molecular Structure of the Liquid-Ordered Phase of Lipid Bilayers," *J. Am. Chem. Soc.*, vol. 136, no. 2, pp. 725–732, Jan. 2014.

[28] A. J. Sodt, R. W. Pastor, and E. Lyman, "Hexagonal Substructure and Hydrogen Bonding in Liquid-Ordered Phases Containing Palmitoyl Sphingomyelin," *Biophys. J.*, vol. 109, no. 5, pp. 948–955, 2015.

[29] H.-M. Wu, Y.-H. Lin, T.-C. Yen, and C.-L. Hsieh, "Nanoscopic substructures of raft-mimetic liquid-ordered membrane domains revealed by high-speed single-particle tracking," *Sci. Rep.*, vol. 6, no. 1, p. 20542, 2016.

[30] S. J. Bussell, D. a. Hammer, and D. L. Koch, "The effect of hydrodynamic interactions on the tracer and gradient diffusion of integral membrane proteins in lipid bilayers," *J. Fluid Mech.*, vol. 258, no. 1, p. 167, 2006.

[31] B. D. Hughes, B. A. Pailthorpe, and L. R. White, "The translational and rotational drag on a cylinder moving in a membrane," *J. Fluid Mech.*, vol. 110, no. 1, p. 349, Sep. 1981.

[32] B. J. Alder and T. E. Wainwright, "Studies in Molecular Dynamics. I. General Method," *J. Chem. Phys.*, vol. 31, no. 2, pp. 459–466, Aug. 1959.

[33] A. Rahman, "Correlations in the Motion of Atoms in Liquid Argon," *Phys. Rev.*, vol. 136, no. 2A, pp. A405–A411, Oct. 1964.

[34] G. Zhao *et al.*, "Mature HIV-1 capsid structure by cryo-electron microscopy and all-atom molecular dynamics," *Nature*, vol. 497, no. 7451, pp. 643–646, 2013.

[35] T. Reddy *et al.*, “Nothing to Sneeze At: A Dynamic and Integrative Computational Model of an Influenza A Virion,” *Structure*, vol. 23, no. 3, pp. 584–597, 2015.

[36] D. E. Shaw *et al.*, “Atomic-Level Characterization of the Structural Dynamics of Proteins,” *Science (80-.).*, vol. 330, no. 6002, pp. 341–346, Oct. 2010.

[37] R. M. Venable, Y. Zhang, B. J. Hardy, and R. W. Pastor, “Molecular Dynamics Simulations of a Lipid Bilayer and of Hexadecane: An Investigation of Membrane Fluidity,” *Science (80-.).*, vol. 262, no. 5131, pp. 223–226, 1993.

[38] R. Goetz and R. Lipowsky, “Computer simulations of bilayer membranes: Self-assembly and interfacial tension,” *J. Chem. Phys.*, vol. 108, no. 17, pp. 7397–7409, 1998.

[39] I. R. Cooke, K. Kremer, and M. Deserno, “Tunable generic model for fluid bilayer membranes,” *Phys. Rev. E - Stat. Nonlinear, Soft Matter Phys.*, vol. 72, no. 1, pp. 2–5, 2005.

[40] A. J. Sodt, M. L. Sandar, K. Gawrisch, R. W. Pastor, E. Lyman, and R. W. Pastor, “The Molecular Structure of the Liquid Ordered Phase of Lipid Bilayers The Molecular Structure of the Liquid Ordered Phase of Lipid Bilayers,” *Jacs*, vol. 136, pp. 725–732, 2013.

[41] M. Javanainen, H. Martinez-Seara, and I. Vattulainen, “Nanoscale Membrane Domain Formation Driven by Cholesterol,” *Sci. Rep.*, vol. 7, no. 1, p. 1143, 2017.

[42] H. Koldsø, T. Reddy, P. W. Fowler, A. L. Duncan, and M. S. P. Sansom, “Membrane Compartmentalization Reducing the Mobility of Lipids and Proteins within a Model Plasma Membrane,” *J. Phys. Chem. B*, vol. 120, no. 34, pp. 8873–8881, 2016.

[43] M. Javanainen, H. Martinez-Seara, R. Metzler, and I. Vattulainen, “Diffusion of Integral Membrane Proteins in Protein-Rich Membranes,” *J. Phys. Chem. Lett.*, vol. 8, no. 17, pp. 4308–4313, 2017.

[44] B. A. Camley, M. G. Lerner, R. W. Pastor, and F. L. H. Brown, “Strong influence of periodic boundary conditions on lateral diffusion in lipid bilayer membranes,” *J. Chem. Phys.*, vol. 143, no. 24, 2015.

[45] M. Vögele and G. Hummer, “Divergent Diffusion Coefficients in Simulations of Fluids and Lipid Membranes,” *J. Phys. Chem. B*, vol. 120, no. 33, pp. 8722–8732, 2016.

[46] R. M. Venable *et al.*, “Lipid and Peptide Diffusion in Bilayers: The Saffman-Delbrück Model and Periodic Boundary Conditions,” *J. Phys. Chem. B*, vol. 121, no. 15, pp. 3443–3457, 2017.

[47] M. J. Abraham *et al.*, “GROMACS: High performance molecular simulations through multi-level parallelism from laptops to supercomputers,” *SoftwareX*, vol. 1–2, pp. 19–25, Sep. 2015.

[48] C. Arnarez *et al.*, “Dry martini, a coarse-grained force field for lipid membrane simulations with implicit solvent,” *J. Chem. Theory Comput.*, vol. 11, no. 1, pp. 260–275, 2015.

[49] D. Frenkel and B. Smit, *Understanding Molecular Simulation: From Algorithms to Applications*, 2nd ed. Academic Press, 2002.

[50] H. Berendsen and W. van Gunsteren, “Practical algorithms for dynamic simulations,” *Mol. Simul. Stat. Mech. Syst.*, pp. 43–65, 1986.

[51] R. B. Best *et al.*, “Optimization of the additive CHARMM all-atom protein force field targeting improved sampling of the backbone φ, ψ and side-chain χ1and χ2Dihedral Angles,” *J. Chem. Theory Comput.*, vol. 8, no. 9, pp. 3257–3273, 2012.

[52] J. P. M. Jämbeck and A. P. Lyubartsev, “An extension and further validation of an all-atomistic force field for biological membranes,” *J. Chem. Theory Comput.*, vol. 8, no. 8, pp. 2938–2948, 2012.

[53] E. Harder *et al.*, “OPLS3: A Force Field Providing Broad Coverage of Drug-like Small Molecules and Proteins,” *J. Chem. Theory Comput.*, vol. 12, no. 1, pp. 281–296, Jan. 2016.

[54] S. J. Marrink, H. J. Risselada, S. Yefimov, D. P. Tieleman, and A. H. De Vries, “The MARTINI force field: Coarse grained model for biomolecular simulations,” *J. Phys. Chem. B*, vol. 111, no. 27, pp. 7812–7824, 2007.

[55] N. Schmid *et al.*, “Definition and testing of the GROMOS force-field versions 54A7 and 54B7,” *Eur. Biophys. J.*, vol. 40, no. 7, pp. 843–856, 2011.

[56] H. C. Andersen, “Molecular dynamics simulations at constant pressure and/or temperature,” *J. Chem. Phys.*, vol. 72, no. 4, pp. 2384–2393, Feb. 1980.

[57] H. J. C. Berendsen, J. P. M. Postma, W. F. Van Gunsteren, A. Dinola, and J. R. Haak, “Molecular dynamics with coupling to an external bath,” *J. Chem. Phys.*, vol. 81, no. 8, pp. 3684–3690, 1984.

[58] S. Nosé, “A molecular dynamics method for simulations in the canonical ensemble,” *Mol. Phys.*, vol. 52, no. 2, pp. 255–268, Jun. 1984.

[59] W. G. Hoover, “Canonical dynamics: Equilibrium phase-space distributions,” *Phys. Rev. A*, vol. 31, no. 3, pp. 1695–1697, Mar. 1985.

[60] G. Bussi, D. Donadio, and M. Parrinello, “Canonical sampling through velocity rescaling,” *J. Chem. Phys.*, vol. 126, no. 1, 2007.

[61] M. Parrinello and A. Rahman, “Polymorphic transitions in single crystals: A new molecular dynamics method,” *J. Appl. Phys.*, vol. 52, no. 12, pp. 7182–7190, 1981.

[62] S. Nosé and M. L. Klein, “Constant pressure molecular dynamics for molecular systems,” *Mol. Phys.*, vol. 50, no. 5, pp. 1055–1076, Dec. 1983.

[63] G. J. Martyna, D. J. Tobias, and M. L. Klein, “Constant pressure molecular dynamics algorithms,” *J. Chem. Phys.*, vol. 101, no. 5, pp. 4177–4189, Sep. 1994.

[64] G. J. Martyna, M. E. Tuckerman, D. J. Tobias, and M. L. Klein, “Explicit reversible integrators for extended systems dynamics,” *Mol. Phys.*, vol. 87, no. 5, pp. 1117–1157, Apr. 1996.

[65] S. E. Feller, Y. Zhang, R. W. Pastor, and B. R. Brooks, “Constant pressure molecular dynamics simulation: The Langevin piston method,” *J. Chem. Phys.*, vol. 103, no. 11, pp. 4613–4621, Sep. 1995.

[66] G. J. Martyna, M. L. Klein, and M. Tuckerman, “Nosé–Hoover chains: The canonical ensemble via continuous dynamics,” *J. Chem. Phys.*, vol. 97, no. 4, pp. 2635–2643, Aug. 1992.

[67] H. Watanabe and H. Kobayashi, “Ergodicity of a thermostat family of the Nosé-Hoover type,” *Phys. Rev. E*, vol. 75, no. 4, p. 040102, Apr. 2007.

[68] E. H. Lee, J. Hsin, M. Sotomayor, G. Comellas, and K. Schulten, “Discovery Through the Computational Microscope,” *Structure*, vol. 17, no. 10, pp. 1295–1306, 2009.

[69] J. C. Phillips *et al.*, “Scalable molecular dynamics with NAMD,” *J. Comput. Chem.*, vol. 26, no. 16, pp. 1781–1802, Dec. 2005.

[70] B. R. Brooks *et al.*, “CHARMM: The Biomolecular Simulation Program,” *J. Comput. Chem.*, vol. 30, no. 10, pp. 1545–1614, 2009.

[71] S. Plimpton, “Fast Parallel Algorithms for Short-Range Molecular Dynamics,” *J. Comput. Phys.*, vol. 117, no. 1, pp. 1–19, Mar. 1995.

[72] I. T. Todorov, W. Smith, K. Trachenko, and M. T. Dove, “DL_POLY_3: new dimensions in molecular dynamics simulations via massive parallelism,” *J. Mater. Chem.*, vol. 16, no. 20, p. 1911, 2006.

[73] R. Salomon-Ferrer, D. A. Case, and R. C. Walker, “An overview of the Amber biomolecular simulation package,” *Wiley Interdiscip. Rev. Comput. Mol. Sci.*, vol. 3, no. 2, pp. 198–210, Mar. 2013.

[74] M. P. Allen and D. J. Tildesley, *Computer simulation of liquids*. 1991.

[75] C. W. Gear, *Numerical Initial Value Problems in Ordinary Differential Equations*. Prentice Hall PTR, 1971.

[76] R. W. Hockney, S. P. Goel, and J. W. Eastwood, “Quiet high-resolution computer models of a plasma,” *J. Comput. Phys.*, vol. 14, no. 2, pp. 148–158, 1974.

[77] W. C. Swope, H. C. Andersen, P. H. Berens, and K. R. Wilson, “A computer simulation method for the calculation of equilibrium constants for the formation of physical clusters of molecules: Application to small water clusters,” *J. Chem. Phys.*, vol. 76, no. 1, pp. 637–649, Jan. 1982.

[78] L. Verlet, “Computer ‘Experiments’ on Classical Fluids. I. Thermodynamical Properties of Lennard-Jones Molecules,” *Phys. Rev.*, vol. 159, no. 1, pp. 98–103, Jul. 1967.

[79] B. Dünweg, “Molecular dynamics algorithms and hydrodynamic screening,” *J. Chem. Phys.*, vol. 99, no. 9, pp. 6977–6982, 1993.

[80] C. Sagui and T. A. Darden, “MOLECULAR DYNAMICS SIMULATIONS OF BIOMOLECULES: Long-Range Electrostatic Effects,” *Annu. Rev. Biophys. Biomol. Struct.*, vol. 28, no. 1, pp. 155–179, Jun. 1999.

[81] P. P. Ewald, “Die Berechnung optischer und elektrostatischer Gitterpotentiale,” *Ann. Phys.*, vol. 369, no. 3, pp. 253–287, 1921.

[82] J. W. Perram, H. G. Petersen, and S. W. De Leeuw, “An algorithm for the simulation of condensed matter which grows as the 3/2 power of the number of particles,” *Mol. Phys.*, vol. 65, no. 4, pp. 875–893, Nov. 1988.

[83] T. Darden, D. York, and L. Pedersen, “Particle mesh Ewald: An N ·log(N)

method for Ewald sums in large systems," *J. Chem. Phys.*, vol. 98, no. 12, pp. 10089–10092, Jun. 1993.

[84] L. Greengard, V. Rokhlin, and L. G. and V. Rokhlin, "A fast algorithm for particle simulations," *J. Comput. Phys.*, vol. 73, no. 2, pp. 325–348, 1987.

[85] L. Greengard, "Fast Algorithms for Classical Physics," *Science (80-.).*, vol. 265, no. 5174, pp. 909–914, Aug. 1994.

[86] L. Greengard and V. Rokhlin, "A new version of the Fast Multipole Method for the Laplace equation in three dimensions," *Acta Numer.*, vol. 6, no. 1997, pp. 229–269, 1997.

[87] E. L. Pollock and J. Glosli, "Comments on P3M, FMM, and the Ewald method for large periodic Coulombic systems," *Comput. Phys. Commun.*, vol. 95, no. 2–3, pp. 93–110, Jun. 1996.

[88] A. Arnold *et al.*, "Comparison of scalable fast methods for long-range interactions," *Phys. Rev. E*, vol. 88, no. 6, p. 063308, Dec. 2013.

[89] A. D. MacKerell *et al.*, "All-Atom Empirical Potential for Molecular Modeling and Dynamics Studies of Proteins †," *J. Phys. Chem. B*, vol. 102, no. 18, pp. 3586–3616, Apr. 1998.

[90] R. Bukowski, K. Szalewicz, G. C. Groenenboom, and A. van der Avoird, "Predictions of the Properties of Water from First Principles," *Science (80-.).*, vol. 315, no. 5816, pp. 1249–1252, Mar. 2007.

[91] W. L. Jorgensen, J. Chandrasekhar, J. D. Madura, R. W. Impey, and M. L. Klein, "Comparison of simple potential functions for simulating liquid water," *J. Chem. Phys.*, vol. 79, no. 2, pp. 926–935, Jul. 1983.

[92] K. Szalewicz, C. Leforestier, and A. van der Avoird, "Towards the complete understanding of water by a first-principles computational approach," *Chem. Phys. Lett.*, vol. 482, no. 1–3, pp. 1–14, Nov. 2009.

[93] A. V. Onufriev and S. Izadi, "Water models for biomolecular simulations," *Wiley Interdiscip. Rev. Comput. Mol. Sci.*, vol. 8, no. 2, p. e1347, Mar. 2018.

[94] O. Lenz and F. Schmid, "A simple computer model for liquid lipid bilayers," *J. Mol. Liq.*, vol. 117, no. 1–3, pp. 147–152, 2005.

[95] S. Izvekov and G. A. Voth, "Solvent-free lipid bilayer model using multiscale coarse-graining," *J. Phys. Chem. B*, vol. 113, no. 13, pp. 4443–4455, 2009.

[96] Z.-J. Wang and M. Deserno, “A Solvent-Free Coarse-Grained Model for Quantitative POPC Bilayer Simulations,” *Biophys. J.*, vol. 96, no. 3, p. 365a, Feb. 2009.

[97] S. Patel, A. D. Mackerell, and C. L. Brooks, “CHARMM fluctuating charge force field for proteins: II Protein/solvent properties from molecular dynamics simulations using a nonadditive electrostatic model,” *J. Comput. Chem.*, vol. 25, no. 12, pp. 1504–1514, Sep. 2004.

[98] H. Yu *et al.*, “Simulating Monovalent and Divalent Ions in Aqueous Solution Using a Drude Polarizable Force Field,” *J. Chem. Theory Comput.*, vol. 6, no. 3, pp. 774–786, Mar. 2010.

[99] S. O. Yesylevskyy, L. V. Schäfer, D. Sengupta, and S. J. Marrink, “Polarizable Water Model for the Coarse-Grained MARTINI Force Field,” *PLoS Comput. Biol.*, vol. 6, no. 6, p. e1000810, Jun. 2010.

[100] U. Góra, R. Podeszwa, W. Cencek, and K. Szalewicz, “Interaction energies of large clusters from many-body expansion,” *J. Chem. Phys.*, vol. 135, no. 22, p. 224102, Dec. 2011.

[101] G. R. Medders, A. W. Götz, M. A. Morales, P. Bajaj, and F. Paesani, “On the representation of many-body interactions in water,” *J. Chem. Phys.*, vol. 143, no. 10, p. 104102, Sep. 2015.

[102] A. D. MacKerell, N. Banavali, and N. Foloppe, “Development and current status of the CHARMM force field for nucleic acids,” *Biopolymers*, vol. 56, no. 4, pp. 257–265, 2000.

[103] J. B. Klauda *et al.*, “Update of the CHARMM All-Atom Additive Force Field for Lipids: Validation on Six Lipid Types,” *J. Phys. Chem. B*, vol. 114, no. 23, pp. 7830–7843, Jun. 2010.

[104] J. Lee *et al.*, “CHARMM-GUI Input Generator for NAMD, GROMACS, AMBER, OpenMM, and CHARMM/OpenMM Simulations Using the CHARMM36 Additive Force Field,” *J. Chem. Theory Comput.*, vol. 12, no. 1, pp. 405–413, Jan. 2016.

[105] J. B. Klauda, R. M. Venable, A. D. MacKerell, and R. W. Pastor, “Chapter 1 Considerations for Lipid Force Field Development,” in *Current Topics in Membranes*, vol. 60, no. 08, 2008, pp. 1–48.

[106] R. W. Pastor and A. D. MacKerell, “Development of the CHARMM Force Field for Lipids,” *J. Phys. Chem. Lett.*, vol. 2, no. 13, pp. 1526–1532, Jul. 2011.

[107] B. Hess, H. Bekker, H. J. C. Berendsen, and J. G. E. M. Fraaije, "LINCS: A linear constraint solver for molecular simulations," *J. Comput. Chem.*, vol. 18, no. 12, pp. 1463–1472, Sep. 1997.

[108] B. Hess, "P-LINCS: A Parallel Linear Constraint Solver for Molecular Simulation," *J. Chem. Theory Comput.*, vol. 4, no. 1, pp. 116–122, Jan. 2008.

[109] L. Monticelli, S. K. Kandasamy, X. Periole, R. G. Larson, D. P. Tieleman, and S. J. Marrink, "The MARTINI coarse grained force field: extension to proteins," *J. Chem. Theory Comput.*, vol. 4, no. 5, pp. 819–834, 2008.

[110] D. H. De Jong *et al.*, "Improved parameters for the martini coarse-grained protein force field," *J. Chem. Theory Comput.*, vol. 9, no. 1, pp. 687–697, 2013.

[111] S. J. Marrink and D. P. Tieleman, "Perspective on the Martini model," *Chem. Soc. Rev.*, vol. 42, no. 16, p. 6801, 2013.

[112] D. Fritz, K. Koschke, V. A. Harmandaris, N. F. A. Van Der Vegt, and K. Kremer, "Multiscale modeling of soft matter: Scaling of dynamics," *Phys. Chem. Chem. Phys.*, vol. 13, no. 22, pp. 10412–10420, 2011.

[113] S. Baoukina, S. J. Marrink, and D. P. Tieleman, "Molecular structure of membrane tethers," *Biophys. J.*, vol. 102, no. 8, pp. 1866–1871, 2012.

[114] S. Baoukina, E. Mendez-Villuendas, and D. P. Tieleman, "Molecular view of phase coexistence in lipid monolayers," *J. Am. Chem. Soc.*, vol. 134, no. 42, pp. 17543–17553, 2012.

[115] E. Flenner, J. Das, M. C. Rheinstädter, and I. Kosztin, "Subdiffusion and lateral diffusion coefficient of lipid atoms and molecules in phospholipid bilayers," *Phys. Rev. E*, vol. 79, no. 1, p. 011907, Jan. 2009.

[116] L. D. Landau and E. M. Lifshitz, *Fluid Mechanics*, vol. 6. Elsevier, 1987.

[117] J. Happel and H. Brenner, *Low Reynolds number hydrodynamics*. Dordrecht: Springer Netherlands, 1981.

[118] C. Pozrikidis, *Boundary Integral and Singularity Methods for Linearized Viscous Flow*. Cambridge University Press, 1992.

[119] F. L. H. Brown, "Continuum simulations of biomembrane dynamics and the importance of hydrodynamic effects," *Q. Rev. Biophys.*, vol. 44, no. 04, pp. 391–432, 2011.

[120] J. B. Fournier, "On the hydrodynamics of bilayer membranes," *Int. J. Non.*

Linear. Mech., vol. 75, pp. 67–76, 2015.

[121] E. M. Purcell, “Life at low Reynolds number,” *Am. J. Phys.*, vol. 45, no. 1, pp. 3–11, 1977.

[122] M. Lisicki, “Four approaches to hydrodynamic Green’s functions -- the Oseen tensors,” pp. 1–5, Dec. 2013.

[123] Z. Zapryanov and S. Tabakova, *Dynamics of Bubbles, Drops and Rigid Particles*, vol. 50. Dordrecht: Springer Netherlands, 1999.

[124] C. W. Oseen, *Neuere Methoden und Ergebnisse in der Hydrodynamik*. Leipzig: Akademische Verlagsgesellschaft, 1927.

[125] I. C. Yeh and G. Hummer, “System-size dependence of diffusion coefficients and viscosities from molecular dynamics simulations with periodic boundary conditions,” *J. Phys. Chem. B*, vol. 108, no. 40, pp. 15873–15879, 2004.

[126] N. Oppenheimer and H. Diamant, “Correlated diffusion of membrane proteins and their effect on membrane viscosity,” *Biophys. J.*, vol. 96, no. 8, pp. 3041–3049, 2009.

[127] S. Ramadurai and a. Holt, “Lateral diffusion of membrane proteins,” *J. Am. Chem. Soc.*, vol. 131, no. 35, pp. 12650–6, 2009.

[128] P. Cicuta, S. L. Keller, and S. L. Veatch, “Diffusion of liquid domains in lipid bilayer membranes,” *J. Phys. Chem. B*, vol. 111, no. 13, pp. 3328–3331, 2007.

[129] Y. Gambin *et al.*, “Lateral mobility of proteins in liquid membranes revisited,” *Proc. Natl. Acad. Sci.*, vol. 103, no. 7, pp. 2098–2102, 2006.

[130] A. Naji, A. J. Levine, and P. A. Pincus, “Corrections to the Saffman-Delbruck mobility for membrane bound proteins,” *Biophys. J.*, vol. 93, no. 11, pp. L49-51, 2007.

[131] H. A. Stone and A. Ajdari, “Hydrodynamics of particles embedded in a flat surfactant layer overlying a subphase of finite depth,” *J. Fluid Mech.*, vol. 369, no. 1998, pp. 151–173, 1998.

[132] N. Oppenheimer and H. Diamant, “Correlated dynamics of inclusions in a supported membrane,” *Phys. Rev. E - Stat. Nonlinear, Soft Matter Phys.*, vol. 82, no. 4, pp. 1–11, 2010.

[133] N. Oppenheimer and H. Diamant, “In-plane dynamics of membranes with immobile inclusions,” *Phys. Rev. Lett.*, vol. 107, no. 25, pp. 1–5, 2011.

[134] C. S. Peskin, “The immersed boundary method,” *Acta Numer.*, vol. 11, no. 2002, pp. 479–517, 2002.

[135] P. J. Atzberger, “A note on the correspondence of an immersed boundary method incorporating thermal fluctuations with Stokesian-Brownian dynamics,” *Phys. D Nonlinear Phenom.*, vol. 226, no. 2, pp. 144–150, 2007.

[136] A. Malevanets and R. Kapral, “Mesoscopic model for solvent dynamics,” *J. Chem. Phys.*, vol. 110, no. 17, pp. 8605–8613, May 1999.

[137] J. T. Padding and A. A. Louis, “Hydrodynamic interactions and Brownian forces in colloidal suspensions: Coarse-graining over time and length scales,” *Phys. Rev. E - Stat. Nonlinear, Soft Matter Phys.*, vol. 74, no. 3, pp. 1–29, 2006.

[138] G. Gompper, T. Ihle, D. M. Kroll, and R. G. Winkler, “Multi-Particle Collision Dynamics -- a Particle-Based Mesoscale Simulation Approach to the Hydrodynamics of Complex Fluids,” *Adv. Polmer Sci.*, Aug. 2008.

[139] R. Kapral, “Multiparticle Collision Dynamics: Simulation of Complex Systems on Mesoscales,” in *Advances in Chemical Physics*, vol. 140, 2008, pp. 89–146.

[140] P. J. Hoogerbrugge and J. M. V. A. Koelman, “Simulating Microscopic Hydrodynamic Phenomena with Dissipative Particle Dynamics,” *Europhys. Lett.*, vol. 19, no. 3, pp. 155–160, Jun. 1992.

[141] S. Chen and G. D. Doolen, “LATTICE BOLTZMANN METHOD FOR FLUID FLOWS,” *Annu. Rev. Fluid Mech.*, vol. 30, no. 1, pp. 329–364, Jan. 1998.

[142] F. Sterpone, P. Derreumaux, and S. Melchionna, “Molecular Mechanism of Protein Unfolding under Shear: A Lattice Boltzmann Molecular Dynamics Study,” *J. Phys. Chem. B*, p. acs.jpcb.7b10796, 2018.

[143] T. Ando and J. Skolnick, “On the importance of hydrodynamic interactions in lipid membrane formation,” *Biophys. J.*, vol. 104, no. 1, pp. 96–105, 2013.

[144] E. Noruzifar, B. A. Camley, and F. L. H. Brown, “Calculating hydrodynamic interactions for membrane-embedded objects,” *J. Chem. Phys.*, vol. 141, no. 12, 2014.

[145] E. Allahyarov and G. Gompper, “Mesoscopic solvent simulations: Multiparticle-collision dynamics of three-dimensional flows,” *Phys. Rev. E*, vol. 66, no. 3, p. 036702, Sep. 2002.

[146] M. Hecht, J. Harting, T. Ihle, and H. J. Herrmann, “Simulation of claylike

colloids," *Phys. Rev. E - Stat. Nonlinear, Soft Matter Phys.*, vol. 72, no. 1, pp. 1–16, 2005.

[147] T. Ihle and D. M. Kroll, "Stochastic rotation dynamics. II. Transport coefficients, numerics, and long-time tails," *Phys. Rev. E*, vol. 67, no. 6, p. 066706, Jun. 2003.

[148] C. C. Huang, G. Gompper, and R. G. Winkler, "Hydrodynamic correlations in multiparticle collision dynamics fluids," *Phys. Rev. E - Stat. Nonlinear, Soft Matter Phys.*, vol. 86, no. 5, pp. 1–10, 2012.

[149] H. Noguchi, N. Kikuchi, and G. Gompper, "Particle-based mesoscale hydrodynamic techniques," *Europhys. Lett.*, vol. 78, no. 1, p. 10005, Apr. 2007.

[150] T. Ihle and D. M. Kroll, "Stochastic rotation dynamics: A Galilean-invariant mesoscopic model for fluid flow," *Phys. Rev. E*, vol. 63, no. 2, p. 020201, Jan. 2001.

[151] T. Ihle and D. M. Kroll, "Stochastic rotation dynamics. I. Formalism, Galilean invariance, and Green-Kubo relations," *Phys. Rev. E*, vol. 67, no. 6, p. 066705, Jun. 2003.

[152] J. Hardy, O. De Pazzis, and Y. Pomeau, "Molecular dynamics of a classical lattice gas: Transport properties and time correlation functions," *Phys. Rev. A*, vol. 13, no. 5, pp. 1949–1961, 1976.

[153] D. d'Humières, P. Lallemand, and U. Frisch, "Lattice Gas Models for 3D Hydrodynamics," *Europhys. Lett.*, vol. 2, no. 4, pp. 291–297, Aug. 1986.

[154] G. R. McNamara and G. Zanetti, "Use of the Boltzmann Equation to Simulate Lattice-Gas Automata," *Phys. Rev. Lett.*, vol. 61, no. 20, pp. 2332–2335, Nov. 1988.

[155] F. J. Higuera and J. Jiménez, "Boltzmann Approach to Lattice Gas Simulations," *Europhys. Lett.*, vol. 9, no. 7, pp. 663–668, Aug. 1989.

[156] C. K. Aidun and J. R. Clausen, "Lattice-Boltzmann Method for Complex Flows," *Annu. Rev. Fluid Mech.*, vol. 42, no. 1, pp. 439–472, 2010.

[157] R. Adhikari, K. Stratford, M. E. Cates, and A. J. Wagner, "Fluctuating lattice Boltzmann," *Europhys. Lett.*, vol. 71, no. 3, pp. 473–479, Aug. 2005.

[158] P. Español, "Hydrodynamics from dissipative particle dynamics," *Phys. Rev. E*, vol. 52, no. 2, pp. 1734–1742, Aug. 1995.

[159] R. D. Groot and P. B. Warren, "Dissipative particle dynamics: Bridging the gap between atomistic and mesoscopic simulation," *J. Chem. Phys.*, vol. 107, no. 11, pp. 4423–4435, Sep. 1997.

[160] D. L. Ermak and J. A. McCammon, "Brownian dynamics with hydrodynamic interactions," *J. Chem. Phys.*, vol. 69, no. 4, pp. 1352–1360, Aug. 1978.

[161] J. Brady, "Stokesian Dynamics," *Annu. Rev. Fluid Mech.*, vol. 20, no. 1, pp. 111–157, 1988.

[162] R. R. Schmidt, J. G. H. Cifre, and J. G. de la Torre, "Comparison of Brownian dynamics algorithms with hydrodynamic interaction," *J. Chem. Phys.*, vol. 135, no. 8, p. 084116, Aug. 2011.

[163] J. Rotne and S. Prager, "Variational Treatment of Hydrodynamic Interaction in Polymers," *J. Chem. Phys.*, vol. 50, no. 11, pp. 4831–4837, Jun. 1969.

[164] H. Yamakawa, "Transport Properties of Polymer Chains in Dilute Solution: Hydrodynamic Interaction," *J. Chem. Phys.*, vol. 53, no. 1, pp. 436–443, Jul. 1970.

[165] E. Wajnryb, K. A. Mizerski, P. J. Zuk, and P. Szymczak, "Generalization of the Rotne-Prager-Yamakawa mobility and shear disturbance tensors," vol. 731, 2013.

[166] W. Guan *et al.*, "RPYFMM: Parallel adaptive fast multipole method for Rotne–Prager–Yamakawa tensor in biomolecular hydrodynamics simulations," *Comput. Phys. Commun.*, vol. 227, pp. 99–108, 2018.

[167] P. J. Atzberger, "Stochastic Eulerian Lagrangian methods for fluid–structure interactions with thermal fluctuations," *J. Comput. Phys.*, vol. 230, no. 8, pp. 2821–2837, Apr. 2011.

[168] Y. Wang, J. K. Sigurdsson, and P. J. Atzberger, "Fluctuating Hydrodynamics Methods for Dynamic Coarse-Grained Implicit-Solvent Simulations in LAMMPS," *SIAM J. Sci. Comput.*, vol. 38, no. 5, pp. S62–S77, Jan. 2016.

[169] M. Müller and G. D. Smith, "Phase separation in binary mixtures containing polymers: A quantitative comparison of single-chain-in-mean-field simulations and computer simulations of the corresponding multichain systems," *J. Polym. Sci. Part B Polym. Phys.*, vol. 43, no. 8, pp. 934–958, Apr. 2005.

[170] A. De Nicola, Y. Zhao, T. Kawakatsu, D. Roccatano, and G. Milano, "Validation of a hybrid MD-SCF coarse-grained model for DPPC in non-

lamellar phases," *Theor. Chem. Acc.*, vol. 131, no. 3, p. 1167, Mar. 2012.

[171] G. Milano, T. Kawakatsu, and A. De Nicola, "A hybrid particle–field molecular dynamics approach: a route toward efficient coarse-grained models for biomembranes," *Phys. Biol.*, vol. 10, no. 4, p. 045007, Aug. 2013.

[172] K. Mussawisade, M. Ripoll, R. G. Winkler, and G. Gompper, "Dynamics of polymers in a particle-based mesoscopic solvent," *J. Chem. Phys.*, vol. 123, no. 14, p. 144905, Oct. 2005.

[173] M. Ripoll, K. Mussawisade, R. G. Winkler, and G. Gompper, "Low-Reynolds-number hydrodynamics of complex fluids by multi-particle-collision dynamics," *Europhys. Lett.*, vol. 68, no. October, pp. 106–112, 2007.

[174] L. Cannavacciuolo, R. G. Winkler, and G. Gompper, "Mesoscale simulations of polymer dynamics in microchannel flows," *EPL (Europhysics Lett.*, vol. 83, no. 3, p. 34007, Aug. 2008.

[175] Y. Yang, V. Marceau, and G. Gompper, "Swarm behavior of self-propelled rods and swimming flagella," *Phys. Rev. E*, vol. 82, no. 3, p. 031904, Sep. 2010.

[176] J. Elgeti, U. B. Kaupp, and G. Gompper, "Hydrodynamics of Sperm Cells Near Surfaces," *Biophys. J.*, vol. 100, no. 9, pp. 2321–2324, May 2011.

[177] H. Noguchi and G. Gompper, "Dynamics of fluid vesicles in shear flow: Effect of membrane viscosity and thermal fluctuations," *Phys. Rev. E - Stat. Nonlinear, Soft Matter Phys.*, vol. 72, no. 1, pp. 1–14, 2005.

[178] H. Noguchi and G. Gompper, "Shape transitions of fluid vesicles and red blood cells in capillary flows," *Proc. Natl. Acad. Sci.*, vol. 102, no. 40, pp. 14159–14164, Oct. 2005.

[179] H. Noguchi and G. Gompper, "Dynamics of vesicle self-assembly and dissolution," *J. Chem. Phys.*, vol. 125, no. 16, p. 164908, Oct. 2006.

[180] M. J. Huang, R. Kapral, A. S. Mikhailov, and H. Y. Chen, "Coarse-grain model for lipid bilayer self-assembly and dynamics: Multiparticle collision description of the solvent," *J. Chem. Phys.*, vol. 137, no. 5, 2012.

[181] M. L. Henle, R. McGorty, A. B. Schofield, A. D. Dinsmore, and A. J. Levine, "The effect of curvature and topology on membrane hydrodynamics," *EPL (Europhysics Lett.*, vol. 84, no. 4, p. 48001, Nov. 2008.

[182] J. K. Sigurdsson and P. J. Atzberger, "Hydrodynamic coupling of particle inclusions embedded in curved lipid bilayer membranes," *Soft Matter*, vol. 12, no. 32, pp. 6685–6707, 2016.

[183] D. E. Shaw *et al.*, "Millisecond-scale molecular dynamics simulations on Anton," *Proc. Conf. High Perform. Comput. Networking, Storage Anal. - SC '09*, no. c, p. 1, 2009.

[184] J. D. Weeks, D. Chandler, and H. C. Andersen, "Role of Repulsive Forces in Determining the Equilibrium Structure of Simple Liquids," *J. Chem. Phys.*, vol. 54, no. 12, pp. 5237–5247, Jun. 1971.

[185] A. Malevanets and R. Kapral, "Solute molecular dynamics in a mesoscale solvent," *J. Chem. Phys.*, vol. 112, no. 16, pp. 7260–7269, Apr. 2000.

[186] S. H. Lee and R. Kapral, "Friction and diffusion of a Brownian particle in a mesoscopic solvent," *J. Chem. Phys.*, vol. 121, no. 22, pp. 11163–11169, 2004.

[187] A. Malevanets and J. M. Yeomans, "Dynamics of short polymer chains in solution," *Europhys. Lett.*, vol. 52, no. 2, pp. 231–237, Oct. 2000.

[188] M. Ripoll, K. Mussawisade, R. G. Winkler, and G. Gompper, "Dynamic regimes of fluids simulated by multiparticle-collision dynamics," *Phys. Rev. E - Stat. Nonlinear, Soft Matter Phys.*, vol. 72, no. 1, pp. 1–14, 2005.

[189] J. K. Whitmer and E. Luijten, "Fluid–solid boundary conditions for multiparticle collision dynamics," *J. Phys. Condens. Matter*, vol. 22, no. 10, p. 104106, 2010.

[190] B. Hess, "Determining the shear viscosity of model liquids from molecular dynamics simulations," *J. Chem. Phys.*, vol. 116, no. 1, pp. 209–217, 2002.

[191] B. J. Palmer, "Transverse-current autocorrelation-function calculations of the shear viscosity for molecular liquids," *Phys. Rev. E*, vol. 49, no. 1, pp. 359–366, 1994.

[192] W. K. den Otter and S. A. Shkulipa, "Intermonolayer Friction and Surface Shear Viscosity of Lipid Bilayer Membranes," *Biophys. J.*, vol. 93, no. 2, pp. 423–433, 2007.

[193] B. Hess, C. Kutzner, D. van der Spoel, and E. Lindahl, "GROMACS 4: algorithms for highly efficient, load balanced, and scalable molecular simulations," *J. Chem. Theory Comput.*, vol. 4, pp. 435–447, 2008.

[194] K. J. Bowers, R. O. Dror, and D. E. Shaw, “Overview of neutral territory methods for the parallel evaluation of pairwise particle interactions,” *J. Phys. Conf. Ser.*, vol. 16, pp. 300–304, 2005.

[195] E. A. Evans and R. M. Hochmuth, “Membrane viscoelasticity,” *Biophys. J.*, vol. 16, no. 1, pp. 1–11, Jan. 1976.

[196] R. E. Waugh, “Surface viscosity measurements from large bilayer vesicle tether formation. I. Analysis,” *Biophys. J.*, vol. 38, no. 1, pp. 19–27, Apr. 1982.

[197] T. T. Hormel, S. Q. Kurihara, M. K. Brennan, M. C. Wozniak, and R. Parthasarathy, “Measuring lipid membrane viscosity using rotational and translational probe diffusion,” *Phys. Rev. Lett.*, vol. 112, no. 18, pp. 1–5, 2014.

[198] Y. Wu *et al.*, “Molecular rheometry: direct determination of viscosity in Lo and Ld lipid phases via fluorescence lifetime imaging,” *Phys. Chem. Chem. Phys.*, vol. 15, no. 36, p. 14986, 2013.

[199] J. T. Mika *et al.*, “Measuring the Viscosity of the Escherichia coli Plasma Membrane Using Molecular Rotors,” *Biophys. J.*, vol. 111, no. 7, pp. 1528–1540, Oct. 2016.

[200] S. A. Shkulipa, W. K. den Otter, and W. J. Briels, “Surface viscosity, diffusion, and intermonolayer friction: simulating sheared amphiphilic bilayers,” *Biophys. J.*, vol. 89, no. 2, pp. 823–829, 2005.

[201] S. A. Shkulipa, W. K. Den Otter, and W. J. Briels, “Thermal undulations of lipid bilayers relax by intermonolayer friction at submicrometer length scales,” *Phys. Rev. Lett.*, vol. 96, no. 17, pp. 1–4, 2006.

[202] A.-F. Bitbol, J.-B. Fournier, M. I. Angelova, and N. Puff, “Dynamical membrane curvature instability controlled by intermonolayer friction,” *J. Phys. Condens. Matter*, vol. 23, no. 28, p. 284102, 2011.

[203] M. Fuhrmans, B. P. Sanders, S.-J. Marrink, and A. H. de Vries, “Effects of bundling on the properties of the SPC water model,” *Theor. Chem. Acc.*, vol. 125, no. 3–6, pp. 335–344, Mar. 2010.

[204] R. M. Venable, E. Hatcher, O. Guvench, A. D. MacKerell, and R. W. Pastor, “Comparing simulated and experimental translation and rotation constants: Range of validity for viscosity scaling,” *J. Phys. Chem. B*, vol. 114, no. 39, pp. 12501–12507, 2010.

[205] A. W. Lees and S. F. Edwards, “The computer study of transport processes

under extreme conditions," *J. Phys. C Solid State Phys.*, vol. 5, no. 15, pp. 1921–1928, Aug. 1972.

[206] D. J. Evans and O. P. Morriss, "Non-Newtonian molecular dynamics," *Comput. Phys. Reports*, vol. 1, no. 6, pp. 297–343, 1984.

[207] W. G. Hoover, C. G. Hoover, and J. Petravic, "Simulation of two- and three-dimensional dense-fluid shear flows via nonequilibrium molecular dynamics: Comparison of time-and-space-averaged stresses from homogeneous Doll's and Sllod shear algorithms with those from boundary-driven shear," *Phys. Rev. E - Stat. Nonlinear, Soft Matter Phys.*, vol. 78, no. 4, pp. 1–14, 2008.

[208] R. Zwanzig, "Time-Correlation Functions and Transport Coefficients in Statistical Mechanics," *Annu. Rev. Phys. Chem.*, vol. 16, no. 1, pp. 67–102, Oct. 1965.

[209] S. R. Durell, B. R. Brooks, and A. Ben-Naim, "Solvent-Induced Forces between Two Hydrophilic Groups," *J. Phys. Chem.*, vol. 98, no. 8, pp. 2198–2202, Feb. 1994.

[210] P. Mark and L. Nilsson, "Structure and Dynamics of the TIP3P, SPC, and SPC/E Water Models at 298 K," *J. Phys. Chem. A*, vol. 105, no. 43, pp. 9954–9960, Nov. 2001.

[211] S. J. Marrink, A. H. de Vries, and A. E. Mark, "Coarse Grained Model for Semiquantitative Lipid Simulations," *J. Phys. Chem. B*, vol. 108, no. 2, pp. 750–760, 2004.

[212] N. Kučerka *et al.*, "Lipid Bilayer Structure Determined by the Simultaneous Analysis of Neutron and X-Ray Scattering Data," *Biophys. J.*, vol. 95, no. 5, pp. 2356–2367, Sep. 2008.

[213] E. Falck, M. Patra, M. Karttunen, M. T. Hyvönen, and I. Vattulainen, "Lessons of Slicing Membranes: Interplay of Packing, Free Area, and Lateral Diffusion in Phospholipid/Cholesterol Bilayers," *Biophys. J.*, vol. 87, no. 2, pp. 1076–1091, Aug. 2004.

[214] S. Reißer, D. Poger, M. Stroet, and A. E. Mark, "Real Cost of Speed: The Effect of a Time-Saving Multiple-Time-Stepping Algorithm on the Accuracy of Molecular Dynamics Simulations," *J. Chem. Theory Comput.*, vol. 13, no. 6, pp. 2367–2372, 2017.

[215] R. M. Venable, F. L. H. Brown, and R. W. Pastor, "Mechanical properties of lipid bilayers from molecular dynamics simulation," *Chem. Phys. Lipids*, vol.

192, pp. 60–74, 2015.

[216] R. M. Venable *et al.*, “CHARMM All-Atom Additive Force Field for Sphingomyelin: Elucidation of Hydrogen Bonding and of Positive Curvature,” *Biophys. J.*, vol. 107, no. 1, pp. 134–145, Jul. 2014.

[217] A. Filippov, G. Orädd, and G. Lindblom, “The effect of cholesterol on the lateral diffusion of phospholipids in oriented bilayers,” *Biophys. J.*, vol. 84, no. 5, pp. 3079–3086, 2003.

[218] N. Kučerka, M.-P. Nieh, and J. Katsaras, “Fluid phase lipid areas and bilayer thicknesses of commonly used phosphatidylcholines as a function of temperature,” *Biochim. Biophys. Acta - Biomembr.*, vol. 1808, no. 11, pp. 2761–2771, Nov. 2011.

[219] P. R. Maulik, P. K. Sripada, and G. G. Shipley, “Structure and thermotropic properties of hydrated N-stearoyl sphingomyelin bilayer membranes,” *BBA - Biomembr.*, vol. 1062, no. 2, pp. 211–219, 1991.

Appendix

PERMISSIONS

Certain figures and passages throughout the text, mostly found in Chapters 4 and 5, are adapted from the journal article “Toward Hydrodynamics with Solvent Free Lipid Models: STRD Martini” published by the author in the Biophysical Journal [1].