# GUIDING SELF-ASSEMBLY OF FUNCTIONALIZED NANOPARTICLES BY COMPUTATIONAL MODELING OF EFFECTIVE INTERACTIONS

A Thesis
Submitted to the Graduate Faculty
of the
North Dakota State University
of Agriculture and Applied Science

By

Vijay Dipak Shah

In Partial Fulfillment of the Requirements
for the Degree of
MASTER OF SCIENCE

Major Department:
Physics

March 2018

Fargo, North Dakota

# NORTH DAKOTA STATE UNIVERSITY

Graduate School

**Title**

GUIDING SELF-ASSEMBLY OF FUNCTIONALIZED NANOPARTICLES BY COMPUTATIONAL MODELING OF EFFECTIVE INTERACTIONS

**By**

Vijay Dipak Shah

The supervisory committee certifies that this thesis complies with North Dakota State University's regulations and meets the accepted standards for the degree of

MASTER OF SCIENCE

SUPERVISORY COMMITTEE:

Prof. Alan Denton
Chair

Prof. Erik Hobbie

Prof. Alexander Wagner

Prof. Dmitri Kilin

Approved:

| | |
|---|---|
| April 11, 2018 | Prof. Sylvio May |
| Date | Department Chair |

# ABSTRACT

Nanoparticles have attracted much attention because of their unusual physical properties, which allow them to be used in many practical applications. The self-assembly of nanocrystals into crystalline arrays can be facilitated by functionalizing the nanocrystals with ligand brushes, allowing for bulk dispersions to be sterically stabilized against aggregation. Studies have been conducted to study the clustering of gold nanoparticle dispersions. To study the self-assembly of gold nanoparticle dispersions based on nanocrystal volume fraction and ligand coverage, we performed Monte Carlo simulations and characterized the ability of the nanoparticle dispersions to self-assemble into crystalline arrays. Experiments have shown that silver nanoparticles can self-assemble into equilibrium superlattices in the presence of free ligands. To better understand the role of adsorbed and free ligands in self-assembly, we extracted the effective pressure between two flat, ligated plates through molecular dynamics simulations. Our results are compared to the theoretical prediction and discrepancies are discussed.

# ACKNOWLEDGEMENTS

First off, a huge thank you to Professor Alan Denton for everything these past 4 years. He has been an amazing academic and research adviser and got me interested in the field of condensed matter physics. I had no programming experience when I started conducting research back in October 2014, but because of his guidance, I was able to learn Java and about the physics of soft materials. Since then, my long-term career goal has become much clearer to me, and I plan to continue conducting research because of my experiences as a part of his research group. I hope that one day, I can become a professor and inspire students as much as he has.

Thank you to Samuel Brown and Professor Erik Hobbie for helping to guide our research and collaborating with our group. The input and guidance that they gave through the experiments that they conducted have helped to motivate the research presented in this thesis and given direction on which values to use for different parameters.

Thank you to all of my current and past professors for their guidance and unyielding patience during the courses that they instructed, and to my friends and family for their support. The professors at North Dakota State University have given me a sound background that allowed me to apply what I have learned in my research experiences and teach others about physics. As a student in a STEM field, having a great group of peers to study with and explain physics to has helped me grow immensely. In addition, my family has supported every opportunity that I have taken and helped me become a successful person.

A big thank you to Professor Erik Hobbie, Professor Alexander Wagner, and Professor Dmitri Kilin for being willing to be a part of the committee for my Master's Thesis, as well.

Finally, I would like to thank the National Science Foundation for supporting this research (Grant No. CBET-1603445) and the North Dakota State University Center for Computationally Assisted Science and Technology (CCAST) for the computing resources.

# TABLE OF CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

Figure Page

# LIST OF ABBREVIATIONS

| | |
|---|---|
| NP | Nanoparticle |
| NC | Nanocrystal |
| MC | Monte Carlo |
| MD | Molecular dynamics |
| Si | Silicon |
| Ag | Silver |
| Au | Gold |
| vdW | van der Waals |
| PL | Photoluminescence |
| CdSe | Cadmium selenide |
| CdS | Cadmium sulfide |
| SAM | Self-assembled monolayer |
| FCC | Face-centered cubic |
| LJ | Lennard-Jones |
| LAMMPS | Large-scale Atomic/Molecular Massively Parallel Simulator |

# LIST OF SYMBOLS

| | |
|---|---|
| $r$ | Center-center distance between two spheres |
| $a$ | Nanocrystal radius |
| $d$ | Contour length of ligands |
| $D$ | Separation distance of two flat plates |
| $v$ | Ligand coverage (ligands per unit area per surface) |
| $u(r)$ | Potential energy as a function of $r$ for molecular dynamics |
| $v_{\mathrm{comp}}(r)$ | The potential energy due to “comp” interactions (Monte Carlo) |
| $\chi$ | Flory $\chi$ parameter |
| $k_B$ | The Boltzmann constant |
| $T$ | Temperature |
| $\epsilon$ | LJ energy parameter |
| $\sigma$ | LJ length parameter |
| $r_c$ | LJ potential cutoff distance |
| $\Delta G$ | Change in Gibbs free energy |
| $\rho$ | Density |
| $i$ | Number of beads per ligand chain |
| $V_1, v_m, v_s$ | Molecular volume of solvent molecule (differ in units) |
| $V_S, v_{\mathrm{ligand}}$ | Molecular volume of a ligand |
| $\delta_s, \delta_m$ | Hildebrand solubility parameters |
| $\phi$ | Nanocrystal volume fraction |
| $\phi_{\mathrm{av}}$ | Ligand volume fraction in shell of radius $d$ around an NC |
| $H$ | Hamaker constant |
| $N$ | Number of nanoparticles |
| $V$ | Volume of system |
| $N_{\mathrm{ligand}}$ | Number of ligands per nanoparticle |
| $A$ | Area |
| $F$ | Force |

| | |
|---|---|
| $P$ | Pressure |
| $n$ | Free ligand concentration |
| $L$ | Thickness of exclusion region of free ligands |
| $g(r)$ | Radial distribution function |
| $S(q)$ | Static structure factor |
| $\phi_{\text{dep}}$ | Free ligand volume fraction |

# 1. INTRODUCTION

Soft condensed matter appears everywhere in our lives. Items such as shampoo, soap, and paints as well as foods such as ketchup are considered to be neither crystalline arrays nor simple liquids [1]. The classification of “soft condensed matter” encompasses these items and foods. Scientists study the physics of these materials, which are composed of macromolecules such as colloids and nanoparticles, due to the very interesting properties that they exhibit. Surprisingly, one does not need quantum mechanics to describe the physics of macromolecules, which have sizes between those of atoms and macroscopic objects; instead, classical statistical mechanics becomes very important due to the fluctuations that occur in soft matter [1]. Materials that are classified as soft matter also have large response functions: a small change to the system leads to vastly different behavior in the system from the state before the change.

Nanoparticles (NPs) are particles that are smaller than macroscopic objects and cells (cells are on the order of $10^{-6}$ m) and larger than electrons and nuclei (which are smaller than $10^{-12}$ m), and are therefore classified as macromolecules [2]. As the name suggests, NPs have sizes on the order of $10^{-9}$ m, also called the “nanometer” and denoted “nm” [3]. In the literature [4], an NP is often defined as an inorganic core (in other words, the core is not made up of hydrogen, carbon, phosphorus, nitrogen, oxygen, or sulfur atoms) along with a surrounding layer of ligands or agents. The inorganic core is often made up of atoms arranged in a lattice, though they need not form a crystal structure [4]. In the work presented here, however, we will consider an inorganic crystalline core, called a nanocrystal (NC). The NCs can be coated with a layer of ligands which are either grafted (chemically bonded to the NC so that one end is fixed to the surface unless the bond is broken) or adsorbed (the ligands can detach and re-attach to the surface depending on the amount of energy they have). The ligand layers stabilize NP dispersions against aggregation due to attraction between the NCs, induced by van der Waals (vdW) interactions [5, 6, 7, 8, 9], and they allow the NP dispersions to self-assemble into crystalline arrays (superlattices) due to the interactions between the ligands and the solvent molecules [7, 8, 10]. Nanoparticles consisting of an NC and a layer of ligands are referred to as “functionalized nanoparticles”.

Nanoparticles have many unusual optical, magnetic, and electronic properties between those of molecular structures and bulk structures [5, 11, 12, 13, 14, 15, 16, 17]. In addition, NPs have been used in many practical applications [1, 2, 3, 15, 16, 17, 18, 19, 20, 21]. These physical properties and practical applications will be discussed in more detail in the following chapter.

There have been many experimental and computational studies of gold nanoparticle (AuNP) dispersions [10, 22, 23]. The computational study given in Khan et al. [10] focuses heavily on the clustering of AuNPs (see Figure 1.1), while experimental studies have focused on both the clustering and the self-assembly of AuNPs into equilibrium superlattices. Khan et al. utilized an effective pair potential derived in a series of papers by Evans, Napper, and Smitham [24, 25, 26, 27] to model the interactions between nanoparticles. The series of papers gave a new approach to describe the interactions between two ligated surfaces. First, Evans et al. [24] derived the effective interactions for two flat, ligated plates and they then extended the interactions to two curved surfaces. Smitham et al. [25] made the effective interactions more concrete by stepping through the mathematical steps for particular ligand density distributions. Finally, Evans et al. [27] incorporated an entropic component to the effective interactions between two ligated surfaces and worked through the mathematical steps for particular distributions.

Although it has been shown that the effective pair potential describing interactions between silicon nanoparticles (SiNPs) consists of vdW interactions that depend on the size and density of the ligands and NC materials [5], it is still a challenge to get NPs to self-assemble into a superlattice in the laboratory. Experiments must be run for long periods of time and are often costly in both money and materials needed to synthesize the NPs, and the cause of self-assembly in the laboratory is not fully understood. For example, nanoparticle dispersions of silver nanocrystals coated with oleylamine ligands have been shown in experiments by Professor Erik Hobbie and Samuel Brown to self-assemble into equilibrium superlattices in the presence of free ligands (ligands that are present in the solvent rather than attached to the surface of an NC), as shown in Figure 1.2. However, the interplay between the repulsive steric (short-range) interactions and attractive depletion interactions caused by the presence of the free ligands is not well understood. The equilibrium structures can change drastically with small changes to parameters such as the volume fraction of the NCs and the length and density of the ligands, and free ligands can increase the strength of depletion interactions, causing the NP dispersions to self-assemble into stable superlattices when

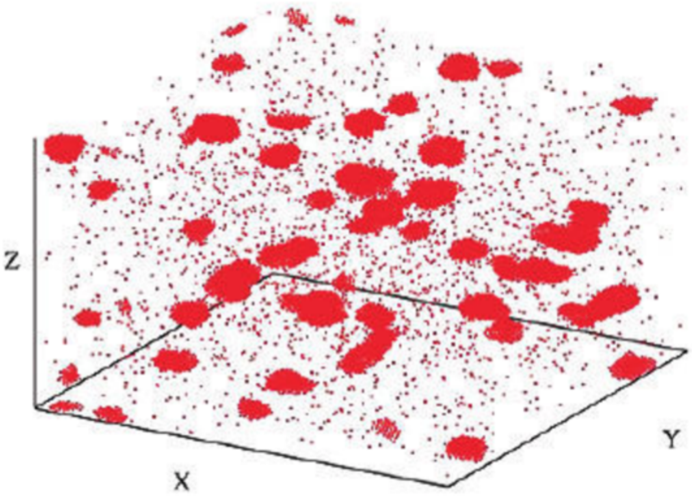

Figure 1.1. A snapshot of a gold nanoparticle dispersion from Brownian dynamics simulations where the volume fraction of nanoparticles was $\phi = 0.005$ [10]. Reprinted with permission from S. J. Khan, F. Pierce, C. M. Sorenson, and A. Chakrabarti. Self-Assembly of Ligated Gold Nanoparticles: Phenomenological Modeling and Computer Simulations. *Langmuir*, 25(24):13861–13868, 2009. Copyright 2009 American Chemical Society.

they would not have done so in the absence of free ligands. Because of these challenges and uncertainties, we have developed computational methods to characterize the structures of self-assembling nanoparticle dispersions based on changes in system parameters, as well as computational methods to determine the effective pressure (force per unit area) between two nanoparticles.

By running Monte Carlo simulations of gold nanoparticle dispersions and silver nanoparticle dispersions and determining the radial distribution functions and static structure factors, we characterized the ability of the nanoparticles to self-assemble into equilibrium superlattices and witnessed the clustering of nanoparticles at low volume fractions. By running molecular dynamics simulations, we calculated the pressures from ligand-ligand and ligand-surface interactions between two flat plates and compared our results with the theory presented in Evans et al. [24, 27] and Smitham et al. [25]. Our molecular dynamics methods will allow for a better description of the self-assembly of AgNP dispersions. These methods could also be extended to model SiNP dispersions and could potentially be used to raise the efficiency of photovoltaic cells due to the increased understanding of nanoparticle self-assembly and therefore increased guidance in the fabrication of equilibrium superlattices. In particular, our methods for calculating the effective pressure between

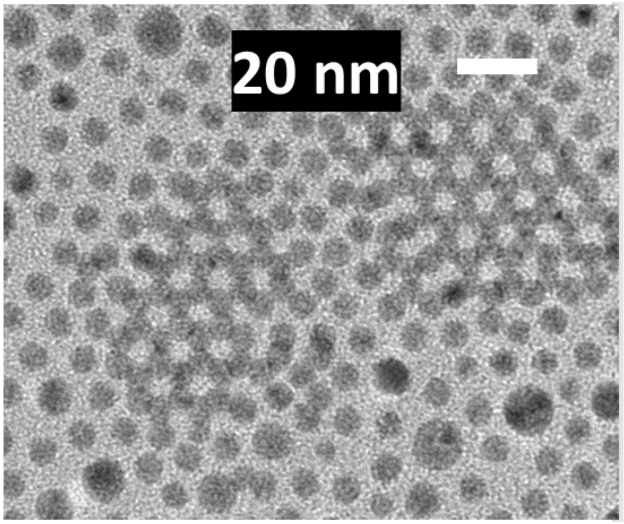

Figure 1.2. A TEM image of a silver nanoparticle dispersion in a stable equilibrium superlattice. Credit to Samuel Brown and Professor Erik Hobbie.

two ligated surfaces could be extended to two spherical surfaces in the future, allowing for the validation of the effective pair potential used in our model for nanoparticle dispersions.

The rest of the thesis is structured as follows. We first summarize work that has been conducted in the field to give the reader a sound understanding of nanoparticles and the self-assembly of nanoparticle dispersions. Next, we describe the models of our systems, including the nanoparticle model and the components for our model to determine the effective pressure between two flat plates. We then describe the components of the effective interparticle pair potential used to extend work found in the literature [10] and compare with results found in experiments. Next, we describe the computational methods used to simulate the nanoparticle dispersions, along with the structural properties of the nanoparticle dispersions that were used to analyze the equilibrium structures. In addition, we present our computational methods to extract the effective pressure between two ligated surfaces. We then present the results that we obtained using Monte Carlo methods to characterize the equilibrium structures of gold nanoparticle dispersions and silver nanoparticle dispersions and the results that we obtained using molecular dynamics methods to extract the effective interactions between two flat, ligated plates. Finally, we summarize the work presented in this thesis and the outlook for the future. The Appendix contains scripts and Java classes that were created and used for the presented research.

# 2. BACKGROUND

Scientists have released several thousands of publications on “nanoparticles” ever since the term first appeared in the literature [17]. Even before the term “nanoparticle” first appeared, much work was done to determine interactions between particles of different geometries and different functionalization properties such as grafting density. The goal of this chapter is to inform the reader of the work that has been completed to study NPs and their interactions with one another, as well as their practical applications. However, we are unable to cover the entire field and instead focus on the topics relevant for the work described in this thesis.

## 2.1. Physical Properties of Nanoparticles

Nanoparticles have a relatively high surface-to-volume ratio, which is something that is commonly seen for macromolecules [1, 17, 28]. In the case of spherical nanoparticles, this is because the radius of an NP is on the nanometer scale and because there are a very large number of atoms making up the surface of the nanoparticle compared to the number of atoms making up its interior. Because the sizes of nanoparticles are intermediate between molecular and bulk scales, nanoparticles have many unusual properties, which has led to several studies in an attempt to characterize these properties. These properties are size-dependent at the nanometer scale [29], and it is important to characterize which properties could be used for practical applications and how they vary with size.

Nanoparticles have many unusual optical properties. Cadmium selenide (CdSe) superlattices have been shown to exhibit photoluminescence (PL) [11]. This was done by synthesizing CdSe NCs of a desired size before using the experimental setup in Figure 2.1 to measure PL. Silicon nanoparticles have also been shown to exhibit PL [14]. Brown et al. [14] used molecular dynamics simulations, density functional theory, and experimental techniques to show that the PL spectrum for SiNCs is dependent on temperature and nanocrystal size. Finally, smaller clusters of gold nanoparticles (AuNPs) show an increase in the intensity of PL. [29].

Nanoparticles made from certain transition metals, such as gold and silver, also exhibit strong surface plasmon absorption [29]. Surface plasmon absorption occurs when there is resonance between the incident electromagnetic wave and the coherent electron motion of the NCs [29]. In other words, the frequencies of oscillation of the electron motion and the electromagnetic wave are

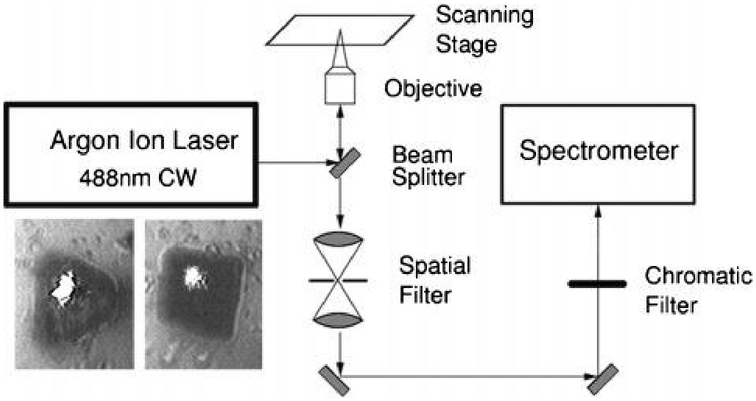

Figure 2.1. An experimental setup used to measure PL of cadmium selenide nanocrystals [11]. Reprinted with permission from N. Zaitseva, Z. R. Dai, F. R. Leon, and D. Krol. Optical properties of cdse superlattices. *J. Am. Chem. Soc.*, 127:10221–10226, 2005. Copyright 2005 American Chemical Society.

very close to each other. A net charge difference arises at the nanoparticle surface due to polarization of the electrons by incoming light, causing dipolar oscillations [29]. It has been found that the color exhibited actually depends on the shape of the NCs: for spherical gold nanoparticles, the solution appeared to be red; and for triangular-branched gold nanoparticles, the solution appeared to be blue [29]. The size of the NCs also affects the exhibited color: experimentally, the absorption spectra shift such that the peak occurs at larger wavelengths (red shifting) and over a larger range of wavelengths as the NC size increases [29].

The shape of the nanoparticle affects the electronic properties as well: plasmon bands can split into degenerate energy levels (different bands yielding the same energy) if the NCs are asymmetric [29]. Finally, while metallic NCs conduct electricity well, functionalizing the NCs with organic ligand chains such as thiol groups hinders the conductivity of the nanoparticles because the chains themselves are poor electrical conductors [30], which could potentially be an issue for practical applications; however, Kovalenko et al. [30] describes a type of functionalization which reduces this hinderance.

CdSe nanoparticles and nanorods have been the focus of studies for several years. Indeed, many publications aside from Zaitseva et al. [11] have studied properties of CdSe nanomaterials. Cadmium sulfide (CdS) has also been of interest, and the structure of chains that functionalize CdSe and CdS nanorods has been studied [31]. For CdSe, increasing the ligand chain length corresponds to an increase in the ordering tempertature, which is the temperature at which the orientation of the ligand chains switches from ordered (pointing in the same direction) to disordered (pointing in random directions). Using molecular dynamics simulations, Widmer-Cooper et al. [31] considered ligand-solvent interactions using a unified-atom potential and the ligand-ligand and ligand-particle interactions using the Lennard-Jones potential (see Section 3.1). By varying the ligand coverage, dimensions of the nanorods, and temperature of the system, the authors characterized the order of the ligand chains and determined whether this would cause interactions between the nanorods to be repulsive or attractive. Due to the toxicity of materials such as cadmium selenide to humans, however, more recent studies have turned towards less toxic materials such as silicon and alloys of different metals [32]. Silicon has low toxicity even compared to metallic nanoparticles and is abundant [5, 12, 14]. Practical applications of nanoparticles will be described in more detail in Section 2.3, but it should be noted that silicon nanoparticles are used in many applications because of their low toxicity and abundance.

## 2.2. Interparticle Interactions and Self-Assembly

It has been shown that van der Waals (vdW) interactions play a large role in the aggregation of small materials such as nanoparticles [1, 5, 6, 9, 33]. The vdW interactions consist partially of dispersion forces (a classification of weak forces which act between all atoms and molecules) [33], which arise from considering instantaneous dipoles in a nonpolar atom. In particular, an attractive force arises between the nonpolar atom and another nearby atom due to the interaction between the instantaneous dipoles in both atoms [33].

Nanoparticles under the influence of only vdW interactions would aggregate because they would always feel attraction toward each other, if they are composed of the same material. However, nanoparticle dispersions can be stabilized into superlattices by functionalizing the particles with ligand brushes. These ligand brushes are often made of one molecule, such as *n*-decane, dodecanethiol ($C_{12}H_{26}S$) [10, 22], or oleylamine [21]. In addition, the ligand brushes can be either grafted or adsorbed to the surface of an NC. "Adsorbed" means that the ligands are physically or chemically attached to the surface of the NC because they do not have enough energy to detach from the surface. Grafted ligands are often considered to be irreversibly bonded due to the strength of the chemical bonds [1], whereas adsorbed ligands can detach or reattach to the surfaces: the amount of energy needed to physically detach from a surface is much less than the amount of energy needed to break a chemical bond. In computational methods, the adsorption and desorption of ligands are often modeled by considering probabilities and certain criteria according to distribution functions (this concept is the basis of Monte Carlo simulations, and will be elaborated upon more in Section 3.3.1).

The interactions governing the structure and the self-assembly of nanoparticle dispersions are studied by immersing the dispersions in a solvent (often toluene or hexane). The term “self-assembly” refers to the assembly of the nanoparticle dispersions without external fields or forces. In other words, the NPs assemble into a stable structure only through the interactions between themselves (hence the “self” in “self-assembly”). The work described in Ouhenia-Ouadahi et al. [34] studies the self-assembly of silver nanoparticles with dodecanethiol or decanethiol ligands in hexane by considering the removal of a coating agent from the solution. Ouhenia-Ouadahi et al. [34] found that the superlattice was larger in size in toluene than in hexane, though the same shape was observed in both solutions. In addition, increasing the chain length in hexane caused the superlattices to shrink in size.

Much work has been done to study the structure of self–assembled monolayers (SAMs) on gold nanoparticles [35]. The term “self–assembled monolayers” should not be confused with “self–assembly”: in the context of our work, SAMs is a term that refers to ordered ligand chains formed by the adsorption of one end of the ligand chains with a surface [35]. Similar to “self–assembly”, the adsorption must occur spontaneously. Thiol groups, such as decanethiol and dodecanethiol, are considered to be SAMs. The phases and orientation of decanethiol on gold substrates have been studied by varying temperature and ligand coverage of the surface, and it was found that for low temperatures (below 100$^o$C), the thiol groups were oriented parallel to the substrate when the ligand coverage was low and perpendicular (and completely straight) when the ligand coverage was high [35]. These results make sense because the ligands would have more room to fully extend if they are oriented perpendicular to the substrate at high ligand coverage. Because of the concrete characterization of thiol groups on gold surfaces, gold nanoparticles with functionalized thiol groups are often used to study the self-assembly of metallic nanoparticle dispersions.

Depletion interactions have also been characterized for many systems of nanoparticles [1, 23, 36, 37, 38, 39, 40, 41]. These interactions arise only if there are free ligands or polymer chains in solution with larger colloidal particles. As NPs get close together, free ligands in solution are excluded from the region between the NPs because they are unable to fit in the gap between the surfaces of the two NPs. This creates a difference in osmotic pressure which causes the two NPs to move closer together [1, 42]. Many studies have been done to study the depletion interactions between different shapes of nanocrystals, such as cubes and octahedra [38] and between two spherical nanoparticles and two nanorods. We will outline some of this work below.

As stated in the previous section, Cd-based semiconducting nanoparticles and nanorods have been a very popular choice used to study the properties of nanoparticles. Zanella et al. [41] describes a study in which the authors fabricated nanorod superlattices by considering Cd-based nanorods and copper-based nanorods with oleylamine and oleic acid chains. By using magnetic properties of the system, the effects of depletion interactions could be characterized in the self-assembly of superlattices [41]. It was found that copper-based rods clustered when in a solution with Cd-based nanorods [41]. In addition, the authors considered tetrapods, and observed that these tetrapods formed structures with only short-range order [41]. Another study [40] better characterized the depletion interactions between two spheres by considering the depletants to be thin, hard rods. The authors' goal was accomplished by using second-order perturbation theory, in which a small change is introduced into a system for which the solution is analytically solveable. The authors noted that a repulsive barrier arose in the second-order approximation, though their results are only formally valid in the regime of low depletant concentration [40].

In addition, Lau et al. [23] characterized how depletion interactions from excess ligand in solution affected the self-assembly of gold nanoparticles with adsorbed oleylamine ligands in hexane and toluene. The authors observed disk-like droplets in contrast to the expected large superlattices [23]. Due to the weak bonds between animes and gold nanoparticles [23], the ligands desorbed from the surface of the gold nanoparticles in toluene, causing aggregation of the nanoparticles due to the destabilization of the nanoparticle dispersions as the amount of ligand on the surface of the gold nanocrystals decreased and the vdW and depletion interactions began to dominate. An interesting observation from the study described in Lau et al. [23] is that if the solvent was evaporated, then excess oleylamine ligands in the solution could actually recrystallize disordered gold nanoparticles. This signifies that the gold nanoparticles could form metastable superlattices due to depletion interactions, making these interactions an important concept in the field of nanoparticles, especially when the ligand-solvent interactions cause repulsion between nanoparticles, as in the case of oleylamine and toluene (discussed in Section 3.2.2).

Wang and Denton [43] determined the effective pair potential governing the interactions between star–branched polyelectrolytes using Monte Carlo simulations and density functional theory. Wang and Denton [43] considered the polyelectrolytes to consist of a fixed core with rigid rod-like arms that were allowed to freely rotate around the core. Charged monomers were placed on these rod-like arms, and counterions in the solution caused the screening of electrostatic interactions between the charged monomers (modeled using the Yukawa potential, which describes attraction/repulsion between charged particles but also accounts for the screening of interactions by including an exponentially decaying factor) and the reduction of the effective charge (valence number) [43]. By considering two cores at a fixed separation distance (varied between simulations but not during simulations), the effective pair potential between polyelectrolytes was calculated using the various methods described in the publication and compared to the theory. It was found that considering a high valence number yielded agreement between theory and the simulations for all separation distances of the polyelectrolytes [43].

The effective forces between two flat, ligated plates have been studied theoretically [26], computationally [44], and experimentally [45, 46, 47, 48]. Evans et al. [26] evaluates previous theories to describe steric stabilization of surfaces coated with polymer chains and describes the shortcomings of these theories, setting the stage for an extension of one of the most accurate solvency theories at the time. For example, some theories fail to take into account (either implicitly or explicitly) the solvent that the surfaces and chains are immersed in [26], even though the solvent can play an extremely important role in the stability and equilibrium structures of nanoparticle dispersions, especially when one is interested in the dynamics of the nanoparticle dispersions [8, 49]. Yethiraj et al. [44] describes Monte Carlo simulations where the volume of the system, temperature, and number of molecules are held constant, which allowed for the calculation of segment density profiles and the effective forces between two flat plates. The chain segments were modeled as hard spheres rather than bead-springs, meaning that two segments could not experience overlap. In this work, the pores on the flat plates in which the fixed chain segments resided in were described and the polymers were treated as a confined fluid [44].

Experimentally, surface force apparatus (SFA) and atomic force microscopy (AFM) are often the methods of choice to determine the effective forces between two ligated surfaces. By using AFM, one can determine both the polymer configurations and the force profiles of the system simultaneously [45]. O'Shea et al. [45] studied block copolymers chemically bonded to a mica substrate and considered three different types of solvent. The authors found that the forces obtained using AFM were of a lower order of magnitude than those obtained from an SFA, though the two force profiles were qualitatively the same in the case of a "good solvent" (in which it is more energetically favorable for the ligands to interact with the solvent molecules than with other ligands): both methods yielded exponential repulsion between the plates [45]. In other words, the force decreased exponentially as the separation distance between the two plates was increased. The authors also found that a change in the ligand coverage caused a change in the force profiles, as expected: as the ligands got closer to one another, the ligands interacted more with each other (attraction if the substrate and ligands were immersed in a "poor solvent" or repulsion if the substrate and ligands were immersed in a "good solvent"). This contributed to a larger force if the ligands on one substrate were closer to those on another substrate, therefore causing a change in the force profiles. Finally, the authors described how the SFA force profiles matched with the Alexander-de Gennes expression, which describes the pressure $P$ on a substrate as a function of the substrate separation distance $D$ and the mean distance $s$ between fixed segments that are chemically bonded to the substrate in the regime where $D < 2d$, where $d$ is the brush thickness (i.e., interpenetration and slight compression of the ligand layers of the two substrates occurs) [45]:

$$P(D) = \frac{k_B T}{s^3}\left[\left(\frac{2d}{D}\right)^{9/4} - \left(\frac{D}{2d}\right)^{3/4}\right]. \tag{2.1}$$

Bridging interactions can cause a net attraction to arise between polymer chains and surfaces [46]. Bridging occurs when two surfaces are close enough together that a polymer which is adsorbed on one of the two surfaces can adsorb onto the second surface. If the ligand coverage of the surfaces is too high, however, then there is a lower probability of bridging due to the lack of adsorption sites on the second surface [46]. Goodman et al. experimentally studied the interactions between spherical polymer brushes and a silicon nitrate tip in aqueous media in order to characterize the brush properties that contributed to bridging [46]. At short separation distances, they observed that steric and electrostatic interactions were dominant and that the magnitude of the steric interactions decreased as the ligand coverage of the surfaces decreased, as expected. This decrease caused a potential well of considerable depth to form at larger separation distances, but as the two surfaces approached contact, steric repulsion still dominated due to the overcrowding of the ligands.

Yamamoto et al. [47, 48] considered silicon substrates covered by end-grafted poly(methyl methacrylate) (PMMA, for short) chains in toluene. By varying the chain length and coverage of the substrates and using AFM to analyze the results, the authors determined the equilibrium length $L_e$, or the separation at which repulsion between the two substrates was first detectable, to describe how the interactions between substrates varied. It was observed that [48]

$$L_e \propto dv^{1/3}, \tag{2.2}$$

where $d$ is the contour (fully extended) length of the ligands and $v$ is the ligand coverage of the substrates. At higher ligand coverage, the separation between graft points of the ligands decreases. Since toluene was a "good solvent" for PMMA, the ligands extended away from the surface to minimize the ligand-ligand interactions, therefore increasing the equilibrium length [47, 48]. From this analysis, it follows that the ligands would be more difficult to compress at high ligand coverage, and therefore if the ligand contour length increased, then it would be more difficult for the two substrates to approach contact [47]. This can be seen in Equation (2.2) because the repulsion between two surfaces would be detectable at a larger separation distance if $d$ increased.

Marla and Meredith [50, 51, 52] have computed the effective force between two fixed spherical nanoparticles by running Monte Carlo simulations in the grand canonical ensemble (fixed chemical potential, system volume, and temperature) using the expanded ensemble method. The authors calculated the chemical potential after several Monte Carlo steps by considering the insertion/deletion of polymer chains and keeping the number of chain segments, volume of the system, and temperature fixed. Once the system had reached chemical equilibrium, the forces and segment densities were calculated. The authors considered three cases: one spherical nanoparticle, in which they introduced their model [50]; two spherical nanoparticles where the ligand chains were adsorbed to the surfaces of the nanocrystals [51]; and two spherical nanoparticles where the ligand chains were grafted to the NC surfaces [52]. Though the authors did not consider an explicit solvent or concretely describe the materials that their simulations represented, their work provided a characterization of the effective forces between two spherical nanoparticles and helped to motivate our model of two ligated surfaces.

### 2.3. Practical Applications

The many different materials used for NPs allow them to be utilized in a plethora of practical applications, such as for renewable energy sources, commercial products, and in the medical industry. Semiconductors are of large interest in the solar cell industry [53]. Silicon, an extremely well-known semiconductor, is one particular material of interest due to the properties of SiNPs described in Section 2.1 [14]. It is well-known that solar energy is not very efficient compared to other energy sources, despite being a promising renewable energy source due to the cost (a high cost of installation but a low cost for regular maintenance) and ability to store energy for use at a later time. Coating a photovoltaic cell with a thin film composed of SiNP superlattices could help to raise the efficiency of solar cells, allowing for a more sustainable renewable energy source [41]. In order to be useful, however, the superlattices must be stable so that the thin film does not break apart and cause the photovoltaic cell to lose the ability to function as an energy storage device.

Semiconducting materials are not the only materials with practical applications. Metallic nanoparticle dispersions are much easier to stabilize than most semiconducting nanoparticle dispersions due to the large dispersion forces between metallic materials [1, 33]. Indeed, van der Waals and electrostatic forces can be very important in the stabilization of NP dispersions, allowing for their use in many practical applications such as in food technology and paints [17, 54]. In particular, silver nanoparticles are important in many applications such as filtering the air or disinfecting water because AgNPs can slow the growth of bacteria and fungi [15]. Given that less than 1% of the water on Earth is safe to drink [15], the ability for AgNPs to be used as a water disinfectant is very important. AgNPs are not only used to inhibit microbial strain growth: they have also been used in paints and creams, and for food preservation [15]. Certain metals can be toxic to humans and to the environment, so while metallic nanoparticles are relatively easy to stabilize, semiconducting materials are often used in practical applications whenever possible in order to maximize the safety of consumers.

Finally, nanoparticles have many applications to drug delivery [19, 28]. Nanoparticles can be used to carry drugs to a part of the body and release the drug at a certain time, causing the drug to act only when it has reached a specific part of the body. It is therefore important to understand the toxicity of different materials: without a clear understanding of how a nanoparticle may harm humans, it cannot be used for drug delivery [28].

# 3. MODEL AND METHODS

In this chapter, we describe the models of our system and the interactions between NPs, as well as the methods that we used to simulate the NP dispersions. After we describe our model and methods, we introduce the radial distribution function and static structure factor, which are two functions that allowed us to determine the structure of the NP dispersions after each simulation reached completion.

## 3.1. Nanoparticle Model

We modeled interactions between NPs, which consisted of inorganic cores of radius $a$ with ligands of length $d$ attached to the surface of the nanocrystals, as a function of the center-center distance $r$ between the NPs. The NPs were immersed in toluene. The NPs were assumed to be spherical in shape using a coarse-grained model (we consider the overall shape of the nanoparticle and differentiate between materials using parameters instead of considering the atoms that make up the NC).

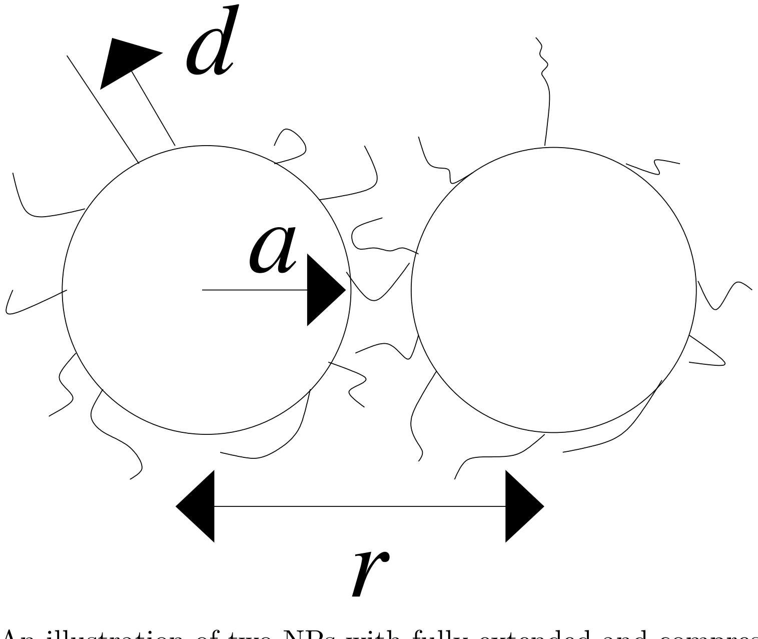

Figure 3.1. An illustration of two NPs with fully extended and compressed ligands.

Figure 3.1 shows an illustration of the NPs used in our research. As shown in the illustration, the ligand layers of two NPs can overlap (they interpenetrate). If two NCs are extremely close to each other, the ligands between the two NCs also undergo compression. The role of the interpenetration and compression of the ligand layers will be described more in Section 3.2. The ligands and solvent were implicit in our Monte Carlo simulations, and different materials were taken into account by varying parameters in the effective interparticle pair potential. We first considered the case where the ligands were modeled by considering their contribution to the effective pair potential without explicitly modeling them (Section 3.3.1). However, before describing the effective pair potential this case, we will first describe our model for our molecular dynamics simulations in which we computed the effective pressure between ligated surfaces.

### 3.1.1. Model: Explicit Ligands

We later modeled the ligands on a more molecular scale than our initial model by considering a bead-spring model (Section 3.3.2). In other words, we considered the ligands to consist of several beads, which were bonded together and could stretch, compress, and rotate via harmonic potentials [55]:

$$E_H(s) = K_1(s - s_0)^2 \tag{3.1}$$

and

$$E_H(\theta) = K_2(\theta - \theta_0)^2. \tag{3.2}$$

In Equation (3.1), $s$ is the separation distance of two beads, $K_1$ is a constant determining how "stiff" the bond between the beads is, and $s_0$ is the equilibrium separation between two beads. $K_1$ is related to the spring constant $k$ in the well-known equation for the potential energy of a spring $E_p = (1/2)k\Delta x^2$. Indeed, any specified value for $K_1$ takes into account the missing factor of 1/2 ($K_1 = k/2$). Similarly in Equation (3.2) where $\theta$ is the bond angle, $K_2$ is a constant that determines how rigid the overall chain is (analogous to $k$), and $\theta_0$ is the equilibrium angle between two bonds.

The subscripts of the prefactors in Equations (3.1) and (3.2) are included to signify that although $K_1$ and $K_2$ both determine how rigid the chain is, they are actually two different values. When the bond length is equal to $s_0$ and the angle between two bonds is $\theta_0$, the harmonic energy is at a minimum value. Figure 3.2 shows an example of two flat plates coated with ligand chains, which were implemented using the bead-spring model described above. Note that so far we have only considered the case where the ligands are grafted to the two flat plates. In addition, the solvent was still implicitly modeled.

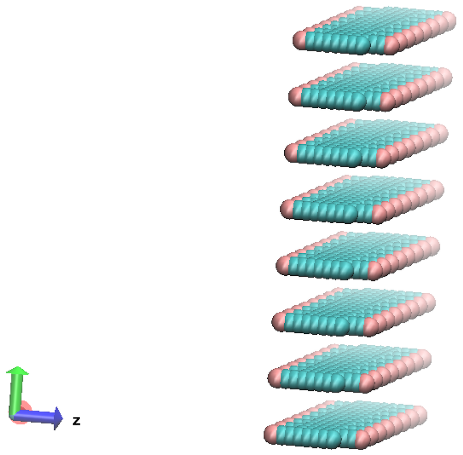

Figure 3.2. Two flat plates (both parallel to the $xy$-plane) coated with ligand chains modeled using the bead-spring model, where the pink beads are fixed beads on the flat plates and the green beads are able to move. The two plates can be close together by initializing the ligands as folded chains. Alternatively, the ligand chains are initialized as fully extended chains if the desired separation distance between the plates is large enough.

To determine the effective pressure between two ligated surfaces, we model the interactions between surfaces by considering the truncated Lennard-Jones (LJ) 12-6 (referring to the powers of the $\sigma/r$ terms) potential between segments of the chains (also called "beads of the ligands") that are attached to the surfaces, given by [33, 50, 51, 52]

$$u_{\mathrm{LJ,s-s}}(r) = \begin{cases} 4\epsilon \left[ \left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6} - \left(\frac{\sigma}{r_c}\right)^{12} + \left(\frac{\sigma}{r_c}\right)^{6} \right] & \text{if } r < r_c \\ 0 \quad \text{if } r > r_c, \end{cases} \tag{3.3}$$

where $\epsilon$ is the LJ energy parameter (representing the depth of the potential well in Equation (3.3)), $\sigma$ is the LJ length parameter chosen such that the potential energy is at a minimum at $r = 2^{1/6}\sigma$, and $r_c$ is the distance at which the pair potential is cut off. Graphically, $r_c$ is the value of $r$ such that $u(r) = 0$ if two particles are a distance $r > r_c$ from each other. In other words, there is no repulsion and no attraction between two particles that are farther than $r_c$ apart from each other. At $r = 2.5\sigma$, the value of the full LJ 12-6 potential is very small compared to $\epsilon$, and it is more computationally efficient to cut off the potential, as one only sacrifices a negligible amount of accuracy by doing so. If we consider the full potential, the last two terms for the first case in Equation (3.3) vanish because $r_c \to \infty$. While there is no strict requirement to set $r_c > 2^{1/6}\sigma$, we only considered $r_c \geq 2.5\sigma$ in this thesis in order to model the attractive interactions between beads. Figure 3.3 summarizes the interpretation of the parameters from Equation (3.3) by considering a very common set of values for $\epsilon$, $\sigma$, and $r_c$.

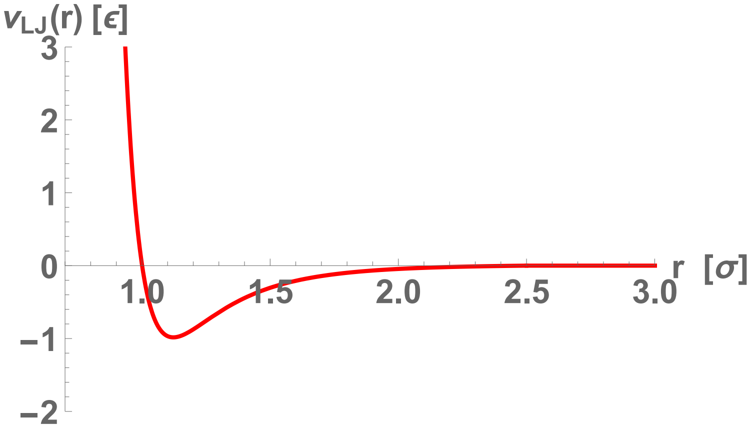

Figure 3.3. The potential energy from Equation (3.3) with $\epsilon = 1$, $\sigma = 1$, and $r_c = 2.5\sigma$.

Note that segments do not interact with their neighbors through the LJ potential, but rather only through the harmonic bonds and angles, as given in Equations (3.1) and (3.2). However, the excluded chain volume effects between bonded beads are still taken into account because the harmonic potentials yield both attractive and repulsive forces between beads. Along with the bead-bead interactions, we also considered the bead-wall interactions. The expression for these interactions differs depending on whether we consider flat plates or spherical surfaces, but both are modeled using a type of LJ potential. In the case of two flat plates, the bead-wall interactions are given by the LJ 9-3 potential (below), which can be obtained by integrating over a spherical surface:

$$u_{\mathrm{s-w}}(r) = \begin{cases} \epsilon_w \left[ \frac{2}{15} \left(\frac{\sigma_w}{r}\right)^9 - \left(\frac{\sigma_w}{r}\right)^3 \right] & \text{if } r < r_{\mathrm{c,w}} \\ 0 \quad \text{if } r > r_{\mathrm{c,w}}. \end{cases} \tag{3.4}$$

The parameters $\epsilon_w$, $\sigma_w$, and $r_{\mathrm{c,w}}$ need not be the same as those defined in Equation (3.3), so we added the subscript $w$ to make the distinction clear. Note, however, that Equation (3.4) is not continuous at $r = r_{\mathrm{c,w}}$: this is because there is currently not a way for us to specify a shift in the bead-wall interactions. The minimum of the full LJ 9-3 potential is located at $r = (2/5)^{1/6}\sigma_w$. Currently, the bead-wall potential is excluded, and we instead consider reflecting walls that represent a "hard wall" potential (in which the velocity of a bead is reversed if the center of the bead overlaps with or goes past the wall) in an attempt to reconstruct the effective interactions presented in Evans et al. [24, 27] and Smitham et al. [25].

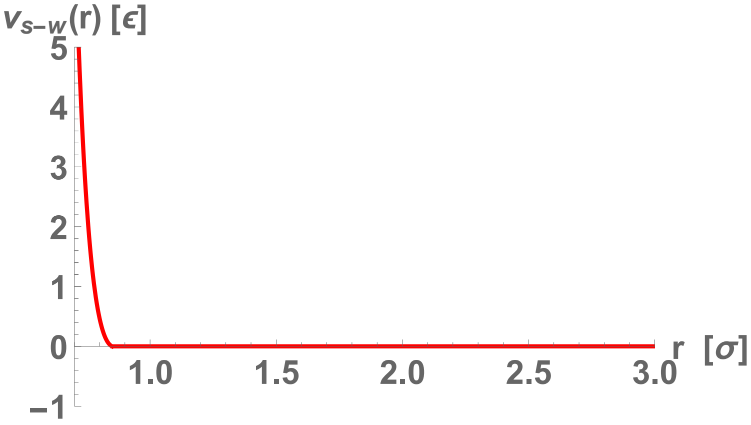

Figure 3.4. The potential energy from Equation (3.4) (shifted to ensure continuity at the cutoff) with $\epsilon_w = 5$, $\sigma_w = 1$, and $r_{\mathrm{c,w}} = 0.85\sigma_w$.

It is important to note that the derivative of Equation (3.3) with respect to $r$ (the negative of the force between two beads) is actually discontinuous: the shift in the effective pair potential given by the final two terms vanishes during differentiation, so differentiating the shifted pair potential yields the same result as differentiating the pair potential before the shift. Therefore, we also considered another modification to the LJ 12-6 pair potential between beads, which was defined in order to ensure continuity in both the effective pair potential and the force when two beads were a distance $r_c$ away from each other:

$$u_{\mathrm{s-s,shifted}}(r) = u_{\mathrm{LJ,s-s}}(r) - (r - r_c)\left.\frac{du_{\mathrm{LJ,s-s}}}{dr}\right|_{r=r_c}, \tag{3.5}$$

where $u_{\mathrm{LJ}}$ refers to the unshifted LJ potential from Equation (3.3).

Figure 3.5 shows a plot for the pair potential given in Equation (3.5). One problem with using Equation (3.5) to model the bead-bead interactions is that the bead-wall interactions from Equation (3.4) (both the pair potential and the force) are not shifted and therefore not continuous. If the pair potential is discontinuous, a bead that is distance $r_{\mathrm{c,w}}$ away from the wall will experience an impulse, which is not consistent with the bead-bead interactions if the force at $r = r_c$ is defined to be 0 $\epsilon/\sigma$. Therefore, we extend the cutoff distances of all of the pair potentials, rather than shifting the pair potentials to ensure that the forces are continuous at the cutoff.

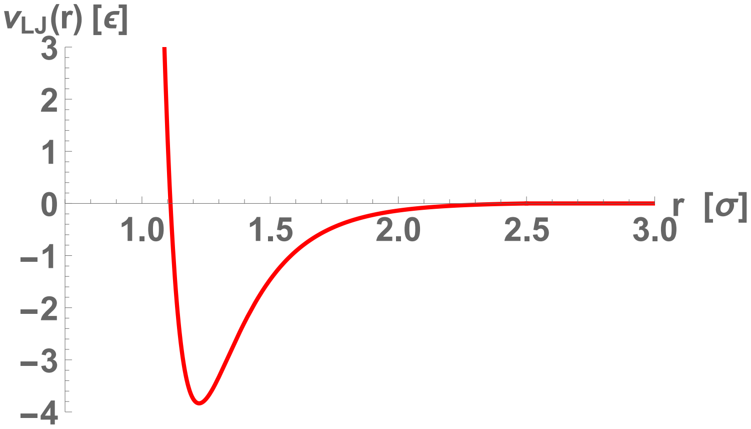

Figure 3.5. The potential energy from Equation (3.5) with $\epsilon = 1$, $\sigma = 1$, and $r_c = 2.5\sigma$. Note that the depth of the potential well is no longer equal to $\epsilon$, due to the last term in Equation (3.5).

While we have not completed simulations for the case of spherical surfaces coated with ligands using the bead-spring model, we succeeded in generating the model for two spherical surfaces. The ligand chains were initialized using a Python script (instead of using a C++ script like the case of two flat plates due to the available functions in Python), and the bead-surface interactions would be different than they are in the case of two flat plates.

### 3.2. Effective Interparticle Pair Potential

The Large-scale Atomic/Molecular Massively Parallel Simulator (LAMMPS) [55] contains many pair potentials which are already coded and ready for scientists to use in their research, such as the LJ potential. We will describe LAMMPS in more detail in Section 3.3. We note here that in order to validate our Monte Carlo methods and consider effective pair potentials that more accurately describe the interactions between NPs, we first considered an effective pair potential

$v_{\text{eff}}(r)$ of the following form:

$$v_{\text{eff}}(r) = v_{\text{vdW}}(r) + v_{\text{steric}}(r), \tag{3.6}$$

where $v_{\text{vdW}}(r)$ represents the van der Waals (vdW) interactions [1, 33] between two NPs and $v_{\text{steric}}(r)$ represents the short-range repulsion between the NPs, given by the truncated Lennard-Jones (LJ) 12-6 pair potential from Equation (3.3). The vdW interactions were described in Chapter 2. Since the validation of our Monte Carlo methods was not a part of the research conducted for this thesis, we will not describe the details here.

### 3.2.1. Theoretical Effective Interactions between Flat Plates

After we validated our Monte Carlo methods, we considered an effective interparticle pair potential between two spheres of equal size, given to be

$$v_{\text{eff}}(r) = v_{\text{vdW}}(r) + v_{\text{mix}}(r) + v_{\text{el}}(r) + v_{\text{dep}}(r),$$

where $v_{\text{vdW}}(r)$ represents the van der Waals interactions, $v_{\text{mix}}(r)$ represents the free energy of mixing, $v_{\text{el}}(r)$ represents the elastic potential, and $v_{\text{dep}}(r)$ represents the depletion interactions (Chapter 2). Before we can present the equations for each term in our effective pair potential between two spheres, we must first consider the case of two flat plates. The derivation of the effective pair potential between two flat plates will allow us to then derive the effective pair potential between two spheres. In addition, we compared the results obtained using molecular dynamics methods with the theoretical effective pair potential between two flat plates. There are many approximations that are made and the derivations given in the literature [24, 25, 26, 27] for $v_{\text{mix}}(r)$ and $v_{\text{el}}(r)$ rely on topics from several other papers. While it is not feasible for us to include every small detail or all of the background in the derivations, we will outline the important points and steps.

The free energy of mixing arises from a change in the Gibbs free energy, whereas the elastic potential arises out of a change in configurational entropy of the ligands [10, 24, 27]. The Gibbs free energy $\Delta G$ is a quantity that allows us to determine whether or not a process can happen spontaneously: if $\Delta G < 0$ for a process, then the system has the ability to arrive in the final state of the process spontaneously. Entropy is a quantity that measures how many possible configurations are available for a system, though it is often referred to as the amount of "disorder" of a system. As mentioned in Chapter 2, many theories [26] attempted to describe steric stabilization of molecules in a solvent; however, it was shown that nearly all of theories were inconsistent in one or more ways with experiments [26]. The goal of Evans et al. [24] was to extend an already existing theory and more accurately describe the steric stabilization due to the interactions between the solvent and chains attached to a surface in the regimes where two surfaces are approaching contact and the chains undergo compression. For the derivations, we will draw freely from References [24, 25, 27, 56].

Consider an infinitesimal volume element $\delta V$ inside a solution containing ligated colloids and solvent. The change in the total Gibbs free energy ($\Delta G$) inside this volume element is then

$$\delta(\Delta G) = \delta(\Delta G_M) + \delta(\Delta G_{\mathrm{SP}}), \tag{3.7}$$

where $\Delta G_M$ denotes the change in the Gibbs free energy due to the mixing of chain segments and solvent inside the volume element and $\Delta G_{\mathrm{SP}}$ is the change in the Gibbs free energy due to all other mixing considerations. By applying Flory-Krigbaum Theory [57], Equation (3.7) can be rewritten as

$$\delta(\Delta G) = k_B T[\ln(1 - v_2) + \chi v_2]\delta n_1, \tag{3.8}$$

where $v_2$ is the volume fraction of the chains, $\chi$ is the Flory chi interaction parameter which describes the solvent quality [10, 22, 58], and $\delta n_1$ is the number of solvent molecules in the volume element, given by

$$\delta n_1 = \frac{1 - \rho_2 V_S}{V_1} \delta V, \tag{3.9}$$

where $V_S$ is the volume of a chain segment, $\rho_2$ is the number density of the segments, and $V_1$ is the volume of a solvent molecule.

Substitution of Equation (3.9) into Equation (3.8) implies that the change in total Gibbs free energy is then

$$\delta(\Delta G) = k_B T \left( \frac{1 - \rho_2 V_S}{V_1} \right) \delta V[\ln(1 - v_2) + \chi v_2]. \tag{3.10}$$

We now need to represent the logarithmic term as a Taylor series. A Taylor series of a function $f(x)$ is an infinite sum of polynomials that can be used to estimate $f(x)$ around some value $x = a$. In general, a Taylor series has the form

$$f(x)|_{x=a} = f(a) + \frac{f'(a)}{1!}(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \frac{f'''(a)}{3!}(x-a)^3 + \cdots, \tag{3.11}$$

where the numerators of the fractions denote the derivatives of $f$ with respect to $x$ evaluated at $x = a$. By Taylor series expansion around $v_2 = 0$, the logarithmic term in Equation (3.10) becomes

$$\ln(1-v_2) = \ln(1) - \frac{v_2}{1-v_2} - \frac{v_2^2}{2(1-v_2)^2} + O(v_2^3). \tag{3.12}$$

Because $v_2 << 1$ and ignoring all terms of order $v_2^3$ and higher, Equation (3.12) becomes

$$\ln(1-v_2) \approx -v_2 - \frac{v_2^2}{2}. \tag{3.13}$$

Therefore, we have

$$\begin{aligned}\delta(\Delta G) &\approx k_BT\left(\frac{1-\rho_2V_S}{V_1}\right)\delta V[-v_2 - \frac{v_2^2}{2} + \chi v_2] \\ &= k_BT\frac{\rho_2V_Sv_2}{V_1}\delta V - k_BT\frac{v_2}{V_1}\delta V + k_BT\frac{\rho_2V_Sv_2^2}{2V_1}\delta V - k_BT\frac{v_2^2}{2V_1}\delta V \\ &+ k_BT\frac{\chi v_2}{V_1}\delta V - k_BT\frac{\rho_2V_s\chi v_2}{V_1}\delta S.\end{aligned} \tag{3.14}$$

Note, however, that $\rho_2V_S = v_2$: by definition of the number density and the volume of a segment, multiplication yields the volume of polymer chains over the total volume, therefore giving us the volume fraction of the polymer chains. Making this substitution into Equation (3.14) and ignoring terms of order $(\rho_2V_S)^3$ and higher,

$$\begin{aligned}\delta(\Delta G) &\approx \frac{k_BT\rho_2^2V_S^2}{V_1}\delta V - \frac{k_BT\rho_2V_S}{V_1}\delta V - \frac{k_BT\rho_2^2V_S^2}{2V_1}\delta V \\ &+ \frac{k_BT\chi\rho_2V_S}{V_1}\delta V - \frac{k_BT\chi\rho_2^2V_S^2}{V_1}\delta V \\ &= k_BT\frac{\rho_2^2V_S^2}{V_1}\left(\frac{1}{2}-\chi\right)\delta V - 2\left(k_BT\frac{\rho_2V_S}{2V_1}\delta V\right) + k_BT\chi\frac{\rho_2V_S}{V_1}\delta V.\end{aligned} \tag{3.15}$$

Simplification yields the result we are looking for:

$$\delta(\Delta G) \approx k_BT\frac{\rho_2^2V_S^2}{V_1}\left(\frac{1}{2}-\chi\right)\delta V - k_BT\frac{\rho_2V_S}{V_1}\left(\frac{1}{2}-\chi\right)\delta V - \frac{1}{2}k_BT\frac{\rho_2V_S}{V_1}\delta V. \tag{3.16}$$

Equation (3.16) applies to polymer chains that are attached to colloids. Let us assume that our colloids are two flat plates. If the chains are irreversibly attached to the plates, and if the segment density $\rho_2$ is assumed to be continuous and constant normal to the surface, we can derive a theoretical prediction that describes the interactions between two flat plates. We will denote $v$ as the number of chains per unit area normal to the surface (also called “ligand coverage”), and $i$ as the number of segments in one polymer chain. Integration of Equation (3.16) over the distance away from the flat plates $x$ yields

$$\frac{\int_0^\infty \delta(\Delta G)dx}{k_BT} = \frac{\Delta G_\infty}{k_BT} = \frac{V_S^2}{V_1}\left(\frac{1}{2}-\chi\right)\int_0^\infty \rho_\infty^2 dV - \frac{V_S}{V_1}\left(\frac{1}{2}-\chi\right)\int_0^\infty \rho_\infty dV - \frac{V_S}{2V_1}\int_0^\infty \rho_\infty dV, \tag{3.17}$$

where the subscript $x=\infty$ refers to the two flat plates being separated by an infinite distance.

It is convenient to normalize the segment density $\rho_2 = \rho_x$ such that $\int \hat{\rho} dx = 1$ [56]. Because $\rho$ is a function of $x$ and is independent of the $y-$ and $z-$coordinates (without loss of generality, $x$ is the direction normal to the plates), $dV = A\ dx$ [24], and it is more convenient to consider one unit of area. Then we have that $\int_0^\infty \rho_\infty dx = vi$. Using this relation and the definition of normalization, Equation (3.17) becomes (now free energy per unit area)

$$\frac{\Delta G_\infty}{k_BT} = \frac{V_S^2}{V_1}\left(\frac{1}{2}-\chi\right)v^2i^2\int_0^\infty \hat{\rho}_\infty^2 dx - \frac{V_S}{V_1}vi\left[\left(\frac{1}{2}-\chi\right)+\frac{1}{2}\right]. \tag{3.18}$$

The wall separation was unspecified until we integrated the infinitesimal change in Gibbs free energy in Equation (3.17), so it follows that when the walls are separated by a finite distance $D$,

$$\frac{\Delta G_D}{k_BT} = \frac{V_S^2}{V_1}\left(\frac{1}{2}-\chi\right)v^2i^2\int_0^D \hat{\rho}_D^2 dx - \frac{V_S}{V_1}vi\left[\left(\frac{1}{2}-\chi\right)+\frac{1}{2}\right]. \tag{3.19}$$

However, Equations (3.18) and (3.19) are for one plate alone, so in order to determine the change in the *total* Gibbs free energy when two plates are interacting, we have that [57]

$$\frac{\Delta G}{k_BT} = \frac{\Delta G_C + \Delta G_I}{k_BT} = 2\left(\frac{\Delta G_D - \Delta G_\infty}{k_BT}\right) + 2\frac{V_S^2}{V_1}\left(\frac{1}{2}-\chi\right)v^2i^2\int_0^D \hat{\rho}_D\hat{\rho}_D' dx, \tag{3.20}$$

where $\hat{\rho}'$ indicates a translation of $\hat{\rho}$ by $x = D$ and a rotation about the $yz-$plane (in other words, it is the segment density of the second flat plate), $\Delta G_C$ indicates the change in free energy due to compression of the ligands, and $\Delta G_I$ indicates the change in free energy due to the interpenetration of two ligand layers (one from each flat plate). Using Equations (3.18) and (3.19), we get

$$\frac{\Delta G}{k_B T} = \frac{2V_S^2}{V_1}\left(\frac{1}{2} - \chi\right) v^2 i^2 \left[\int_0^D \hat{\rho}_D^2 dx - \int_0^\infty \hat{\rho}_\infty^2 dx + \int_0^D \hat{\rho}_D \hat{\rho}'_D dx\right]. \tag{3.21}$$

Equation (3.21) is the equation for the change in Gibbs free energy per unit area between two flat plates due to mixing. We use this equation to determine the concrete expression for the free energy of mixing of ligands and solvent if the ligands are attached to two flat plates which are separated by the distance $D$. Note that the $\hat{\rho}'_x$ must be non-negative for any $x$.

In our system, the ligand chains have a finite length $d$, and when $2d < D$ it must be that the free energy of mixing is 0. When $d < D < 2d$, we assume there is no compression of the ligands. This implies that the first two integrals in Equation (3.21) are 0. Therefore,

$$\frac{\Delta G}{k_B T} = \frac{2V_S^2}{V_1}\left(\frac{1}{2} - \chi\right) v^2 i^2 \int_0^D \hat{\rho}_D \hat{\rho}'_D dx. \tag{3.22}$$

The quantity in Equation (3.22) is clearly positive if $\chi < 1/2$, 0 if $\chi = 1/2$, and negative if $\chi > 1/2$. When $\chi < 1/2$, this implies that the solvent in which the two flat plates with ligand chains is immersed is a “good solvent”. Recall that a “good solvent” is a solvent that the ligand chains would rather interact with over themselves, signifying that it is more energetically favorable for the ligands to be as far away from each other as possible and causing an increase in the Gibbs free energy due to mixing to produce a repulsive force between the walls. On the other hand, $\chi > 1/2$ means that the solvent is a “poor solvent”. This means that the ligands would rather interact with themselves than with the solvent, producing a negative change in the Gibbs free energy due to mixing and providing an attractive force between the walls to minimize the ligand-solvent interactions. If $\chi = 1/2$, the solvent is called a “$\theta$-solvent”. In a “$\theta$-solvent”, the ligands do not care if they interact with themselves or with the solvent and there is no change in the Gibbs free energy due to the mixing of ligands with solvent.

If $0 < D < d$, we cannot immediately set any of the integrals in Equation (3.21) to 0. However, the segment density $\hat{\rho}'_D$ must be greater than or equal to $\hat{\rho}'_\infty$, so by positivity of the integrals the free energy of mixing is still positive if we have a "good solvent", negative if we have a "poor solvent", and 0 (trivially) if we have a "$\theta$-solvent".

Before we move on to the calculation of $\Delta G$ for our particular work, we derive the general formula for $v_{\text{el}}$ between two flat plates [27]. As stated earlier in this section, $v_{\text{el}}$ arises from a loss of configurational entropy of the ligand chains because they are trying to fit into a space that is too small for them. Similarly to people in an overcrowded room, the ligands have very little space to move and therefore fewer possible configurations in this space. This generates a repulsive force between the two flat plates so that the ligands can regain a higher number of possible configurations. While $v_{\text{el}}$ realistically is non-zero in the regime where $d < D < 2d$ because there is still a possibility for the ligands to compress, it is set to zero because the segment densities are much lower than in the regime where $D < d$ and the free energy of mixing dominates the effective interactions [27]. Here we will only consider the regime where $D < d$ and draw freely on Evans et al. [27].

Given one flat plate coated with ligand chains, we use the Planck-Boltzmann relationship [27]

$$\Delta G_{\text{el}} = -k_B T \ln \frac{\Omega_f}{\Omega_i}, \tag{3.23}$$

where $\Omega_f$ and $\Omega_i$ stand for the number of accessible microstates when the ligands are compressed and uncompressed, respectively, given by

$$\ln \Omega_f = \sum_f v_f \ln \left( \frac{w_f v}{v_f} \right), \tag{3.24}$$

where $w_f$ denotes the probability of the ligand being perfectly straight and normal to the flat plate in the range $\Delta x$ given by $w_f = \hat{\rho}_{x_f} \Delta x$ and $v_f$ is the volume fraction of the chains that satisfies this condition. Clearly then, $v = \sum_f v_f$.

Plugging Equation (3.24) into Equation (3.23) and taking into account the presence of two flat plates, we have

$$\frac{\Delta G_{\text{el}}}{k_B T} = -2 \sum_f v_f \ln \left[ \frac{\hat{\rho}_{x_f} \delta \Delta x}{\hat{\rho}_{x_f/\delta} \Delta x} \right] = -2 \sum_f v_f \ln \left[ \frac{\delta \hat{\rho}_{x_f}}{\hat{\rho}_{x_i}} \right], \tag{3.25}$$

where $\delta \equiv x_f/x_i$. Note that $\Delta G_{\mathrm{el}}$ is always non-negative because the argument of the logarithmic term is never larger than 1.

Using Equations (3.21) and (3.25), we will now present the steps used to determine the theoretical effective interactions between two ligated flat plates [25]. Following the literature [10], we considered a uniform segment density, represented by

$$\rho_D(x) = \Theta(d - x). \tag{3.26}$$

The Heaviside function (also known as the step function) $\Theta(x)$ is defined such that when the argument of the function is less than 0, $\Theta = 0$, and when the argument is greater than 0, $\Theta = 1$. Put more formally, for any real number $z$,

$$\Theta(z) = \begin{cases} 0, & \text{if } z < 0 \\ 1, & \text{if } z > 0. \end{cases}$$

First, we evaluate Equation (3.21) to obtain the free energy of mixing, so consider the regime where $d < D < 2d$. We must normalize our segment density, and note that in this regime,

$$\int_0^D \rho_D(x)dx = \int_0^d dx = d.$$

Therefore, our normalized segment density is

$$\hat{\rho}_D = \hat{\rho}_\infty = \frac{1}{d}\Theta(d - x). \tag{3.27}$$

The integrals in Equation (3.21) then evaluate to

$$\begin{aligned} &\int_0^D \hat{\rho}_D^2 dx = \frac{1}{d^2}\int_0^d dx = \frac{1}{d}; \\ &\int_0^\infty \hat{\rho}_\infty^2 dx = \frac{1}{d}; \\ \text{and } &\int_0^D \hat{\rho}_D\hat{\rho}'_D dx = \frac{1}{d^2}\int_0^D \Theta(d-x)\Theta(x+d-D)dx = \frac{1}{d^2}\int_{D-d}^d dx = \frac{1}{d^2}(2d - D). \end{aligned} \tag{3.28}$$

As shown in Equation (3.22), the first two integrals in Equation (3.21) cancel by subtraction. Therefore,

$$\frac{\Delta G}{k_B T} = \frac{2V_S^2}{d^2 V_1}\left(\frac{1}{2} - \chi\right) v^2 i^2 (2d - D) = \frac{4V_S^2}{dV_1}\left(\frac{1}{2} - \chi\right) v^2 i^2 \left[1 - \left(\frac{D}{2d}\right)\right]. \tag{3.29}$$

Now, consider the regime where $D < d$. We then have to compute both the free energy of mixing from Equation (3.21) and the elastic energy from Equation (3.25). $\hat{\rho}'_\infty$ is still the same in this regime as for the previous regime, but we have to now (crudely) approxmiate $\hat{\rho}_D$. In order to ensure that the expressions for the segment densities match up at $D = d$, we take

$$\hat{\rho}_D = \frac{1}{D}. \tag{3.30}$$

Equation (3.30) is a simple approximation that clearly matches Equation (3.27) at $D = d$. We cannot use a more accurate formula for the segment density in this case because, to our knowledge, there is no exact expression to describe the segment density when the chains compress. One argument for why Equation (3.30) may be an invalid approximation to use is that the segment density is undefined at $D = 0$. However, if enough ligands are present (in other words, if the ligand coverage $v$ is high), the repulsive elastic potential term from Equation (3.25) will not allow the two flat walls to make contact at $D = 0$ due to the potential barrier that is created. Therefore, we believe that our approximation is still valid to use. To see that our approximation is normalized, note that as required,

$$\int_0^D \hat{\rho}_D dx = \int_0^D \frac{1}{D} dx = 1.$$

Equation (3.21) then becomes

$$\begin{aligned}\frac{\Delta G}{k_B T} &= \frac{2V_s^2}{V_1}\left(\frac{1}{2} - \chi\right) v^2 i^2 \left[\int_0^D \frac{1}{D^2} dx - \int_0^d \frac{1}{d^2} dx + \int_0^D \left(\frac{1}{D}\right)^2 dx\right] \\ &= \frac{2V_s^2}{V_1}\left(\frac{1}{2} - \chi\right) v^2 i^2 \left[\frac{2}{D} - \frac{1}{d}\right].\end{aligned} \tag{3.31}$$

Note that the free energy of mixing matches up at $D = d$ in Equations (3.29) and (3.31), as required. In addition, Equation (3.25) becomes

$$\frac{\Delta G_{\text{el}}}{k_B T} = -2 \sum_f v_f \ln \delta = -2v \ln \left( \frac{D}{d} \right). \tag{3.32}$$

Using Equations (3.29), (3.31), and (3.32), we have formulated an expectation for our molecular dynamics simulations where the ligand chains are modeled using the bead-spring model, and we can also derive the effective interactions between two spheres. The latter will be done in the next section.

Differentiating Equations (3.29), (3.31), and (3.32) with respect to $D$ to compute the pressure between two flat plates from the free energy of mixing and the elastic potential, we find that for $d < D < 2d$,

$$\frac{F_{\text{mix}}(D)}{A} = P_{\text{mix}}(D) = \frac{2V_S^2}{d^2 V_1} \left( \frac{1}{2} - \chi \right) v^2 i^2, \tag{3.33}$$

and for $D < d$,

$$P_{\text{mix}}(D) = \frac{4V_S^2}{V_1 D^2} \left( \frac{1}{2} - \chi \right) v^2 i^2 \tag{3.34}$$

and

$$P_{\text{el}}(D) = \frac{2v}{D}. \tag{3.35}$$

Table 3.1 shows the fixed parameters that we used for our effective interactions between two ligated flat plates. Note that each bead had a radius of $0.85\sigma$. Figure 3.6 shows the effective pressure (force per unit area) between two flat plates, as predicted from [24, 25, 27]. Note that the pressures are discontinuous when $D = d$: while we could introduce a shift in the derived effective pressures, we are unsure on to how to justify this shift physically. Note that in Figure 3.6(c) and (d) that the pressure becomes negative at low surface separations due to $\chi > 0.5$ (the "poor solvent" regime). This indicates that the two flat plates undergo attraction at low separations.

### 3.2.2. Theoretical Effective Interactions between Similar Spheres

By extending the model found in the literature [10, 24, 25], we considered an isotropic effective pair potential to model the interactions between NPs. The effective pair potential was

Table 3.1. A list of fixed parameters used to generate the plots in Figure 3.6 in Lennard-Jones units.

| System Parameter | Value |
|---|---|
| Bead radius | $0.85\sigma$ |
| Ligand coverage $v$ | $0.022\sigma^{-2}$ |
| Contour ligand length $d$ | $8\sigma$ |
| Number of segments per chains $i$ | 8 |
| Volume of solvent molecule $v_s$ | $0.84\sigma^3$ |
| Ligand molecular vol. $v_{\text{ligand}}$ | $2.572\sigma^3$ |

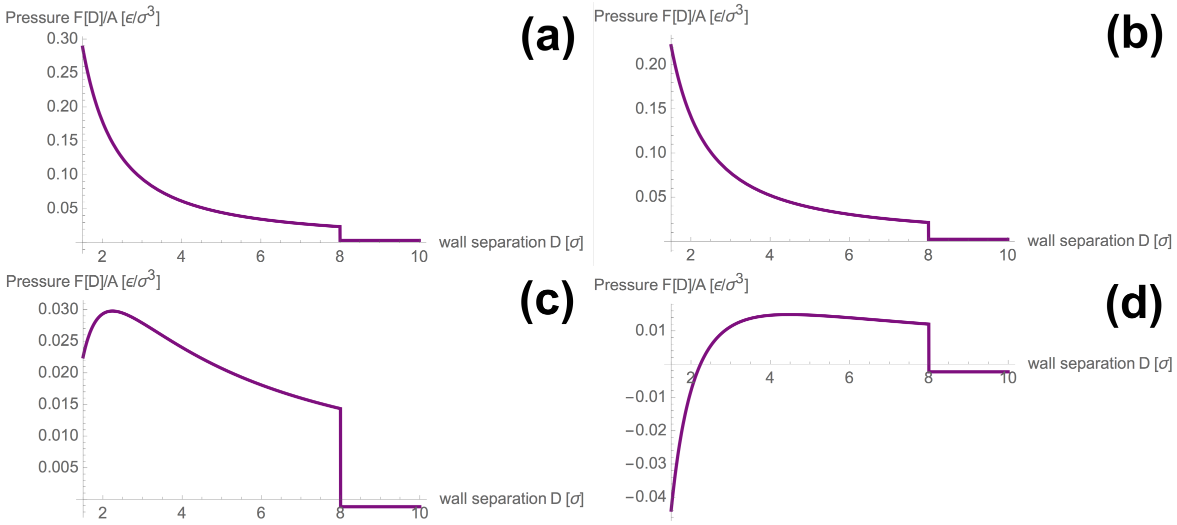

Figure 3.6. The effective pressures between two flat, ligated plates given from Equations (3.33), (3.34), and (3.35) with the fixed parameters listed in Table 3.1 at reduced temperature $T^* = 3.0k_BT/\epsilon$, where the Flory chi parameter is (a) $\chi = 0.35$; (b) $\chi = 0.40$; (c) $\chi = 0.55$; and (d) $\chi = 0.60$.

given to be

$$v_{\text{eff}}(r) = v_{\text{vdW}}(r) + v_{\text{mix}}(r) + v_{\text{el}}(r) + v_{\text{dep}}(r), \tag{3.36}$$

where $v_{\text{dep}}(r)$ represents the depletion interactions between two NPs. The vdW interactions are given by [1, 10, 22, 33]

$$v_{\text{vdW}}(r) = -\frac{H}{12}\left[\frac{1}{r^2-1} + \frac{1}{r^2} + 2\ln\left(1 - \frac{1}{r^2}\right)\right], \tag{3.37}$$

where $H$ is the effective Hamaker constant. Note that in the limit of $r \to \infty$, we get that

$$v_{\mathrm{vdW}}(r) = -\frac{H}{36r^6},$$

therefore recovering the well-known $1/r^6$ dependence [1, 33].

It turns out that $v_{\mathrm{mix}}(r)$ and $v_{\mathrm{el}}(r)$ for the case of two similar spheres are actually derived from Equations (3.21) and (3.25) by considering the Derjaguin Approximation [24, 33, 59]. This approximation allows one to relate the force between two spheres of radii $R_1, R_2$ (which need not be equal) and surface separation $D$ with the potential energy between two flat plates with the same surface separation $D$ using [33]

$$F_S(D) = 2\pi \frac{R_1 R_2}{R_1 + R_2} U(D), \tag{3.38}$$

where $F_S(D)$ is the force between the two spheres and $U(D)$ is the potential energy between two flat plates. Note that if $R_1 = R_2 = a$ (in the case of our system),

$$F_S(D) = 2\pi \frac{a^2}{2a} U(D) = \pi a U(D).$$

In order to use the Derjaguin approximation to derive the change in Gibbs free energy for two equal spheres, we must consider the case where $d << a$, for a reason that will become clear soon. By considering the contributions to the change in Gibbs free energy from rings around the spheres (see Figure 3.7), we have that [24]

$$\Delta G_D^{\mathrm{spheres}} = 2\pi \int_0^\infty h \Delta G_D^{\mathrm{FP}} dh, \tag{3.39}$$

where $\Delta G_D^i$ refers to the change in Gibbs free energy for two spherical or planar surfaces with separation distance $D$. By geometry (Figure 3.7), we have that

$$\frac{s - D}{2} = a - \sqrt{a^2 - h^2}.$$

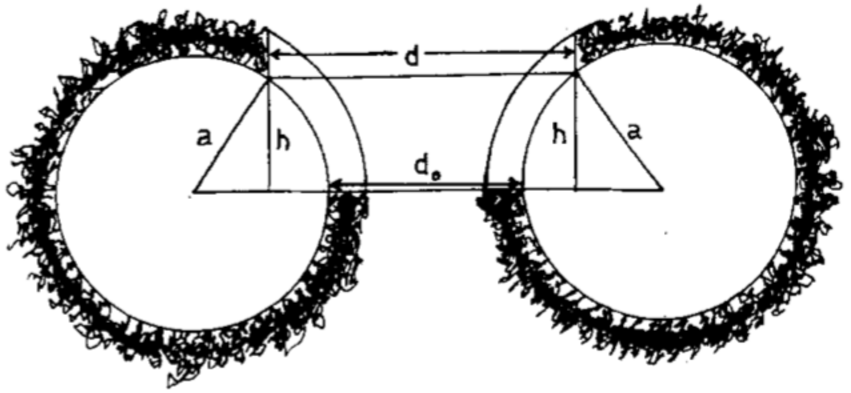

Figure 3.7. A representation of the Derjaguin approximation, used to derive the effective potential between two ligated spheres of equal size [24]. Note that we represent $d$ as $s$ and $d_0$ as $D$ in the formulas in this section because $d$ in this figure is *not* the contour length of the ligands. Reprinted by permission from RightsLink Permissions Springer Customer Service Centre GmbH: Springer Nature Kolloid-Z. u. Z. Polymere, R. Evans and D. H. Napper, Steric Stabilization II: A generalization to Fischer's solvency theory, Copyright 1973. `https://link.springer.com/journal/396`

Differentiating both sides (noting that $D$ and $a$ are constant when we flatten or widen the trapezoid) implies that

$$\frac{1}{2}ds = \frac{2h}{2\sqrt{a^2-h^2}}dh$$

or equivalently,

$$a\sqrt{1-\frac{h^2}{a^2}}ds \approx ads = 2hdh,$$

where we have used the approximation that $a >> h$ from the Derjaguin approximation [24]. Using Equation (3.21), Equation (3.39) then becomes

$$\frac{\Delta G_D^{\text{spheres}}}{k_BT} = \pi a \int_0^\infty \frac{\Delta G_D^{\text{FB}}}{k_BT} ds = 2\pi a \frac{V_S^2}{V_1}\left(\frac{1}{2}-\chi\right) v^2 i^2 \int_{\delta_0}^\infty I d\delta, \tag{3.40}$$

where we have introduced the dimensionless variable $\delta = D/\langle r^2\rangle^{1/2}$ ($\langle r^2\rangle^{1/2}$ is the root-mean-square length of the ligands in free solution) and the letter $I \equiv \langle r^2\rangle^{1/2}[\int_0^D \hat{\rho}_D^2 dx + \int_0^D \hat{\rho}_D\hat{\rho}'_D dx - \int_0^D \hat{\rho}_\infty^2 dx]$.

We can now take the terms that we calculated in the case of two flat plates and derive the free energy of mixing for two equal spheres. For $d < D < 2d$, we use Equation (3.29):

$$\frac{\Delta G_D^S}{k_BT} = 2\pi a \frac{V_S^2 v^2 i^2}{V_1}\left(\frac{1}{2}-\chi\right)\int_\delta^\infty \frac{2\langle r^2\rangle^{1/2}}{d}\left[1-\frac{D}{2d}\right] d\delta. \tag{3.41}$$

Note that since the free energy of mixing vanishes when $D > 2d$, the upper bound of the integral is replaced with $2d/\langle r^2\rangle^{1/2}$, or $\delta_{2d} = \delta_\infty$. Equation (3.41) then becomes

$$\begin{aligned}
\frac{\Delta G_D^S}{k_B T} &= 2\pi a \frac{V_S^2 v^2 i^2}{V_1}\left(\frac{1}{2} - \chi\right)\int_\delta^{\delta_{2d}} \frac{4}{\delta_{2d}}\left[1 - \frac{\delta}{\delta_{2d}}\right] d\delta \\
&= 2\pi a \frac{V_S^2 v^2 i^2}{V_1}\left(\frac{1}{2} - \chi\right)\left[\frac{4\delta}{\delta_{2d}} - \frac{2\delta^2}{\delta_{2d}^2}\right]_\delta^{\delta_{2d}} \\
&= 2\pi a \frac{V_S^2 v^2 i^2}{V_1}\left(\frac{1}{2} - \chi\right) 2 - \frac{4\delta}{\delta_{2d}} + \frac{2\delta^2}{\delta_{2d}^2} \\
&= 2\pi a \frac{V_S^2 v^2 i^2}{V_1}\left(\frac{1}{2} - \chi\right) 2\left(\frac{\delta^2}{\delta_{2d}^2} - 2\frac{\delta}{\delta_{2d}} + 1\right) \\
&= 2\pi a \frac{V_S^2 v^2 i^2}{V_1}\left(\frac{1}{2} - \chi\right) 2\left(\frac{\delta}{\delta_{2d}} - 1\right)^2 \\
&= 2\pi a \frac{V_S^2 v^2 i^2}{V_1}\left(\frac{1}{2} - \chi\right) 2\left(\frac{D}{2d} - 1\right)^2 \\
&= 4\pi a \frac{V_S^2 v^2 i^2}{V_1}\left(\frac{1}{2} - \chi\right)\left(1 - \frac{D}{2d}\right)^2 .
\end{aligned} \tag{3.42}$$

Using the center-center distance $r = 2a + D$ in SI units, we have that $D = r - 2a$. Then Equation (3.42) simplifies further to

$$\begin{aligned}
\frac{\Delta G_D^S}{k_B T} &= 4\pi a \frac{V_S^2 v^2 i^2}{V_1}\left(\frac{1}{2} - \chi\right)\left(1 - \frac{r - 2a}{2d}\right)^2 \\
&= \pi a^3 \frac{V_S^2 v^2 i^2}{2d^2 V_1}\left(\frac{1}{2} - \chi\right)(2d + 1 - r)^2 \quad \text{(now in units of particle diameter } 2a\text{)} \\
&= \frac{\pi a^3}{2V_1}\phi_{\text{av}}^2\left(\frac{1}{2} - \chi\right)(r - (1 + 2d))^2 ,
\end{aligned} \tag{3.43}$$

where $\phi_{\text{av}} \propto V_s v i/d$ if $d << a$.

For $D < d$, we use Equations (3.31) and (3.32). For the free energy of mixing, we have that

$$\frac{\Delta G_D^S}{k_B T} = 2\pi a \frac{V_S^2 v^2 i^2}{V_1}\left(\frac{1}{2} - \chi\right)\langle r^2\rangle^{1/2}\int_\delta^\infty \left[\frac{2}{D} - \frac{1}{d}\right] d\delta. \tag{3.44}$$

Noting that the highest value for $D$ is $d$ in this regime, it follows that we have

$$\begin{aligned}
\frac{\Delta G_D^S}{k_BT} &= 2\pi a \frac{V_S^2 v^2 i^2}{V_1}\left(\frac{1}{2}-\chi\right)\int_\delta^{\delta_d}\left[\frac{2}{\delta}-\frac{1}{\delta_d}\right]d\delta \\
&= 2\pi a \frac{V_S^2 v^2 i^2}{V_1}\left(\frac{1}{2}-\chi\right)\left[2\ln\delta-\frac{\delta}{\delta_d}\right]_\delta^{\delta_d} \\
&= 2\pi a \frac{V_S^2 v^2 i^2}{V_1}\left(\frac{1}{2}-\chi\right)\left[2\ln\delta_d-2\ln\delta-1+\frac{\delta}{\delta_d}\right] \\
&= 2\pi a \frac{V_S^2 v^2 i^2}{V_1}\left(\frac{1}{2}-\chi\right)\left[2\ln\frac{\delta_d}{\delta}+\frac{\delta}{\delta_d}-1\right] \\
&= 2\pi a \frac{\phi_{\text{av}}^2 d^2}{V_1}\left(\frac{1}{2}-\chi\right)\left[2\ln\frac{d}{D}+\frac{D}{d}-1\right].
\end{aligned} \tag{3.45}$$

We once again substitute $D = r - 2a$ in order to find $v_{\text{mix}}(r)$:

$$\begin{aligned}
\frac{\Delta G_D^S}{k_BT} &= 2\pi a \frac{\phi_{\text{av}}^2 d^2}{V_1}\left(\frac{1}{2}-\chi\right)\left[2\ln\frac{d}{D}+\frac{D}{d}-1\right] \\
&= 2\pi a \frac{\phi_{\text{av}}^2 d^2}{V_1}\left(\frac{1}{2}-\chi\right)\left[2\ln\frac{d}{r-2a}+\frac{r-2a}{d}-1\right] \\
&= \pi a^3 \frac{\phi_{\text{av}}^2 d^2}{V_1}\left(\frac{1}{2}-\chi\right)\left[2\ln\frac{d}{r-1}+\frac{r-1}{d}-1\right] \quad \text{(units of particle diameter)}
\end{aligned} \tag{3.46}$$

In order to obtain continuity with Equation (3.43) at $r = 1+d$, we add a constant, which is included in Equation (3.50).

For the elastic potential, Equation (3.32) yields ($\delta_\infty = \delta_d = d/d = 1$)

$$\begin{aligned}
\frac{\Delta G_{\text{el}}}{k_BT} &= -2v\pi ad\int_\delta^{\delta_\infty}\ln\left(\frac{\delta}{\delta_\infty}\right)d\delta \\
&= -2v\pi ad\left[\delta\ln\delta-\delta\right]_\delta^1 \\
&= -2v\pi ad\left[-1-\delta\ln\delta+\delta\right] \\
&= 2va\pi\left[D\ln\frac{D}{d}-D+d\right] \\
&= 2va\pi\left[D\left(\ln\frac{D}{d}-1\right)+d\right].
\end{aligned} \tag{3.47}$$

This implies that

$$\begin{aligned}\frac{\Delta G_{\text{el}}}{k_B T} &= 2va\pi\left[D\left(\ln\frac{D}{d}-1\right)+d\right] \\ &= 2va\pi\left[(r-2a)\left(\ln\frac{r-2a}{d}-1\right)+d\right] \\ &\to va^2\pi\left[(r-1)\left(\ln\frac{r-1}{d}-1\right)+d\right] \quad \text{(units to particle diameter).}\end{aligned} \tag{3.48}$$

We summarize the equations that we obtained from the derivation below, using the notation found in Khan et al. [10] and that which is used for our simulations. For the first regime $1+d < r < 1+2d$ we use Equation (3.43) and have

$$\frac{v_{\text{mix}}}{k_B T} = \frac{\pi a^3}{2v_m}\phi_{\text{av}}^2\left(\frac{1}{2}-\chi\right)[r-(1+2d)]^2, \tag{3.49}$$

and the second regime ($r < 1+d$) is given by

$$\frac{v_{\text{mix}}}{k_B T} = \frac{\pi a^3}{v_m}\phi_{\text{av}}^2\left(\frac{1}{2}-\chi\right)d^2\left[2\ln\frac{d}{r-1}+\frac{r-1}{d}-\frac{1}{2}\right], \tag{3.50}$$

where $v_m$ is the molecular volume of a toluene molecule in $\text{Å}^3$ ($V_1$ from the derivation), $\phi_{\text{av}}$ is the volume fraction of ligands in the ligand layer ($1 < r < 1+d$), and $\chi$ is the Flory $\chi$ parameter given by [10, 58, 60]

$$\chi = \frac{v_s N_A(\delta_s-\delta_m)^2}{RT}+0.34, \tag{3.51}$$

where $v_s$ is the molecular volume of the toluene molecules in $\text{m}^3$, and $\delta_s$ and $\delta_m$ are the Hildebrand solubility parameters (HSP) for the solvent and ligands (respectively) in $\text{Pa}^{1/2}$, given by [61]

$$\delta_i = \sqrt{\rho\frac{\Delta H_v - RT}{M}}, \tag{3.52}$$

where $\rho$ is the density of the molecule, $\Delta H_v$ is the heat of vaporization, $R$ is the ideal gas constant, and $M$ is the molar mass of the molecule.

The change in free energy due to the loss of configurational entropy $v_{\text{el}}$ is nonzero when the ligands of two NPs are undergoing compression ($r < 1 + d$), and is given to be [10, 22, 27]

$$\frac{v_{\text{el}}}{k_B T} = \pi v a^2 \left[ (r-1) \left( \ln \frac{r-1}{d} - 1 \right) + d \right]. \tag{3.53}$$

Table 3.2. Input parameters for AuNCs with dodecanethiol ligands in toluene [10, 22].

| System Parameter | Value |
|---|---|
| Particle radius $a$ | 25 Å |
| Contour ligand length $d$ | 17.74 Å |
| Hamaker constant $H$ | 75.5 $k_B T$ |
| HSP of solvent $\delta_s$ | $1.82 \times 10^4$ Pa$^{1/2}$ |
| HSP of ligand $\delta_m$ | $1.60 \times 10^4$ Pa$^{1/2}$ |
| Volume of solvent molecule $v_s$ | $1.78 \times 10^{-28}$ m$^3$ |
| Ligand molecular vol. $v_{\text{ligand}}$ | $3.98 \times 10^{-28}$ m$^3$ |

Table 3.2 shows the parameters used for our simulations of AuNCs with dodecanethiol ligands in toluene. The effective Hamaker constant $H$ was estimated to be the Hamaker constant of two AuNCs across a dodecanethiol medium [10]. We did not consider toluene when calculating the effective Hamaker constant because the ligand layers were assumed to affect the interactions more than the solvent molecules due to the proximity of the ligands to the NCs [10]. One may argue that when $v$ is low (corresponding to fewer ligands attached to the surface of the NC), our estimate for $H$ is not accurate. While this may be true, we determined that the change in $H$ would not have a large effect on the effective pair potential. In addition, the gold-gold Hamaker constant is fairly large due to the high polarizability of gold [49], and it is therefore unnecessary to consider the solvent when calculating the effective Hamaker constant due to the negligible change induced by it. In order to calculate $H$ for two NCs of the same material acting across a medium, we use the approximation [33, 62]

$$H = H_{131} \simeq (\sqrt{H_{11}} - \sqrt{H_{33}})^2, \tag{3.54}$$

where $H_{11}$ is the NC-NC Hamaker constant in vacuum and $H_{33}$ is the ligand-ligand Hamaker constant in vacuum. The Hamaker constant of AuNPs has been found to change with the size of the NPs and with temperature [63], but both of these values are fixed across our simulations, so we do not have to consider the effects of changing the NC radius or the temperature.

To simulate AuNP dispersions, we varied $v$ and the NC volume fraction $\phi$. By varying $v$, parameters such as the average volume fraction of ligands in the spherical shell around the NC of thickness $d$ (denoted by $\phi_{\text{av}}$) also varied. By varying $\phi$, the dimensions of our system varied, as the volume of the system was given to be

$$V = \frac{\pi}{6}\frac{N}{\phi}, \tag{3.55}$$

where $V$ is the volume of the system (a cube) and $N$ is the number of nanoparticles. We will now briefly discuss the calculations involved for determining values for the parameters in our simulations.

The surface area of a spherical NC is $A_{\text{NC}} = 4\pi a^2$. Using the value of $a$ from Table 3.2, we find that $A_{\text{NC}} = 7.8 \times 10^3$ Å$^2$. We can then approximate the number of ligands per NP to be $N_{\text{ligand}} = A_{\text{NC}} v$, which is on the order of $10^2$ ligands per NP [10].

The molecular volume of toluene $v_{\text{mol}}$ is calculated using the molar mass and density via

$$v_{\text{mol}} = \frac{\text{molar mass, g/mol}}{\text{density, g/m}^3}\frac{1}{N_A}, \tag{3.56}$$

where $N_A$ is Avogadro's number. The values of $N_{\text{ligand}}$ and $v_{\text{mol}}$ are then used to estimate $\phi_{\text{av}}$ by [10]

$$\phi_{\text{av}} = N_{\text{ligand}}\frac{v_{\text{mol}}}{V_{\text{shell}}} = N_{\text{ligand}}\frac{v_{\text{mol}}}{\frac{4}{3}\pi\left[(a+d)^3 - a^3\right]}, \tag{3.57}$$

where $V_{\text{shell}}$ is the volume of the spherical shell of thickness $d$ around the NC.

For all of our simulations of AgNP dispersions and about half of our simulations of AuNP dispersions, we actually used the incorrect equation for $v_{\text{mix}}(r)$ in the regime where $1 < r < 1 + d$:

$$\frac{v_{\text{mix}}}{k_B T} = \frac{\pi a^3}{v_m}\phi_{\text{av}}^2\left(\frac{1}{2} - \chi\right)d^2\left[3\ln\frac{d}{r-1} + 2\frac{r-1}{d} - \frac{3}{2}\right]. \tag{3.58}$$

Equation (3.58) is the formula for $v_{\text{mix}}$ that was found in Khan et al. and Smitham et al. [10, 25]. However, upon working through the derivation and reading Evans et al. [27], it was discovered that Smitham and coworkers had added an extra factor of 2 in one of their steps. Figure 3.9 shows the difference in the effective pair potential when we consider the incorrect and the correct expressions

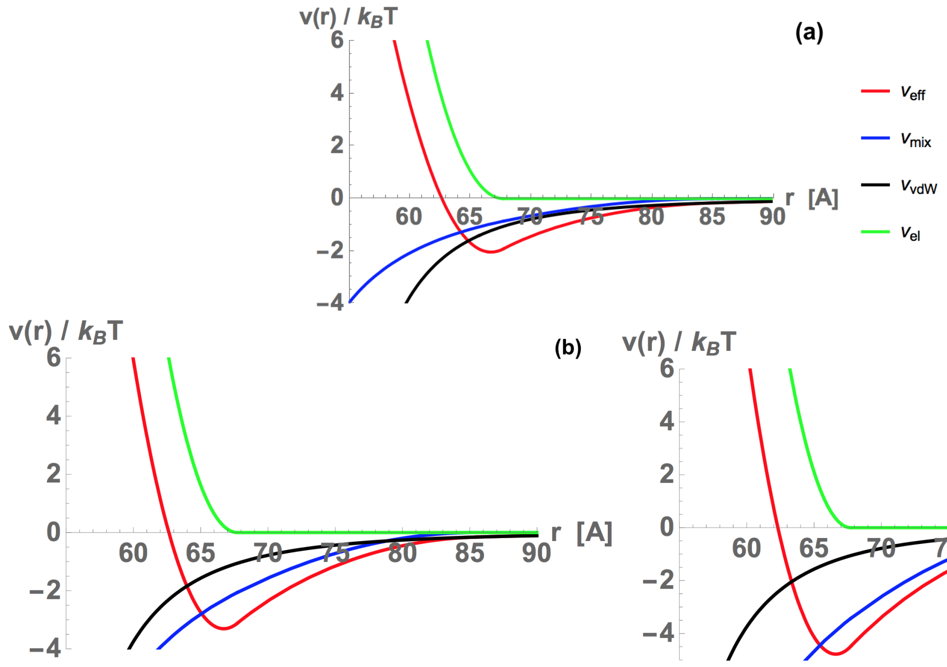

Figure 3.8. The effective pair potential (no depletion interactions) from Equation (3.36) where the ligand coverage is (a) $\upsilon = 0.03$ Å$^{-2}$, (b) $\upsilon = 0.0465$ Å$^{-2}$, and (c) $\upsilon = 0.06$ Å$^{-2}$.

for the free energy of mixing in the regime where the ligands can undergo compression. The difference becomes apparent when the NCs are very close to contact. However, the change in the expression for $v_{\text{mix}}$ had a negligible effect on our results, and for systems with a low average volume fraction $\phi$, the repulsive force prevents the NCs from becoming close enough for the difference to play a large role in the equilibrium structures. Note that the blue (correct) curve is steeper than the red (incorrect) curve, signifying that there is a higher repulsive force between NCs at low separation distance. This is consistent with our observations as we ran simulations to verify that the change in the expression for $v_{\text{mix}}$ would be negligible.

The depletion interactions were not considered for the model of AuNP dispersions. This is because the ligands were grafted to the surface of the AuNCs and there was assumed to be no free ligand in the system. However, in our model for AgNP dispersions, the oleylamine ligands were adsorbed to the surface of the AgNCs. In addition, toluene is a “good solvent” for oleylamine, and if depletion interactions are not included, then the NP dispersions could not self-assemble into a stable superlattice when the initial NC volume fraction was no more than $\phi = 0.15$. Therefore, in order to accurately model the system, we believe that depletion interactions are needed in order to account for the presence of free ligand in the system. While the amount of free ligand could change

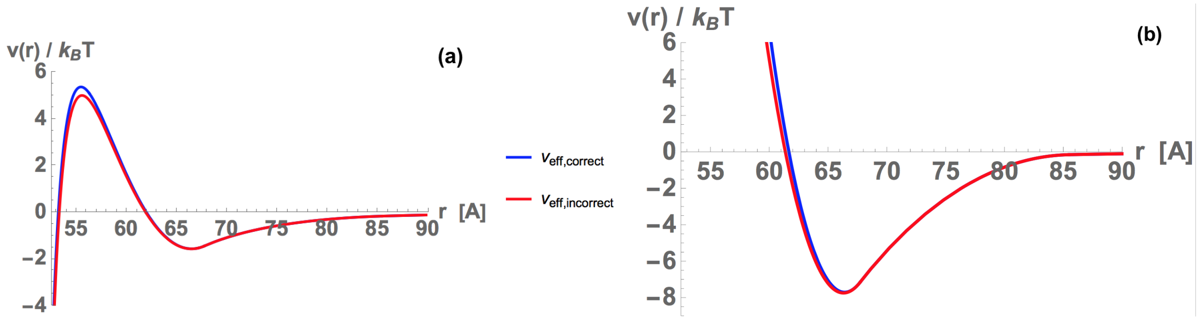

Figure 3.9. The effective pair potential from Equation (3.36) without depletion interactions. The plots show the change in $v_{\rm eff}$ between the incorrect (Equation (3.58)) and correct (Equation (3.50)) free energy of mixing terms in the regime of $1 < r < 1 + d$, where the ligand coverage is (a) $v = 0.02$ Å$^{-2}$ and (b) $v = 0.08$ Å$^{-2}$.

as ligands adsorb and desorb to the surface of the NCs, we kept the concentration of free ligand fixed in our simulations for simplicity.

The term for the depletion interactions depends on parameters such as the number of free ligands per unit volume of solution (the number density) $n$ (which represents the osmotic pressure if energy is in units of $k_BT$) and the thickness of the exclusion region $L$, and is given by the formula [1]

$$\frac{v_{\rm dep}}{k_BT} = -nV_{\rm dep}, \tag{3.59}$$

where $V_{\rm dep}$ is the volume of overlap between the exclusion regions around two NPs, given to be [1, 24, 25, 42]

$$V_{\rm dep} = \frac{4\pi}{3}(a+d+L)^3\left(1 - \frac{3r}{4(a+d+L)} + \frac{r^3}{16(a+d+L)^3}\right). \tag{3.60}$$

We approximate each free ligand as a sphere of radius $L = d/2$ (a coarse-grained model). The bounds of the depletion interactions were taken to be $2a < r < 2(a + d + L)$, though the literature uses the bounds $2a < r < 2(a + L)$ and takes $d = 0$ for hard spheres.

Because the NPs consisted of an NC and a ligand layer, one may argue that the bounds for the depletion interactions should be $2(a + d) < r < 2(a + d + L)$. However, the bounds $2a < r < 2(a + d + L)$ suggest that the ligands are flexible rather than rigid brushes, which is consistent with the experimental observations. If the bounds were taken to be $2(a + d) < r < 2(a + d + L)$, then the depletion interactions would not compete with the elastic pair potential and the free ligands therefore would not contribute to the self-assembly of the NPs, which is inconsistent with experimental observations. By considering the bounds $2a < r < 2(a + d + L)$, we were able to generate a potential well with a depth of several $k_BT$ at a center-center separation of about 87 Å. As two NCs approach contact, the adsorbed ligand brushes may repel the free ligands away from the NPs because toluene is a "good solvent" in the presence of oleylamine. This is why our upper bound is taken to be $2(a + d + L)$. As the AgNCs approach contact, the ligands may also desorb from the surface of the NCs to avoid interactions with other ligands, therefore maximizing interactions with the toluene molecules. This is another reason why we took the lower bound to be $2a$ instead of $2(a + d)$. We hope that our molecular dynamics simulations will help to verify our reasoning for choosing the bounds as we did.

Table 3.3. Input parameters for AgNCs with oleylamine ligands in toluene [10, 62, 63, 64]. Depletion interactions are present in this system due to the ligands being adsorbed rather than grafted.

| System Parameter | Value |
|---|---|
| Particle radius $a$ | 30 Å |
| Ligand coverage $v$ | 0.0465 Å$^{-2}$ |
| Contour ligand length $d$ | 27.72 Å |
| Hamaker constant $H$ | 16.29 $k_BT$ |
| HSP of solvent $\delta_s$ | $1.82 \times 10^4$ Pa$^{1/2}$ |
| HSP of ligand $\delta_m$ | $1.66 \times 10^4$ Pa$^{1/2}$ |
| Volume of solvent molecule $v_s$ | $1.78 \times 10^{-28}$ m$^3$ |
| Ligand molecular vol. $v_{\text{ligand}}$ | $5.46 \times 10^{-28}$ m$^3$ |
| Radius of depletant sphere $L$ | $d/2$ |

Table 3.3 displays the fixed values used to generate the plots in Figure 3.10. Note that the Hamaker constant was calculated using Equation (3.54) by estimating the Hamaker constant of oleylamine using the Hamaker constant of hexane [65]. Figure 3.10 shows different scenarios for the effective pair potential by ignoring different terms in the effective pair potential. For the reasons stated above, we settled on using the effective pair potential in Figure 3.10(d). Although we initially varied the ligand coverage and the free ligand concentration of the nanoparticles across our simulations (of course, the initial NC volume fraction was varied as well), we later varied only the free ligand concentration and the initial NC volume fraction because the ligand coverage was

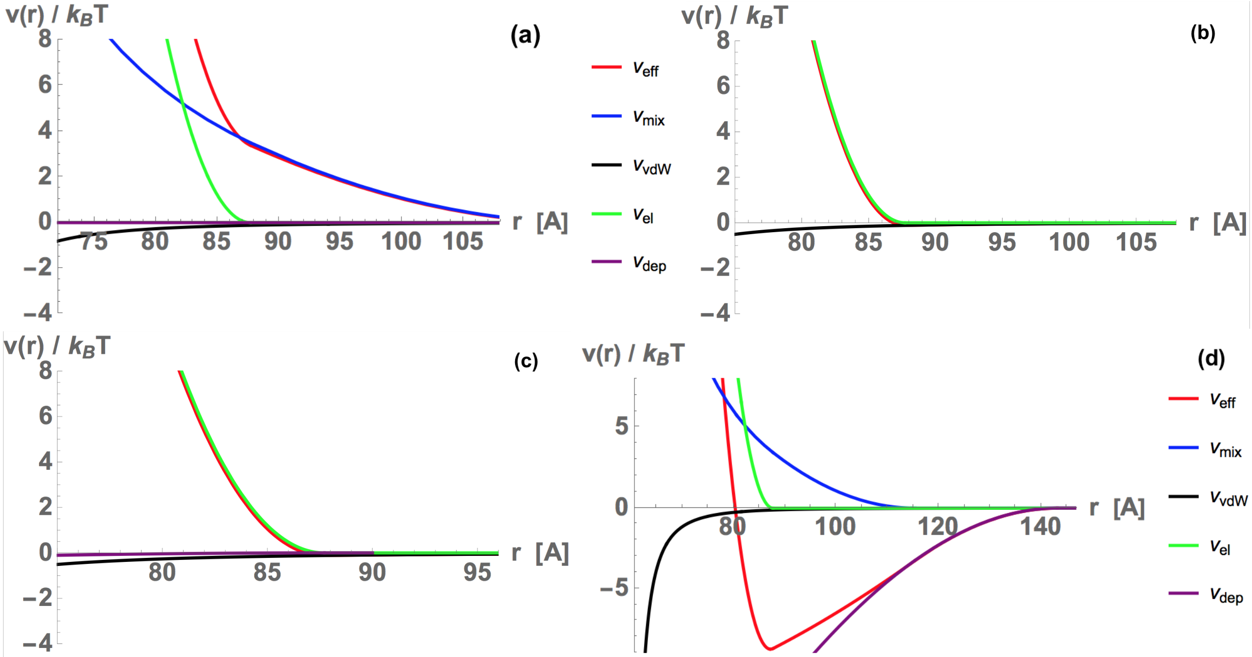

Figure 3.10. (a) The effective pair potential from Equation (3.36) without depletion interactions for AgNP dispersions at $T = 298$ K. The repulsive free energy of mixing implies that stable superlattices cannot form unless $\phi > 0.22$ with the three components shown. (b) The effective pair potential without the free energy of mixing and depletion interactions. (c) The effective pair potential without the free energy of mixing but with the depletion interactions (purple curve) using $\phi_{\mathrm{dep}} = 0.112$. Note that the bounds of the depletion interactions are $2a < r < 2(a+L)$ for this plot, showing that the exclusion of free ligands by only the bare nanocrystals does not yield a significant attractive interaction. (d) The effective pair potential from Equation (3.36) where $\phi_{\mathrm{dep}} = 0.446$. The bounds of the depletion potential in this plot are $2a < r < 2(a + d + L)$.

unknown in the experiments that we compared our results with. The volume fraction of the free ligands in the simulation box is given to be

$$\phi_{\mathrm{dep}} = \frac{4}{3}\pi L^3 n. \tag{3.61}$$

## 3.3. Computational Methods

There are many open source packages which have been developed and maintained for many years, and these packages can be used for a myriad of research topics. Density functional theory, molecular dynamics (MD) methods, and Monte Carlo (MC) methods are just a few of the tools available to a computational physicist. We used both MC and MD methods for our work, and in this section we will describe the basics of both approaches and the methods that we developed.

### 3.3.1. Monte Carlo Methods

Monte Carlo methods consist of trial moves which are rejected or accepted based on probability. Since we are only interested in the equilibrium structure and stability of the NP dispersions, Monte Carlo methods provide a valid approach to analyzing equilibrium structures without generating large (tens or hundreds megabytes in size) output files or solving Newton's equations at every step, as done in classical molecular dynamics. Hence, Monte Carlo methods are not able to describe the actual dynamics of NP dispersions. Indeed, the state at each Monte Carlo step is independent of the velocities of the NPs, which are never stored in our methods.

Monte Carlo methods allow researchers to solve problems (not restricted to science) by simulating random variables and considering probability distribution functions [66]. The general flow of a basic Monte Carlo code is to [66]

1. Make a trial move;
2. Using a probability distribution function, accept or reject the trial move;
3. Update the state of the system;
4. Repeat.

Monte Carlo methods have a lot of applications. For example, they can be used to describe collisions of particles and the mean free path length, scattering of particles, and the equilibrium state of nanoparticle dispersions.

Our methods were created in the Open Source Physics Library [67]. The Open Source Physics Library is a free-to-use library containing many different simulations of numerical methods and physical systems written in the programming language of Java. By using an already-existing package, we did not have to create our own visualization package for our simulations. In addition, many necessary calculations for our methods have already been created in the Open Source Physics Library. This includes calculating the distance between two NPs given periodic boundary conditions (PBCs), which were implemented in all of our Monte Carlo simulations. Periodic boundary conditions are used so that if a particle goes past one face of the simulation box, it re-appears near the opposite face of the box. Particles are therefore able to interact with each other from opposite sides of the simulation box. For example, we have to consider the interactions between particles on the far left-hand-side of the box and particles on the far right-hand-side of the box. While PBCs allow us to determine the equilibrium structures of very large NP dispersions using around $N = 500$ NPs, we also have to be careful with the number of NPs used. When $N < 250$, finite size effects can begin to influence the structure of the NP dispersions. In addition, 500 NPs is not enough to see phase separation in our systems.

We used the Metropolis algorithm in the canonical ensemble (fixed $NVT$) for our simulations [68, 69]. For each step in a simulation, a trial displacement was attempted for one NP. This was done by using random numbers to generate a change in the $x-, y-$, and $z-$coordinates of the NP and then calculating the difference of the total energy $\Delta E$ after the trial displacement and the energy of the system before the trial displacement was attempted. If $\exp[-\beta\Delta E]$ (where $\beta \equiv 1/(k_BT)$) was less than a random number chosen between 0 and 1 or if the trial displacement caused overlap of two NCs, then the trial displacement was rejected and we moved on to the next NP. If the trial displacement was accepted, then the energy of the system was updated and another particle underwent trial displacement. After a specified amount of MC steps known as the "equilibration time" have passed in the simulation, the simulation updated the plots for the radial distribution function and the static structure factor (see Section 3.4). The equilibration time is a period in which the plots are not updated so that the initial configuration of the structure does not contribute to the calculations. This is because the equilibrium structure could be vastly different from the initial configuration of the system. It is also important to note that any equilibrium structure can only be classified as metastable or unstable: it is always possible that if a simulation is run for a larger number of MC steps, then the NP dispersion will leave a metastable state.

### 3.3.2. Molecular Dynamics Methods

Molecular dynamics simulations can be used if one *is* interested in the actual dynamics of the system. As stated in Section 3.3.1, "classical molecular dynamics" means that Newton's equations of motion are solved during each timestep of the simulation, and the velocities and positions of the particles are updated accordingly. Molecular dynamics methods can be used to simulate a large variety of systems at the nanoscale.

We used LAMMPS [55] (`http://lammps.sandia.gov`) to run our simulations of surfaces covered with ligand chains. LAMMPS uses the Verlet algorithm to update the trajectories of the particles in each timestep. As in our MC methods, we ran our simulations in the canonical ensemble

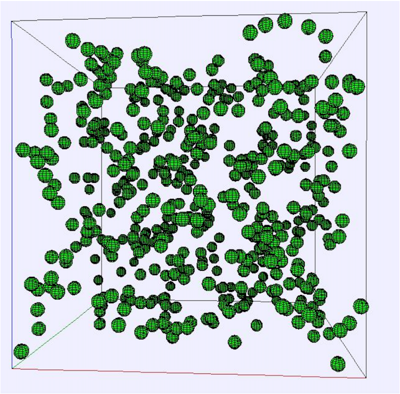

Figure 3.11. A snapshot of our MC simulation. The visualization updated as the simulations progressed.

and imposed PBCs in the $x$-direction and the $y$-direction. However, we imposed a fixed boundary in the $z$-direction. This is because the two flat plates were parallel to the $xy$-plane and the segments of the ligand chains were bounded between the two flat plates.

Before we actually started a simulation, we initialized the chains on the correct surfaces and bonded neighboring atoms in a chain together. As stated in Section 3.1, the initialization of the chains was done using scripts that we wrote in C++ (flat plates) and Python (spherical surfaces). These scripts output the atom types (fixed or moving) and the Cartesian coordinates of each atom. The bonding of the atoms was accomplished by using a .tcl script with Visual Molecular Dynamics (VMD) [70], adapted by Professor Alan Denton from a tutorial written by Axel Kohlmeyer. The .tcl script generated a data file with atom, bonding, and angle information along with information about the boundaries of the simulation box. This data file allowed the system to be initialized in LAMMPS, while a separate input script was used to run the simulation. All of these scripts are included in the Appendix.

During the simulation (every $1 \times 10^4$ timesteps out of $1.1 \times 10^6$ total steps), LAMMPS wrote the coordinates of each atom (along with the atom numbers) to one file, and the total energy, temperature, and pressure into another file. The first $1 \times 10^5$ steps of each simulation were used as the equilibration time, allowing for the temperature of the system to reach a nearly constant value. An observant reader may wonder why the temperature needed time to reach a "nearly constant" value when it should be fixed in the canonical ensemble. The temperature actually varies slightly around the temperature specified in the input script: in order to keep the temperature "constant", LAMMPS rescales the velocities of the particles using the Nosé-Hoover thermostat. Therefore, the temperature fluctuates around the equilibrium temperature.

After a simulation reached completion, we used VMD and the Open Visualization Tool (OVITO) [71] (`http://ovito.org/`) to generate videos and snapshots of our system at various timesteps. Using OVITO also allowed us to make sure that we were setting the boundaries of our box correctly, that the chains initialized correctly (without overlap of 2 chains, for example), and to get a visualization on whether or not the effective force between two surfaces might be attractive or repulsive (qualitatively). In addition, we created Python scripts to calculate the effective pressure between the two surfaces. The forces were calculated directly (i.e. by using the coordinates of each bead and the derivative of Equation (3.3) with respect to $r$) at each timestep that the segment coordinates were output from the simulation and the $x-$, $y-$, and $z-$components of the force were calculated. The bead-bead forces were then added up to determine the components of the effective force. Finally, the returned value of the effective force (divided by the area of the flat plates in our simulation) was averaged over all timesteps. The effective pressures during the equilibration time were not calculated or included in the time-average.

Mathematically, the effective force was given to be [50, 51, 52]

$$F_{\text{eff}}(D) = \left\langle \frac{1}{2}(F_{z,\text{left}-\text{right}} - F_{z,\text{right}-\text{left}}) \right\rangle_{\text{timesteps}}, \tag{3.62}$$

where the subscript $z$ refers to the $z$-component of the force at a specific timestep (the two surfaces were separated in the $z$-direction), $i - j$ means "the force on surface $i$ from surface $j$", and the angular brackets denote an ensemble average (in the canonical ensemble, the ensemble average corresponds to an average over timesteps). A positive value for $F_{\text{eff}}$ implies that the force between two surfaces is repulsive and a negative value implies that the force is attractive. We have included our Python script in the Appendix.

## 3.4. Structural Properties

### 3.4.1. Radial Distribution Function

The radial distribution function $g(r)$ was used to determine the equilibrium structure of the NP dispersions by plotting the number of NPs at a distance $r$ away from another NP. In other words, it allowed us to determine the probability of finding one NP at our location if we were a distance $r$ away from the center of some other NP. We computed $g(r)$ by using the average position of each NP and the formula

$$g(r) = \frac{N(r)}{4\pi r^2 \rho \Delta r}, \tag{3.63}$$

where $N(r)$ is the number of particles in a spherical shell of thickness $\Delta r$ with center-center distance $r$ from a NP at the "origin" ($r = 0$ at the center of this NP) and $\rho$ is the average number density of the NPs. Note that as $r \to \infty$, $g(r) \to 1$, signifying that at any large $r$ away from an NP, we should find another NP. However, this is not obvious from our plots shown in Section 4: due to the PBC, we are unable to plot $g(r)$ for very large $r$, and it is not computationally feasible to run a simulation with hundreds of thousands of atoms in order to obtain $g(r)$ plots which show $r >> 1$.

The process for calculating $g(r)$ started off by taking a pair of NPs and calculating the center-center distance $r$ between them. If this distance was greater than a fixed value (which we set to half of the simulation box length), we did nothing and moved on to the next pair of NPs. If the distance was smaller than the fixed value, we calculated $n = r/\Delta r$ and rounded up to the nearest integer. We then stored the number of pairs of NPs for which we obtained the value $n$ in the bin for a histogram given by $N(r)$. $g(r)$ was then obtained by normalizing the function $N(r)$ using the denominator of Equation (3.63).

By analyzing the positions and heights of the peaks in $g(r)$ at the end of the simulations, we determined if the structure was that of a fluid or the type of crystalline lattice that was present. In particular, if the first three peaks of $g(r)$ for a "perfect" crystalline structure were present in the simulation results, then it was possible that the equilibrium structure was that of a crystal. The theoretical $g(r)$ functions for SC, BCC, and FCC lattices are easy to determine by computing the coordination numbers and positions of atoms in the lattice and then creating a data file containing these positions and coordination numbers. The Appendix contains the class that we used in the Open Source Physics Library to calculate $g(r)$ in our MC simulations.

### 3.4.2. Static Structure Factor

In addition to computing $g(r)$, we also computed the static structure factor $S(q)$, which allowed us to characterize the order of the nanoparticle dispersions on average over a simulation. The static structure factor is a function of the magnitude $q$ of the scattered wave vector $\mathbf{q}$ and is proportional to the intensity of scattered light from the NP dispersion. It is well-known that this function can be obtained by taking the Fourier Transform of $g(r)$ [1, 72]. In practice, we computed $S(q)$ following the simulation directly from the particle coordinates using the equation [72]:

$$S(q) = 1 + \frac{2}{N} \sum_{i<j=1}^{N} \left\langle \frac{\sin(qr_{ij})}{qr_{ij}} \right\rangle, \tag{3.64}$$

where $r_{ij}$ is the center-center distance between particles $i$ and $j$ and the angular brackets denote an average over configurations. Equation (3.64) is equivalent to taking the Fourier Transform of $g(r)$. The Appendix contains the class that we created to calculate $S(q)$ in our Monte Carlo simulations.

The static structure factor helped us analyze the structure and stability of the NP dispersions at the end of a simulation. The Hansen-Verlet freezing criterion states that if the height of the first peak of $S(q)$ is greater than 2.85, then the equilibrium structure is that of a superlattice [73]. This was a rough estimate for our effective pair potential, but when combined together with the criterion set for $g(r)$, we could classify the equilibrium structures as crystals or fluids.

# 4. RESULTS AND DISCUSSION

In this chapter, we present the results that we obtained using both MC and MD methods. In our previous work, we validated our MC methods by comparing our results with results obtained from MD simulations. After validating our methods, we implemented the effective pair potential in Equation (3.36) [74], setting the stage for us to characterize the structures of AuNP and AgNP dispersions to extend the work found in the literature [10] and compare with experimental results, respectively. As stated in Chapter 1, our MD simulations were motivated by our results for AgNP dispersions, and we will therefore end this chapter by presenting and discussing our MD results.

## 4.1. Gold Nanocrystals: Equilibrium Structures

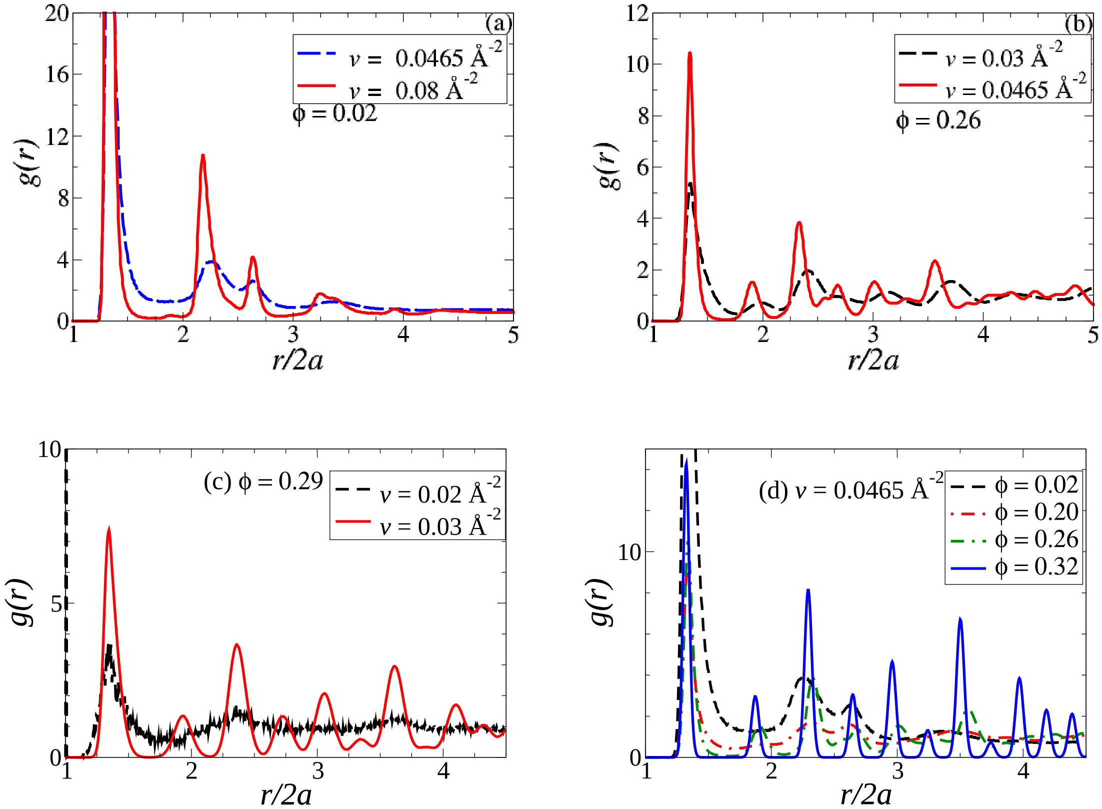

Figure 4.1. Radial distribution function $g(r)$ for different values of hard core volume fraction $\phi$ and ligand coverage $v$: (a) $\phi = 0.02$; (b) $\phi = 0.26$; (c) $\phi = 0.29$; (d) $v = 0.0465$ Å$^{-2}$.

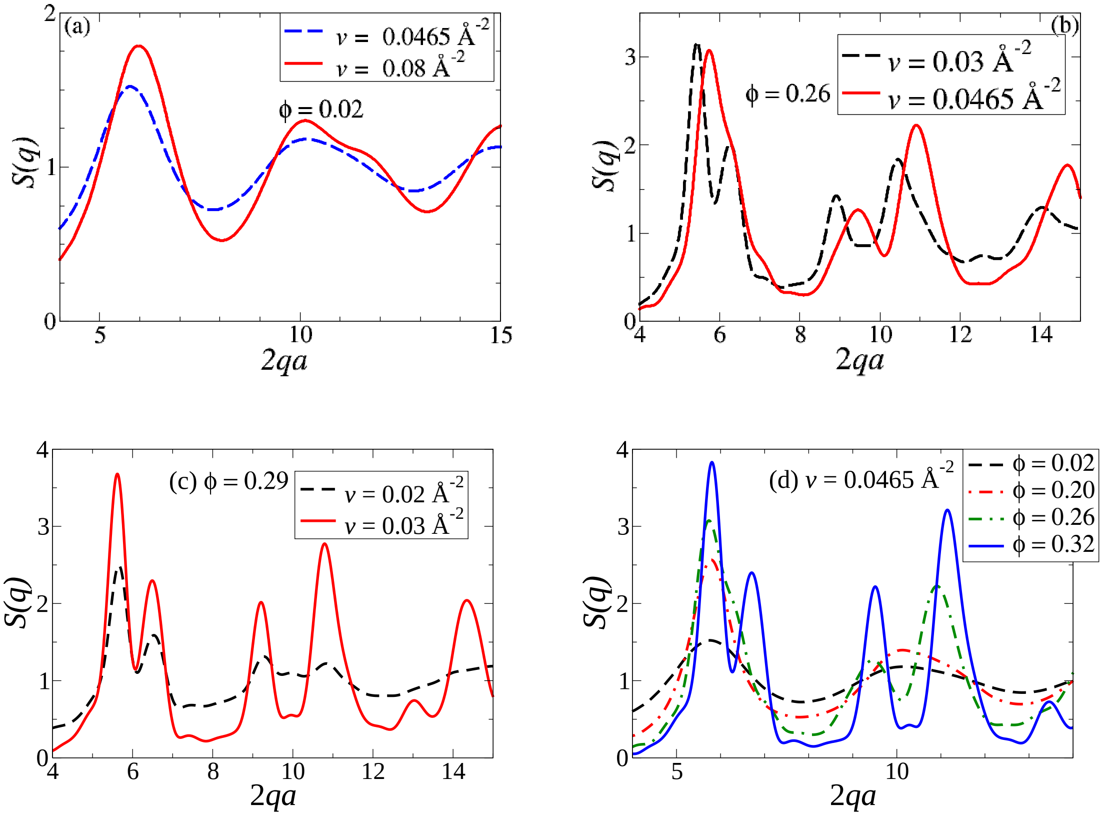

Figure 4.2. Static structure factor $S(q)$ for different values of hard core volume fraction $\phi$ and ligand coverage $v$: (a) $\phi = 0.02$; (b) $\phi = 0.26$; (c) $\phi = 0.29$; (d) $v = 0.0465$ Å$^{-2}$.

Figures 4.1 and 4.2 show plots for $g(r)$ and $S(q)$, respectively, given different values for ligand coverage and hard core volume fraction. For $\phi = 0.02$ (Figure 4.1(a)), it can be seen that the heights of the peaks increase with ligand coverage, therefore implying that there is a higher probability of finding an NP at a certain distance from another NP. The first peak in both cases is very high, signifying that we have a very high probability of finding one NP close to a second NP. As $r$ increases, we see that there are a couple of distinct peaks and $g(r)$ levels off to 1. This implies that the structures are those of fluids, which is further backed up by the static structure factors in Figure 4.2(a) where the main peak heights are well below a 2.85. Because the heights of the first peaks are large and near contact of the NCs, it is likely that the NP dispersions assembled into amorphous clusters. The snapshots of the NP dispersions seemed to verify this conclusion. The narrower peaks in Figure 4.1(a) where $v = 0.08$ Å$^{-2}$ suggest that the clusters were better defined at higher values of ligand coverage.

In Figure 4.1(b), we see $g(r)$ for the cases where the ligand coverage is $\upsilon = 0.03$ Å$^{-2}$ and $\upsilon = 0.0465$ Å$^{-2}$ and the hard core volume fraction is $\phi = 0.26$. The $g(r)$ plot for $\upsilon = 0.03$ Å$^{-2}$ looks like that of a fluid, and we can see that the height of the peak closest to contact is very low compared to that for the $\upsilon = 0.03$ Å$^{-2}$ case when $\phi = 0.02$ in Figure 4.1(a). Because of this low peak height, the structure may have been that of a disordered fluid (no or very little evident short-range or long-range order) rather than an amorphous cluster. In the case where $\upsilon = 0.0465$ Å$^{-2}$, $g(r)$ in Figure 4.1(b) seems to imply that the structure is that of an FCC superlattice. This is because the peaks that are present in $g(r)$ are well-defined, and a comparison with the theoretical $g(r)$ for an FCC array showed that most of the positions of the peaks for the computational $g(r)$ matched with the positions of the peaks for the theoretical $g(r)$. When we look at Figure 4.2(b), however, we see that the height of the main peak for the $\upsilon = 0.03$ Å$^{-2}$ case is slightly higher than that for the $\upsilon = 0.0465$ Å$^{-2}$ case, and the peaks (including the first peak) have also split into two. The heights of the main peaks suggest that both structures are superlattices, and after considering all aspects of the plots, we classified both structures as superlattices.

The reader may have noticed that $g(r)$ in Figure 4.1(c) ($\phi = 0.29$) when $\upsilon = 0.02$ Å$^{-2}$ is very noisy and looks like that of a fluid, with a sharp line at $r = 1$. The sharp line at $r = 1$ is actually something that appears in every $g(r)$ for $\upsilon = 0.02$ Å$^{-2}$ that we obtained, regardless of the value of $\phi$. The noise may signify that there are enhanced fluctuations near the freezing transition, but further analyses should be conducted to verify this hypothesis. One may argue that $g(r)$ could be accurately describing a fluid phase, but the peak at $r = 1$ is too narrow for the curve to be that of an amorphous cluster and too high for the curve to be that of a disordered fluid. Indeed, even $S(q)$ in Figure 4.2(c) looks like that of a crystalline array in shape; however, the peak heights are very low, and if we were to use the Hansen-Verlet Freezing Criterion, the structure would be classified as a disordered fluid. We see in Figure 4.1(c) that $g(r)$ for $\upsilon = 0.03$ Å$^{-2}$ looks like that of a crystalline array, which is something that is backed up by $S(q)$ in Figure 4.2(c).

Finally, in Figure 4.1(d), the ligand coverage was kept fixed at $\upsilon = 0.0465$ Å$^{-2}$ and $\phi$ was varied across four simulations. Here, we see that as $\phi$ increases, the structure changes from that of an amorphous cluster to a disordered fluid to a superlattice, as expected. The progression from fluid to superlattice can also be seen in Figure 4.2(d), where the height of the main peak increases with increasing $\phi$ and the single peaks split into multiple peaks.

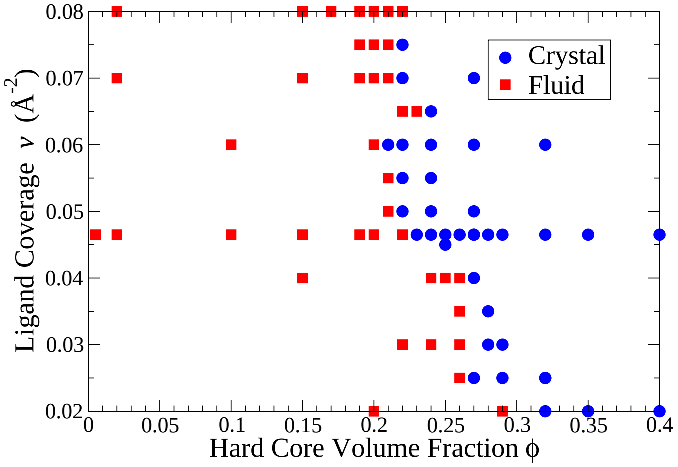

Figure 4.3. Phase diagram of ligand coverage vs. average volume fraction for AuNP dispersions. "Fluid" here means disordered fluid or amorphous cluster. "Crystal" here means superlattice.

Figure 4.3 shows final structures for AuNP dispersions given various average volume fraction and ligand coverage values. Note that the volume fractions shown are those of the hard cores, and the effective nanoparticle volume fractions would be larger than the hard core volume fractions due to the grafted ligands. It can be seen that at low $\phi$, the structures were those of fluids, regardless of the value of $v$. According to our snapshots and our plots in Figures 4.1 and 4.2, however, $\phi = 0.02$ was capable of producing what looked like amorphous clusters rather than disordered fluids. For points closer to the phase boundary, we obtained what looked like disordered fluids instead of amorphous clusters. Around $\phi \in [0.20, 0.30]$, we see the expected trend that an increase in $v$ can lead to a final structure of a superlattice instead of a disordered fluid. Starting at $\phi << 1$, increasing $\phi$ first leads to a disordered fluid phase, as the strength of attraction between NPs increases with $\phi$. However, the final structures tend to become closer to superlattices instead of amorphous clusters given the large initial attraction between NPs, whereas the NPs tend to form amorphous clusters at $\phi << 1$ due to the low attraction between NPs which are not very close together. Note that as $v$ increases near $\phi = 0.25$, the phase boundary is not exactly vertical. This could signify that at high ligand coverage, there was not enough room for ligands to compress between two NCs, implying that steric repulsion dominated and caused the NP dispersions to form a fluid instead of a stable superlattice.

Some points in the phase diagram, such as ($\phi = 0.23$, $v = 0.065$ Å$^{-2}$), are right on the boundary between being classified as crystalline arrays and fluids. An example of the corresponding $g(r)$ and $S(q)$ plots can be seen in Figure 4.1(b) and Figure 4.2(b). In order to distinguish between the two, we continued to analyze the radial distribution functions and static structure factors by comparing our data with a theoretical $g(r)$ and $S(q)$ for an FCC lattice, but we also sometimes ran simulations for a longer period of time (5 or 10 times as long as usual) to check if the system would dissolve into a disordered fluid. The results from the longer simulations are the ones that are included in Figure 4.3.

### 4.1.1. Comparison with the Literature

Khan et al. [10] considered the cases where $\phi = 0.005$ and $\phi \approx 0.02$ using Brownian dynamics with nearly $10^5$ nanoparticles, whereas we considered $N = 500$ NPs. Although the authors of the publication computed the dynamic structure factor (a function that models the order of a system and is proportional to the intensity of scattered light, but can change with time) rather than the radial distribution function and static structure factor in their simulations, we were still able to qualitatively compare the morphologies of the nanoparticle dispersions that we simulated with the morphologies that they obtained by using the snapshots of the NP dispersions and further characterizing our morphologies using the plots for $g(r)$ and $S(q)$.

Figure 1.1 shows a snapshot of an AuNP dispersion from Khan et al. [10] where $\phi = 0.005$. It can be seen that the Brownian dynamics simulations yielded amorphous cluster formations, which seemed to grow in size as the simulations progressed [10]. In Figure 4.3, both of the results that we compared with Khan et al. [10] were classified as fluids. Based on the snapshots that we generated (such as in Figure 4.4) and the analysis given above for the $\phi = 0.02$ case (see Figures 4.1(a) and 4.2(a)), we believe that our results (which were generated using Monte Carlo methods but using the same effective pair potential and parameters as Khan et al. [10]) are consistent with those previously obtained.

Figure 4.5 shows $g(r)$ and $S(q)$ for the case where $\phi = 0.005$ and $v = 0.0465$ Å$^{-2}$. It can be seen once again that the height of the first peak in $g(r)$ is very large compared to, say, those in Figure 4.1(b). Because most of the peaks are poorly defined, we conclude that the structure is possibly an amorphous cluster. This conclusion is backed up further by the plot for $S(q)$, which looks similar to those shown in Figure 4.2(a) in that the structure is not a crystalline array. The

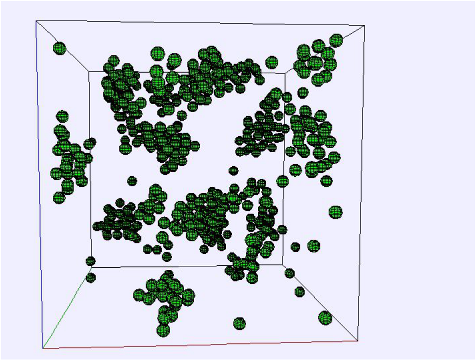

Figure 4.4. The final structure for the Monte Carlo simulation where $\phi = 0.02$ (approximately equal to the volume fraction studied in the literature) and $v = 0.0465$ Å$^{-2}$. There was not a significant change in the structure, so we concluded that the nanoparticle dispersion had reached the metastable state of an amorphous cluster.

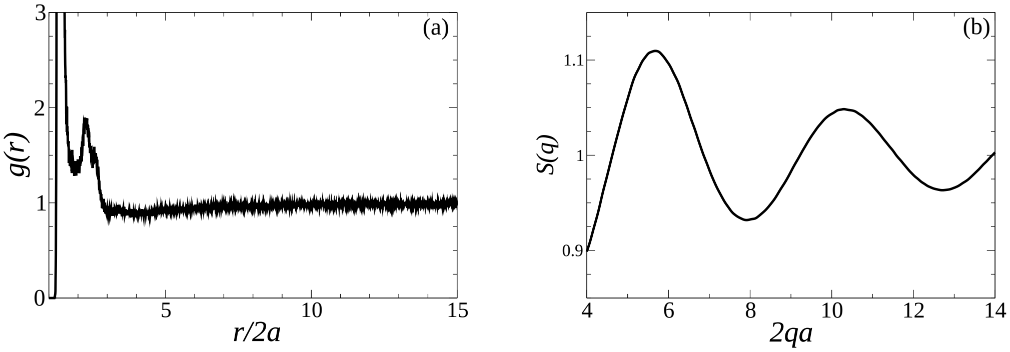

Figure 4.5. (a) $g(r)$ and (b) $S(q)$ from a Monte Carlo simulation of an AuNP dispersion with $\phi = 0.005$ and $v = 0.0465$ Å$^{-2}$.

snapshots were actually not very helpful with comparing to the results in the literature because we only used 500 NPs in a very large box (yielding the low volume fraction). However, the particles still seemed to form clusters in the snapshots. In conclusion, the results from our Monte Carlo simulations seem to qualitatively match those presented in Khan et al. [10], which were obtained using Brownian dynamics methods.

## 4.2. Silver Nanocrystals: Equilibrium Structures

For my senior project in Spring 2017 [74], we attempted to describe the self-assembly of stable superlattices for silver nanoparticles with oleylamine ligands in toluene using Equation (3.36) without considering depletion interactions at all. However, due to the lack of a potential well in our effective pair potential it seemed that we had failed to do so, and we questioned whether the model we were using was accurate for nanoparticle suspensions where the ligands were adsorbed rather than grafted to the surface of the NCs. We eventually turned to considering depletion interactions, though we just barely scratched the surface of exploring the effects of the depletion interactions on the effective pair potential and structures of the nanoparticle dispersions. For this thesis, we explored the effect of adding depletion interactions to the model that we used for AuNP dispersions in greater detail and report the results here.

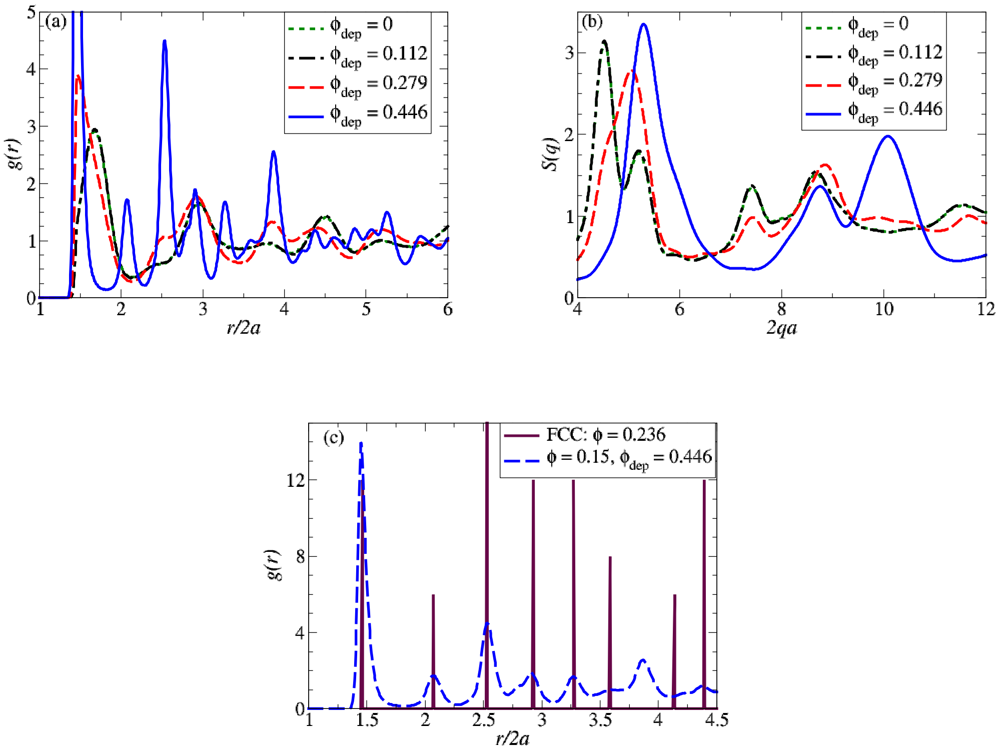

Figure 4.6. (a) $g(r)$ for silver nanoparticle dispersions where the ligand coverage is fixed at $\upsilon = 0.0465$ Å$^{-2}$ (about 41% surface coverage) and the initial NC volume fraction is $\phi = 0.15$. (b) $S(q)$ for the runs shown in panel (a). (c) A comparison between $g(r)$ for the run where $\phi = 0.15$ and $\phi_{\text{dep}} = 0.446$ and a perfect FCC lattice where $\phi = 0.236$.

Figure 4.6(a) shows four radial distribution functions that we obtained from our MC simulations when the ligand coverage was fixed at $\upsilon = 0.0465$ Å$^{-2}$ and the initial NC volume fraction was $\phi = 0.15$, for varying free ligand concentration values, and Figure 4.6(b) shows the corresponding static structure factors. There is barely any difference in $g(r)$ and $S(q)$ between the two simulations where there is no free ligand in the solution and where some free ligand (filling less than 15% of the system volume) exists. This is likely because there is not a significant potential well in the effective pair potential with a free ligand volume fraction of $\phi_{\text{dep}} = 0.11$: the potential well has a depth of only about 1 $k_BT$ and the minimum appears when two nanoparticles are at a center-center separation of about 105 Å (1.75 particle diameters), implying that the nanoparticles would not attract towards one another (if the initial NC volume fraction is $\phi = 0.15$) because the free energy would then increase. However, as the free ligand concentration increases, the equilibrium structures seem to show more order in the sense that more peaks are visible in $g(r)$ when the free ligand volume fraction rose from 0.112 to 0.279 (2.5 times as much free ligand per unit volume). When the free ligand volume fraction rises to 0.446, the equilibrium structure turned into that of an FCC lattice, as shown more explicitly in Figure 4.6(c) (we will discuss this plot in more detail below). It was a bit surprising to see that the height of the main peak in $S(q)$ decreased when the free ligand volume fraction rose from 0.112 to 0.279, as seen in Figure 4.6(b). This may imply that the equilibrium structure changed from a disordered fluid (for $\phi_{\text{dep}} = 0$ and $\phi_{\text{dep}} = 0.112$) to an amorphous cluster (for $\phi_{\text{dep}} = 0.279$) to an superlattice ($\phi_{\text{dep}} = 0.446$). In conclusion, the results shown in Figure 4.6 show that depletion interactions are important in aiding the self-assembly of silver nanoparticles into equilibrium superlattices.

Figure 4.7(a) shows the dependence of the equilibrium structures on the free ligand concentration and the ligand coverage of the nanoparticles, given an initial NC volume fraction of $\phi = 0.22$. We expected that for lower values of ligand coverage, the NPs would start to aggregate due to there being very little stabilization against the van der Waals and (depending on the free ligand concentration) depletion interactions. It was indeed observed that the initial FCC lattice "melted", and we believe that our prediction was correct and that aggregation was starting to occur when $v = 0.005$ Å$^{-2}$ because the energy per particle was on the order of $-10^3$ $k_BT$ at the end of the simulations.

In addition, we expected that the equilibrium structures of the nanoparticle dispersions would not be those of superlattices at low free ligand concentration (less than 2.16 $(2a)^{-3}$, or a free ligand volume fraction of about $\phi_{\text{dep}} = 0.11$ and lower). However, for all values of free ligand concentration that were tested where $v > 0.01$ Å$^{-2}$ (about 9% surface coverage), we observed equilibrium superlattices. One reason as to why these equilibrium structures were FCC lattices is because an initial NC volume fraction of $\phi = 0.22$ places the nearest neighbors of the FCC lattice

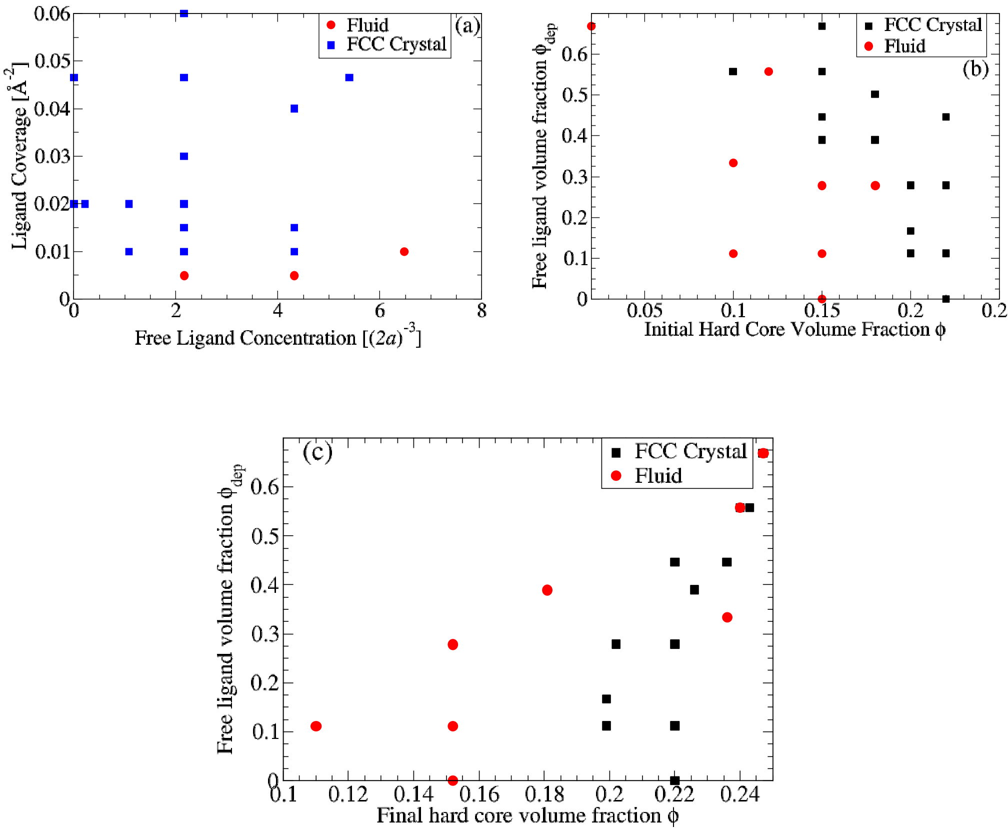

Figure 4.7. Three different phase diagrams for silver nanoparticle dispersions, showing (a) the final structures of simulations given an initial NC volume fraction of $\phi = 0.22$ where the free ligand concentration and ligand coverage could vary; (b) the final structures of simulations where the ligand coverage was fixed at $\upsilon = 0.0465$ Å$^{-2}$; and (c) the final structures of simulations where the ligand coverage was fixed at $\upsilon = 0.0465$ Å$^{-2}$. Two overlapping points means that the final NC volume fraction was obtained by one system in an equilibrium fluid and one system in an equilibrium crystal, implying that phase coexistence may appear in larger systems.

at a center-center distance of about 8.7 nm. When there is no free ligand in the solution and the surfaces of the NCs are about 41% covered in ligands, Figure 3.10(a) shows that two nanoparticles undergo repulsion whenever their ligand layers are overlapping (as in the case of $\phi = 0.22$), and if a trial displacement were to move one nanoparticle closer to any other nanoparticle, there would be a very large increase in the total free energy of the system and therefore the trial displacements would have a very low probability of being accepted. Therefore, we conclude that an initial NC volume fraction of $\phi = 0.22$ is past the freezing transition when more than about 10% of the NC surfaces are covered in ligands, and therefore the equilibrium structures will be superlattices. Similarly, Figure 3.10(d) implies that when free ligand *is* present, the center-center distance between two nanoparticles coincides with the minimum of the pair potential, and any trial displacement would again increase the free energy per particle. In connection to Figure 1.2, Figure 4.7(a) seems to suggest that the experimentally observed superlattice started out at a lower NC volume fraction, as the observed superlattice had a volume fraction of $\phi = 0.22$, and depletion interactions were important in the self-assembly of this superlattice.

In our simulations, it was found that it may have actually been possible for the nanoparticles to self-assemble into a superlattice of higher volume fraction as the solvent was evaporated if the nanoparticle dispersion started off in a superlattice with an NC volume fraction of, say, $\phi = 0.15$. Indeed, Figures 4.7(b) and (c) (which show phase diagrams for simulations where the free ligand concentration and the initial NC volume fraction were varied) seem to suggest that free ligands and depletion interactions are still important in the self-assembly of the silver nanoparticle dispersions into equilibrium superlattices, given initial volume fractions below about $\phi = 0.22$.

For an initial NC volume fraction of $\phi = 0.15$ and no free ligand, $g(r)$ and $S(q)$ (Figure 4.6(a) and (b)) implied that the equilibrium structure was that of a fluid. For similar reasons to those given for Figure 4.7(a), it was expected that the nanoparticle dispersion would "melt" from an initial FCC lattice into a fluid when there was very little or no free ligand in the solution. However, when about 44% of the volume of the system was occupied by free ligands, the nanoparticle dispersions actually formed an equilibrium FCC lattice where the hard core volume fraction was about $\phi = 0.236$ (see Figure 4.6(c)). This signifies that the nanoparticles moved closer to one another in order to minimize the free energy, eventually reaching a lattice where the nearest neighbor center-center distance was about 87 Å. We are currently running simulations where the nanoparticles start in different initial configurations (with all other parameters fixed) to determine whether this equilibrium "compressed FCC lattice" is dependent on the initial configuration. A comparison between Figures 4.7(b) and 4.7(c) (the latter shows the final NC volume fraction, which was calculated by comparing $g(r)$ to that of a perfect FCC lattice, instead of the initial NC volume fraction) shows that the compressed FCC lattice is not unique to the system where $\phi = 0.15$ and $\phi_{\text{dep}} = 0.446$, but rather that it also appears given a higher free ligand concentration for $\phi = 0.15$ and for runs where the initial NC volume fraction is $\phi = 0.18$ as well.

In summary, depletion interactions could be the correct approach to describe the self-assembly observed in the experiments. However, as stated in Section 3.2.2, we have not yet verified that our bounds for the depletion interactions are indeed the correct ones or that depletion interactions work to describe the self-assembly of NP dispersions with adsorbed ligands in all cases. Our results in the following section have set the stage for our hypotheses to be tested.

### 4.3. Molecular Dynamics Simulations: Effective Interactions

Unless otherwise stated, all results are for chains of length $8\sigma$ (8 beads per chain) with 64 chains per plate. We attempted to fit the theoretical prediction to our simulation results by varying $\chi > 0$ in the theory, as $\chi$ was unspecified in our simulations. In general, we qualitatively estimated what a good fit would be between the simulation and theory by taking into account the values of pressure at each wall separation distance from the simulations. We used $\epsilon = 1$, $\sigma = 1$, and $T^* = 3.0 k_B T/\epsilon$. The parameters from Equations (3.1) and (3.2) were $K_1 = 100\epsilon/\sigma^2$ and $s_0 = 1\sigma$, and $K_2 = 10\epsilon$ and $\theta_0 = 120^{\circ}$, respectively. Future work could include determining the dependence of effective pressures on $\theta_0$, as this parameter determines how rigid the ligands are.

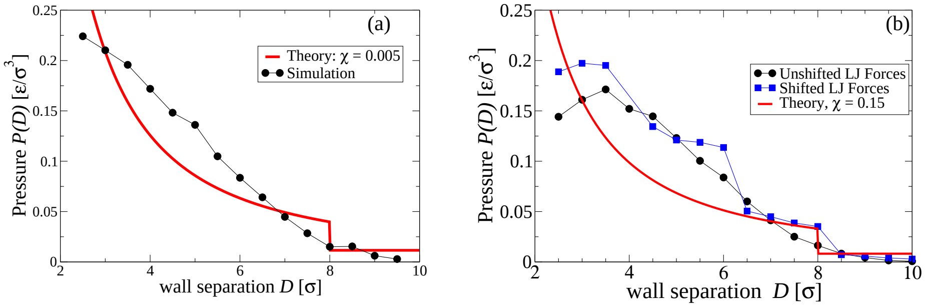

Figure 4.8. Pressures between two flat, ligated plates (the ligands were initialized on rectangular lattices as shown in Figure 4.9(c)) with $r_c = 2.5\sigma$ where (a) $v$ varied with separation distance of the plates and (b) the LJ force between non-bonded beads was and was not shifted to ensure well-definedness at $r_c$ and $v = 0.022\sigma^{-2}$ for all separation distances ($\chi = 0.15$ in the theoretical curve).

Figure 4.8(a) shows the first results we obtained from our MD simulations. Here, we see that the trend is fairly similar with the theory in that the pressure is almost always increasing as wall separation decreases. If the plates were closer than $D = d$ apart, then the ligands would "fold" over to make sure that the beads did not go outside of the volume enclosed by the two flat plates. An issue with the plot shown in Figure 4.8(a) is that we only specified the number of chains attached to one plate as well as the separation distance between the two plates. This was a big problem: the ligand coverage of the two flat plates varied with the separation distance, even though the ligands were grafted to the two flat plates. This can be seen because the number of ligands on each flat plate remained constant while the area of the two flat plates increased as surface separation decreased. We fixed this issue by allowing the user of the script that initialized the ligands to specify a ligand coverage so that both the area and the number of ligands per flat plate would remain fixed as the wall separation was varied.

As discussed in Section 3.1, the force between two beads was ill-defined at the cutoff distance of the LJ 12-6 potential $r_c$. Figure 4.8(b) shows the change in the effective pressure between two flat plates if we introduce a shift in the LJ forces between non-bonded beads. It can be seen that the difference in the effective pressures between two flat plates was significant when a shift was introduced. This may be due to the pair potential changing faster with $r$ when a shift in the force is introduced, as shown in Figure 3.5, so we should expect for there to be a large difference between the two data sets plotted in Figure 4.8(b), and our expectations held true.

### 4.3.1. Effect of Fixed Bead Configurations on Effective Interactions

Figure 4.10(a) shows the pressures for different configurations of the ligand chains: snapshots of the different lattices are shown in Figure 4.9(a), (c), and (e), and the different names for the lattices refer to the layout of the blue beads, which represent the fixed beads. The data points represent the mean values of the pressure when the plates were separated by a distance $D$ over three independent runs, and the error bars show one standard deviation away from the means. While the plot of the pressure for the square lattice is clearly different from those of the rectangular and hexagonal lattices, the differences between the profiles for the rectangular and hexagonal lattices are very small. This could be because the hexagonal lattice *is* essentially the rectangular lattice, except half of the ligands are translated in two directions and the entire lattice is more closely packed in the $x$-direction than the rectangular lattice.

In addition, Figure 4.10(b) shows the effective pressures given two different square lattices: the one without the label “no offset” refers to the lattice shown in Figure 4.9(a), whereas the data with the label “no offset” were generated for a configuration in which the square lattices on opposing walls were not shifted (i.e., the fixed beads were only separated in the $z$-direction, but

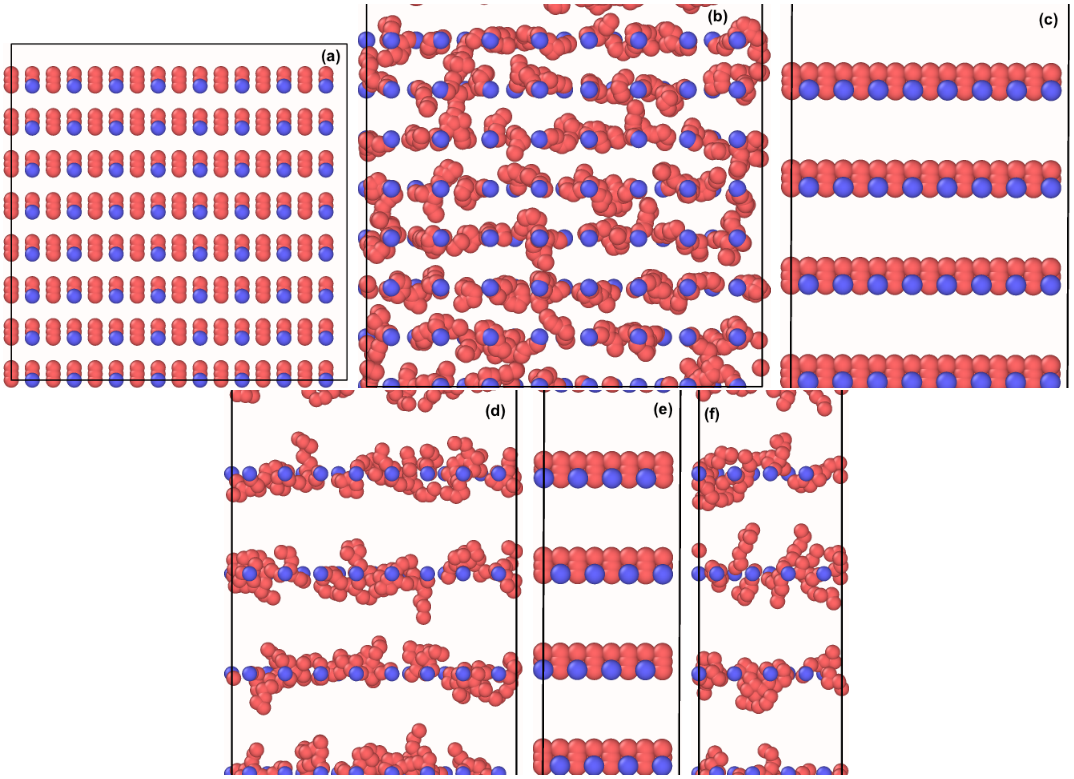

Figure 4.9. Different initial and final configurations of the ligands on the two flat plates, with $v = 0.0222\sigma^{-2}$ and $r_c = D$. We used 1024 beads in our systems. It can be seen that the ligand layers are offset between the two walls, ensuring that there is no overlap of the ligands. (a) Ligands initialized on square lattices. (b) The final configuration of the ligands on square lattices. (c) Ligands initialized on rectangular lattices. (d) The final configuration of the ligands on rectangular lattices. (e) Ligands initialized on hexagonal lattices. (f) The final configuration of the ligands on hexagonal lattices.

not in the $x$- or $y$-directions. Note that the minimum wall separation is $D = 5.5\sigma$ if the lattices on the two flat plates are not offset from each other: this is because the ligands from one wall would overlap or bond with the ligands from the second wall at shorter wall separations.

One can see that there is not a large difference in the pressure between two flat plates regardless of if the offset is present or not. This may be because the dimensions of the flat plates are unchanging when we introduce an offset in the square lattices from opposing plates. On average, one might expect for there to be more beads closer to any given bead. However, it seems that there is cancellation between most added repulsive terms and most added attractive terms for the beads that are within the cutoff range.

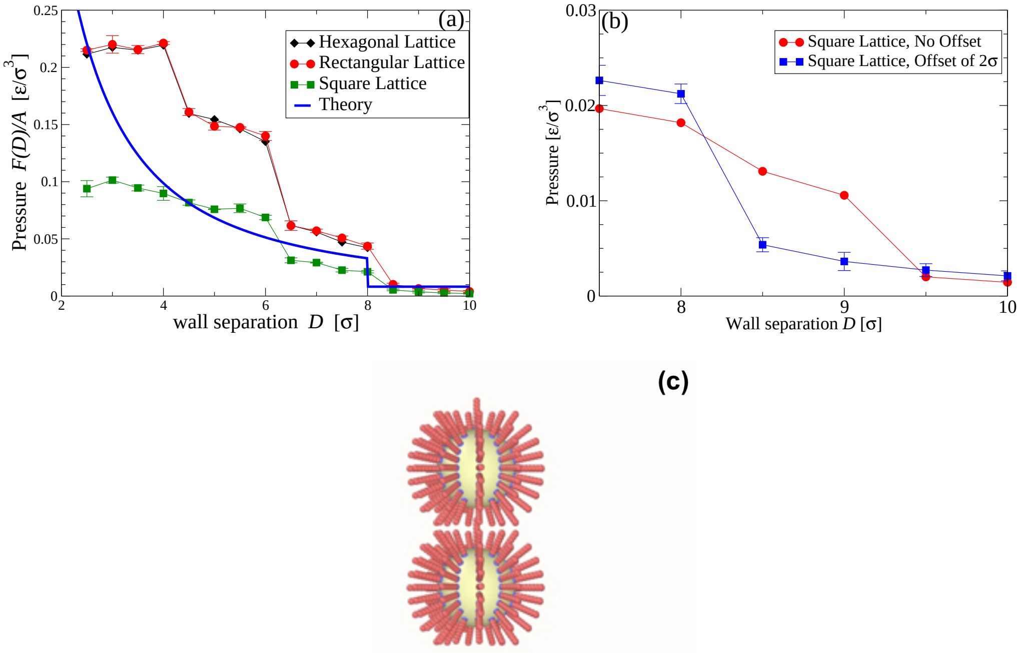

Figure 4.10. The effects of the initial configurations of the ligands (Figure 4.9(a), (c), and (e)) on the effective pressure between two walls, with $v = 0.0222\sigma^{-2}$ and $r_c = D$. The solid line shows the theoretical prediction with $\chi = 0.15$, set to fit the data given square lattices. (a) The pressure as a function of separation distance between the two flat, ligated plates for the configurations shown in Figure 4.9 as well as a theoretical prediction from Evans et al. [24] and Smitham et al. [25]. (b) The effective pressure for square lattices that are offset from each other and square lattices that are not offset from each other. (c) Two spherical, ligated surfaces generated using our methods.

Figure 4.10(c) shows the initial configuration of the beads in a spherical lattice. As stated earlier, we have not obtained results for the case of two spherical, ligated surfaces (representing two nanoparticles).

### 4.3.2. Effect of Varying Temperature on the Effective Interactions

Finally, Figure 4.11 shows comparison between theory and simulation for varying the temperature of the system. Theoretically, Equation (3.51) predicts that if the temperature of our system is lowered, then $\chi$ should increase and therefore the solvent quality would decrease, causing the forces between two ligated surfaces to become more attractive. This is shown graphically in Figure 4.11 by the solid and dashed lines. However, our simulations showed that the opposite happened: a decrease in temperature implied a more repulsive force between two flat plates. This result contradicts the results predicted by the theoretical pair potential that we are using. Changing the temperature does not change the LJ 12-6 potential, but the average velocity of the beads

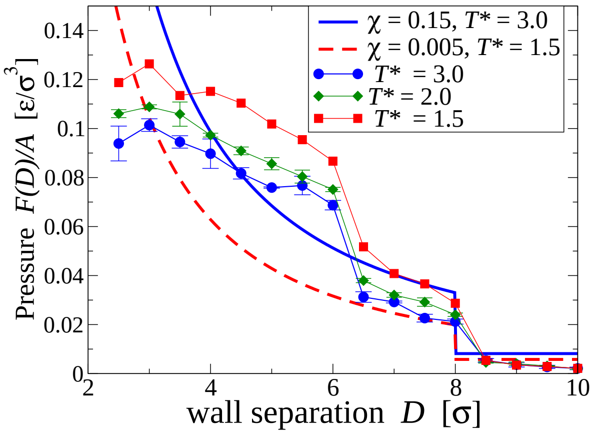

Figure 4.11. The effective pressures between two flat, ligated plates (square lattices with an offset of $2\sigma$ from each other) with $r_c = D$ and $v = 0.0222\sigma^{-2}$ in which we studied the dependence of the pressure on the temperature of the system, and comparison to the trend observed in the theory. Note that the solid blue line is the theoretical prediction at the higher temperature, while the dashed line shows that the effective forces become more attractive as temperature decreases in the theory.

should be lower at a lower temperature. This means that the effective forces between two walls should be, at most, equal to the forces between two walls at a higher temperature. This implies that the theory indeed predicts an opposite trend from our results. The dashed line shows the theoretical prediction for $\chi = 0.005$, which was chosen in an attempt to match with the results for the simulations where $T^* = 1.5$. However, it can be seen that this low of a value for $\chi$ does not yield a good fit for the simulation data. On the other hand, $\chi = 0.15$ (the solid line) seems to yield a good fit for the data obtained for the simulations where $T^* = 3.0$.

One possible explanation for why our simulation results do not match with the theory is that the theory assumed a continuous, uniform step distribution of beads for ligands of length $d$; however, the ligands are much more explicit in our simulations than they are in the theory. In addition, we do not calculate the segment densities from our simulations. However, the segment density at the final timestep of the simulation could be calculated in order to compare with the approximation used in the theory. It was shown in Smitham et al. [25] that the theoretical prediction that we are comparing with actually overestimated experimental results for $D < 5$ nm. While this would not explain the increasing pressure with decreasing temperature, it may explain the quantitative difference between the theory and our simulations.

Another possible explanation for the inconsistency of our results may be that our ligand chains are too short. Chains with 8 beads are considered to be short polymers, and therefore Flory theory may not apply to our chains. Longer chains could be simulated in the future, as this may determine whether or not Flory theory can apply to our system. Overall, however, there are still many simulations and analyses that need to be completed in order to understand the effective interactions between two flat, ligated plates.

# 5. SUMMARY AND OUTLOOK

## 5.1. Summary

Nanoparticles have many interesting properties due to their size, which is between those of molecular and bulk structures. These properties have allowed for many useful practical applications, as nanoparticles have been utilized in drug delivery, photovoltaic cells, and lotions and shampoo. However, despite all of the research that has been conducted, the self-assembly of nanoparticle dispersions into equilibrium superlattices is not something that is well understood. There have been theoretical attempts to describe the ligand-solvent interactions and the ligand-ligand interactions, as well as computational and experimental studies to model the self-assembly of gold nanoparticles dispersions into equilibrium superlattices and the clustering of gold nanoparticles. Our collaborators (Samuel Brown and Professor Erik Hobbie) have also obtained stable superlattices of silver nanoparticle dispersions in the laboratory, though the cause of this self-assembly is not completely understood.

In this thesis, we have presented the Monte Carlo methods that we developed for determining the equilibrium structures for self-assembling nanoparticle dispersions. These methods were used to extend the work found in the literature by characterizing the equilibrium structures of nanoparticle dispersions consisting of gold nanocrystals covered by dodecanethiol ligands, immersed in toluene. In addition, we characterized the equilibrium structures of nanoparticle dispersions consisting of silver nanocrystals covered by oleylamine ligands in toluene by considering depletion interactions caused by free ligand present in the solution and compared our results with those obtained in the laboratory. The model that we used for the gold nanoparticle dispersions was only able to describe the self-assembly of silver nanoparticle dispersions at volume fractions above about $\phi = 0.22$ due to the effective volume fraction being high enough that the nanoparticles could not break out of their initial lattice. However, by considering depletion interactions we were able to observe the self-assembly of nanoparticle dispersions into equilibrium superlattices at an initial NC volume fraction of $\phi = 0.15$. Motivated by our results for silver nanoparticle dispersions, we developed molecular dynamics methods and used LAMMPS to model the ligands using a bead-spring model. These ligands were attached to two flat plates, though we also developed a model for ligands that are chemically bonded to two spherical surfaces.

Our work is very important because it allows for better characterization of systems that have already been studied extensively (gold nanoparticle dispersions) and it allows for the description of the self-assembly of silver nanoparticle dispersions, for which equilibrium superlattices were obtained in the laboratory recently. In addition, our methods can easily be extended to model other systems such as silicon nanoparticle dispersions, therefore helping to guide the fabrication of superlattices and potentially improving the usability of nanoparticles for practical applications. In particular, understanding the self-assembly of silicon nanoparticle dispersions could help to increase the efficiency of photovoltaic cells if the cells are coated by a thin film of silicon nanoparticle dispersions.

### 5.2. Conclusions

As the average volume fraction increased, the gold nanoparticle dispersions formed crystalline arrays instead of fluids, and as the ligand coverage increased, we saw that the nanoparticle dispersions crystallized at lower average volume fractions before this threshold increased once more at high coverage, possibly due to the inability of the ligands to compress any further. In the case of silver nanoparticle dispersions, the depletion interactions allowed for a potential well of several $k_BT$ to appear, and this caused the nanoparticle dispersions to self-assemble into equilibrium superlattices with final NC volume fractions higher than the initial NC volume fractions. We obtained results consistent with experimental results when we considered depletion interactions, although there is still uncertainty in the interplay between depletion and steric interactions.

In the case of two flat, ligated plates, we found that the results from our molecular dynamics simulations qualitatively matched the theory for a “good solvent” in that the pressure between two plates was positive and increased as the flat plates drew closer together, therefore indicating that the force between the two plates was repulsive and increasing in magnitude as wall separation decreased. As expected, we found that the initial configuration of the ligands affects the effective pressures: when we considered the ligands to be on square lattices, we obtained smaller pressures than we obtained for rectangular and hexagonal lattices. However, lowering the temperature of our system by a factor of 50% corresponded to a more repulsive force than in the case of a higher temperature and therefore a “better solvent”. This is contradictory to the theoretical prediction of our effective pair potential, and one possible explanation could be that our ligands are too short to follow Flory's theory for polymer chains.

### 5.3. Future Work

We are writing journal articles on our work for gold and silver nanoparticle dispersions and plan to finalize and submit these articles for review so they can be published in scientific journals. In the future, the analyses for the molecular dynamics simulations of the ligands on a more molecular scale will need to be completed. This can be accomplished by first considering the case where the ligands are grafted to two spherical surfaces. Next, the cases where the ligands are adsorbed (both in the case of two flat walls and the case of two spherical surfaces) can be considered. The latter case would allow for the accurate characterization of the interplay between the steric repulsion and depletion attraction between silver nanoparticles coated with adsorbed oleylamine ligands in toluene. The analyses of these results could verify that the depletion interactions are contributing to the self-assembly of the silver nanoparticle dispersions as thought or they would pave the way for a new explanation to describe the self-assembly. By using the concept of "pipeline modeling" to simulate many nanoparticles with the effective interactions obtained from the MD simulations, it can be determined whether or not depletion interactions may be an accurate description for self-assembly of the silver nanoparticle dispersions. Finally, additions could be made to our Monte Carlo code so that the sizes of amorphous clusters can be better characterized.

# REFERENCES

[1] R. A. L. Jones. *Soft Condensed Matter*. Oxford University Press, Oxford, United Kingdom, 2002.

[2] O. V. Salata. Applications of nanoparticles in biology and medicine. *Journal of Nanobiotechnology*, 2(3), 2004.

[3] M. Grzelczak, J. Vermant, E. M. Furst, and L. M. Liz-Marzán. Directed self-assembly of nanoparticles. *ACS Nano*, 4(7):3591–3605, 2010.

[4] C. A. S. Batista, R. G. Larson, and N. A. Kotov. Nonadditivity of nanoparticle interactions. *Science*, 350:1242477, 2015.

[5] J. B. Miller, J. M. Harris, and E. K. Hobbie. Purifying colloidal nanoparticles through ultracentrifugation with implications for interfaces and materials. *Langmuir*, 10:7936–7946, 2014.

[6] P. C. Ohara, D. V. Leff, J. R. Heath, and W. M. Gelbart. Crystallization of opals from polydisperse nanoparticles. *Physical Review Letters*, 75(19):3466–3469, 1995.

[7] R. Pool, P. Schapotschnikow, and T. J. H. Vlugt. Solvent effects in the adsorption of alkyl thiols of gold structures: A molecular simulation study. *J. Phys. Chem. C*, 111:10201–10212, 2007.

[8] C. Waltmann, N. Horst, and A. Travesset. Capping ligand vortices as “atomic orbitals” in nanocrystal self-assembly. *ACS Nano*, 11(11):11273–11282, 2017.

[9] H. Yan, S. Cingarapu, K. J. Klabunde, A. Chakrabarti, and C. M. Sorensen. Nucleation of gold nanoparticle superclusters from solution. *Physical Review Letters*, 102:095501, 2009.

[10] S. J. Khan, F. Pierce, C. M. Sorenson, and A. Chakrabarti. Self-Assembly of Ligated Gold Nanoparticles: Phenomenological Modeling and Computer Simulations. *Langmuir*, 25(24):13861–13868, 2009.

[11] N. Zaitseva, Z. R. Dai, F. R. Leon, and D. Krol. Optical properties of cdse superlattices. *J. Am. Chem. Soc.*, 127:10221–10226, 2005.

[12] J. B. Miller, N. Dandu, K. A. Velizhanin, R. J. Anthony, U. R. Kortshagen, D. M. Kroll, S. Kilina, and E. K. Hobbie. Enhanced luminescent stability through particle interactions in silicon nanocrystal aggregates. *ACS Nano*, 9:9772, 2015.

[13] J. B. Miller, A. C. P. Usselman, R. J. Anthony, U. R. Kortshagen, A. J. Wagner, A. R. Denton, and E. K. Hobbie. Phase separation and the 'coffee-ring' effect in polymer-nanocrystal mixtures. *Soft Matter*, 10:1665–1675, 2014.

[14] S. L. Brown, D. J. Vogel, J. B. Miller, T. M. Inerbaev, R. J. Anthony, U. R. Kortshagen, D. S. Kilin, and E. K. Hobbie. Enhancing silicon nanocrystal photoluminescence through temperature and microstructure. *J. Phys. Chem. C*, 120:18909–18916, 2016.

[15] Q. H. Tran, V. Q. Nguyen, and A. Le. Silver nanoparticles: synthesis, properties, toxicology, applications, and perspectives. *Adv. Nat. Sci.: Nanosci. Nanotechnol.*, 4, 2013.

[16] S. K. Verma, A. K. Mukherjee, and I. Sinha. Simulating interactions between nanoparticles in lennard-jones liquids. *Chemical Physics Letters*, 572:85–89, 2013.

[17] H. Goesmann and C. Feldmann. Nanoparticulate functional materials. *Angew. Chem. Int. Ed.*, 49:1362–1395, 2010.

[18] V. G. L. Souza and A. L. Fernando. Nanoparticles in food packaging: Biodegradability and potential migration to food - A review. *Food Packaging and Shelf Life*, 8:63–70, 2016.

[19] W. H. D. Jong and P. J. Borm. Drug delivery and nanoparticles: Applications and hazards. *Int J Nanomedicine*, 3(2):133–149, 2008.

[20] M. Gadogbe, S. M. Ansar, G. He, W. E. Collier, J. Rodriguez, D. Liu, I. Chu, and D. Zhang. Determination of colloidal gold nanoparticle surface areas, concentrations, and sizes through quantitative ligand adsorption. *Analytical and Bioanalytical Chemistry*, 405(1):413–422, 2013.

[21] S. Mourdikoudis and L. M. Liz-Marzán. Oleylamine in nanoparticle synthesis. *Chem. Mater.*, 25:1465–1476, 2013.

[22] N. Goubet, J. Richardi, P. A. Albouy, and M. P. Pileni. How to Predict the Growth Mechanism of Supracrystals from Gold Nanocrystals. *The Journal of Physical Chemistry Letters*, 2:417–422, 2011.

[23] C. Y. Lau, H. Duan, F. Wang, C. B. He, H. Y. Low, and J. K. W. Yang. Enhanced ordering in gold nanoparticles self-assembly through excess free ligands. *Langmuir*, 27:3355–3360, 2011.

[24] R. Evans and D. H. Napper. Steric Stabilization II: A generalization to Fischer's solvency theory. *Kolloid-Z. u. Z. Polymere*, 251:329–336, 1973.

[25] J. B. Smitham, R. Evans, and D. H. Napper. Analytical Theories of the Steric Stabilization of Colloidal Dispersions. *Journal of the Chemical Society, Faraday Transactions 1: Physical Chemistry in Condensed Phases*, 71:285–297, 1974.

[26] R. Evans and D. H. Napper. Steric stabilization i: Comparison of theories with experiment. *Kolloid-Z. u. Z. Polymere*, 251:409–414, 1973.

[27] R. Evans, J. B. Smitham, and D. H. Napper. Theoretical prediction of the elastic contribution to steric stabilization. *Colloid and Polymer Science*, 255:161–167, 1977.

[28] P. Galvin, D. Thompson, K. B. Ryan, A. McCarthy, A. C. Moore, C. S. Burke, M. Dyson, B. D. MacCraith, Y. K. Gun'ko, M. T. Byrne, Y. Volkov, C. Keely, E. Keehan, M. Howe, C. Duffy, and R. MacLoughlin. Nanoparticle-based drug delivery: case studies for cancer and cardiovascular applications. *Cell. Mol. Life Sci.*, 69:389–404, 2012.

[29] C. Burda, X. Chen, R. Narayanan, and M. A. El-Sayed. Chemistry and properties of nanocrystals of different shapes. *Chem. Rev.*, 105:1025–1102, 2005.

[30] M. V. Kovalenko, B. Spokoyny, J. Lee, M. Scheele, A. Weber, S. Perera, D. Landry, and D. V. Talapin. Semiconductor nanocrystals functionalized with antimony telluride zintl ions for nanostructured thermoelectrics. *J. Am. Chem. Soc.*, 132:6686–6695, 2010.

[31] A. Widmer-Cooper and P. L. Geissler. Ligand–mediated interactions between nanoscale surfaces depend sensitively and nonlinearly on temperature, facet dimensions, and ligand coverage. *ACS Nano*, 10:1877–1887, 2016.

[32] P. D. Howes, R. Chandrawati, and M. M. Stevens. Colloidal nanoparticles as advanced biological sensors. *Science*, 346(1247390), 2014.

[33] J. Israelachvili. *Intermolecular and Surface Forces.* Academic Press, second edition.

[34] K. Ouhenia-Ouadahi, A. Andrieux-Ledier, J. Richardi, P. Albouy, P. Beaunier, P. Sutter, E. Sutter, and A. Courty. Tuning the growth mode of 3d silver nanocrystal superlattices by triphenylphosphine. *Chemistry of Materials*, 28:4380–4389, 2016.

[35] F. Schreiber. Self-assembled monolayers: from 'simple' model systems to biofunctionalized interfaces. *Journal of Physics: Condensed Matter*, 16(R881), 2004.

[36] S. Asakura and F. Oosawa. On interaction between two bodies immersed in a solution of macromolecules. *The Journal of Chemical Physics*, 22:1255, 1954.

[37] F. Oosawa and S. Asakura. Surface tension of high-polymer solutions. *The Journal of Chemical Physics*, 22:1255, 1954.

[38] J. Henzie, M. Grünwald, A. Widmer-Cooper, P. L. Geissler, and P. Yang. Self–assembly of uniform polyhedral silver nanocrystals into densest packings and exotic superlattices. *Nature Materials*, 11:131–137, 2011.

[39] D. Baranov, A. Fiore, M. van Huis, C. Giannini, A. Falqui, U. Lafont, H. Zandbergen, M. Zanella, R. Ciogolani, and L. Manna. Assembly of colloidal semiconductor nanorods in solution by depletion attraction. *ACS Nano Latters*, 10:743–749, 2010.

[40] Y. Mao, M. E. Cates, and H. N. W. Lekkerkerker. Depletion stabilization by semidilute rods. *Physical Review Letters*, 75(24):4548–4551, 1995.

[41] M. Zanella, G. Bertoni, I. R. Franchini, R. Brescia, D. Baranov, and L. Manna. Assembly of shape-controlled nanocrystals by depletion attraction. *Chem. Commun.*, 47:203–205, 2011.

[42] X. Ye, T. Narayanan, and P. Tong. Neutron Scattering Study of Depletion interactions in a Colloid-Polymer Mixture. *Physics Review Letters*, 76(24):4640–4643, 1996.

[43] H. Wang and A. R. Denton. Effective electrostatic interactions in solutions of polyelectrolyte stars with rigid rodlike arms. *The Journal of Chemical Physics*, 123(244901):1–9, 2005.

[44] A. Yethiraj and C. K. Hall. Monte carlo simulation of polymers confined between flat plates. *Macromolecules*, 23:1865–1872, 1990.

[45] S. J. O'Shea, M. E. Welland, and T. Rayment. An atomic force microscope study of grafted polymers on mica. *Langmuir*, 9:1826–1835, 1993.

[46] D. Goodman, J. N. Kizhakkedathu, and D. E. Brooks. Attractive bridging interactions in dense polymer brushes in good solvent measured by atomic force microscopy. *Langmuir*, 20:2333–2340, 2004.

[47] S. Yamamoto, M. Ejaz, Y. Tsujii, M. Matsumoto, and T. Fukuda. Surface interaction forces of well-defined, high-density polymer brushes studied by atomic force microscopy. 1. effect of chain length. *Macromolecules*, 33:5602–5607, 2000.

[48] S. Yamamoto, M. Ejaz, Y. Tsujii, and T. Fukuda. Surface interaction forces of well-defined, high-density polymer brushe s studied by atomic force microscopy. 2. effect of graft density. *Macromolecules*, 33:5608–5612, 2000.

[49] S. Hajiw, J. Schmitt, M. Impéror-Clerc, and B. Pansu. Solvent-driven interactions between hydrophobically-coated nanoparticles. *Soft Matter*, pages 1–19, 2015.

[50] K. T. Marla and J. C. Meredith. Nanoscale colloids in a freely adsorbing polymer solution: A monte carlo simulation study. *Langmuir*, 20:1501–1510, 2004.

[51] K. T. Marla and J. C. Meredith. Simulation of interaction forces between nanoparticles in the presence of lennard–jones polymers: Freely adsorbing homopolymer modifiers. *Langmuir*, 21:487–497, 2005.

[52] K. T. Marla and J. C. Meredith. Simulation of interaction forces between nanoparticles: End-grafted polymer modifiers. *J. Chem. Theory Comput.*, 2:1624–1631, 2006.

[53] G. Wang, N. D. Eastham, T. J. Aldrich, B. Ma, E. F. Manley, Z. Chen, L. X. Chen, M. Olvera de la Cruz, R. P. H. Chang, F. S. Melkonyan, A. Facchetti, and T. J. Marks. Photoactive blend morphology engineering through systematically tuning aggregation in all-polymer solar cells. *Advanced Energy Materials*, 1702173, 2018.

[54] J. M. H. M. Scheutjens and G. J. Fleer. Interaction between two adsorbed polymer layers. *Macromolecules*, pages 1882–1900, 1985.

[55] S. Plimpton. Fast Parallel Algorithms for Short-Range Molecular Dynamics. *J Comp Phys*, 117:1–19, 1995.

[56] F. Th. Hesselink, A. Vrij, and J. Th. G. Overbeek. On the theory of the stabilization of dispersions by adsorbed macromolecules. ii. interaction between two flat particles. *The Journal of Physical Chemistry*, 75(14):2094–2103, 1971.

[57] P. J. Flory and W. R. Krigbaum. Statistical mechanics of dilute polymer solutions. ii. *The Journal of Chemical Physics*, 18:1086, 1950.

[58] P. C. Painter, J. Graf, and M. M. Coleman. Coal solubility and swelling. 1. solubility parameters for coal and the flory chi parameter. *Energy & Fuel*, 4, 1990.

[59] J. K. Stolarczyk, T. Sainsbury, and D. Fitzmaurice. Evaluation of interactions between functionalised multi-walled carbon nanotubes and ligand-stabilised gold nanoparticles using surface element integration. *J. Computer-Aided Mater. Des.*, 14:151–165, 2007.

[60] R. F. Blanks and J. M. Prausnitz. Thermodynamics of polymer solubility in polar and nonpolar systems. *I and EC Fundamentals*, 3(1), 1964.

[61] R. S. Ruoff, D. S. Tse, R. Malhotra, and D. C. Lorents. Solubility of $c_{60}$ in a variety of solvents. *J. Phys. Chem.*, 97:3379–3383, 1993.

[62] A. Fernández-Nieves, A. Fernández-Barbero, B. Vincent, and F. J. de las Nieves. Reversible Aggregation of Soft Particles. *Langmuir*, 17:1841–1846, 2001.

[63] A. O. Pinchuk. Size-Dependent Hamaker Constant for Silver Nanoparticles. *The Journal of Physical Chemistry C*, 116:20099–20102, 2012.

[64] A. Evelyn Di Mauro, M. Striccoli, N. Depalo, E. Fanizza, L. Cano, C. Ingrosso, A. Agostiano, M. Lucia Curri, and A. Tercjak. Selective Confinement of Oleylamine Capped Au Nanoparticles in Self-Assembled PS-b-PEO Diblock Copolymer Templates. *Soft Matter*, 10:1676–1684, 2014.

[65] A. Loubat, M. Impéror-Clerc, B. Pansu, F. Meneau, B. Raquet, G. Viau, and L. Lacroix. Growth and Self-Assembly of Ultrathin Au Nanowires into Expanded Hexagonal Superlattice Studied by in Situ SAXS. *Langmuir*, 30:4005–4012, 2014.

[66] I. M. Sobol. *A Primer for the Monte Carlo Method.* CRC Press, first edition, 1994.

[67] H. Gould, J. Tobochnik, and W. Christian. *An Introduction to Computer Simulation Methods: Applications to Physical Systems.* Addison-Wesley, third edition, 2006.

[68] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equation of State Calculations by Fast Computing Machines. *The Journal of Chemical Physics*, 21(6):1087–1092, 1953.

[69] D. Frenkel and B. Smit. *Understanding Molecular Simulation: From Algorithms to Applications.* Academic Press, second edition, 2002.

[70] William Humphrey, Andrew Dalke, and Klaus Schulten. VMD – Visual Molecular Dynamics. *Journal of Molecular Graphics*, 14:33–38, 1996.

[71] A. Stukowski. Visualization and analysis of atomistic simulation data with ovito – the open visualization tool. *Modelling Simul. Mater. Sci. Eng.*, 18:015012, 2010.

[72] J. Hansen and I. McDonald. *Theory of Simple Liquids.* Academic Press, London, United Kingdom, third edition, 2008.

[73] J. P. Hansen and L. Verlet. Phase transitions of the lennard-jones system. *Phys. Rev.*, 184, 1969.

[74] V. Shah and A. Denton. Structure and stability of self–assembling nanoparticle dispersions. Technical report, North Dakota State University, 2017.

# APPENDIX. CODE USED FOR THE DESCRIBED WORK

This Appendix contains Java classes that we created for our Monte Carlo simulations and scripts that we created and used for our molecular dynamics simulations, mainly to generate the input for LAMMPS and analyze the results.

**The Java class RDF was used to compute the radial distribution function in our Monte Carlo simulations of nanoparticle dispersions with effective pair interactions. This was developed by Professor Alan Denton, Department of Physics, North Dakota State University.**

```
package org.opensourcephysics.sip.lj;

import java.util.ArrayList;

import org.opensourcephysics.numerics.PBC;

public class RDF {
        private int N;
        private double [] x, y, z;
        private int [] nr;
        private double [] gr;
        private double side;
        private double deltaR = 0.005;
        //private double deltaR = 0.05;
        private int nshell;
        private double maxR;
        private int count = 0;

        public RDF(double x[], double y[], double z[], double side){
                N = x.length;
                this.x = x;
                this.y = y;
                this.z = z;
                this.side = side;
                maxR = side/2;
                nshell = (int) Math.ceil(maxR/deltaR) + 1;
                nr = new int [nshell];
                gr = new double [nshell];
        }

        public void update(){
                count++;
                for(int i = 0; i < N; i++){
                    for(int j = 0; j < N; j++){
                        if(i != j){
                            double rx = PBC.separation(Math.abs(x[i]-x[j
        ]), side);
                            double ry = PBC.separation(Math.abs(y[i]-y[j
        ]), side);
                            double rz = PBC.separation(Math.abs(z[i]-z[j
        ]), side);
                  //double rx = x[i]-x[j];
                  //double ry = y[i]-y[j];
                  //double rz = z[i]-z[j];
                  double r2=rx*rx+ry*ry+rz*rz;
                            if(r2 <= maxR*maxR){
                                double r = Math.sqrt(r2);
                                int n = (int) Math.ceil(r/deltaR);
        // nth histogram bin
                                nr[n]++;
                            }
                        }
                    }
                }
        }

        public double [] calcDistribution(){
                double rho = (double)N/Math.pow(side, 3); // number density
                double norm = 4*Math.PI*rho*Math.pow(deltaR, 3)*N*count; //
        normalization
```

```
        for(int n = 1; n < gr.length; n++){
            double r = deltaR*n;
            if(r <= maxR){
                gr[n] = nr[n] / (norm*n*n);
            }
        }

        double [] grClone = new double[gr.length];
        System.arraycopy(gr, 0, grClone, 0, gr.length);
        return grClone;
    }

    public String distributionData(){
        StringBuffer data = new StringBuffer();
        gr = calcDistribution();
        for(int i =0; i < gr.length; i++){
            data.append(i*deltaR + ” ” + gr[i] + ”\n”);
        }
        String finalData = data.toString();
        return finalData;
    }

    public String nrData(){
        StringBuffer data = new StringBuffer();
        for(int i =0; i < nr.length; i++){
            data.append(i*deltaR + ” ” + nr[i] + ”\n”);
        }
        String finalData = data.toString();
        return finalData;
    }

}
```

**The Java class SSF was used to compute the static structure factor in our Monte Carlo simulations of nanoparticle dispersions with effective pair interactions.**

```
package org.opensourcephysics.sip.lj;


import java.util.ArrayList;


import org.opensourcephysics.numerics.PBC;


public class SSF{
    private int i,j,k;
    private int N; // number of particles
    private double [] x,y,z;
    private double side;
    private int kMax = 5000; // Number of wave vectors computed
    private double Sq, q;
    private double dq = 0.005;
    private double sum;
    private int count = 0;
    private double diameter;
    private double [] Sqterms;
    private double [] Sqvalues;



    public SSF(double x[], double y[], double z[], double side){
        N = x.length;
        this.x = x;
        this.y = y;
        this.z = z;
        this.side = side;
        Sqterms = new double [N];
        sum = 0.00;
        Sqvalues = new double [kMax];
    }

    public void updateSQ(){
        count++;
        for(k=10; k<kMax; k++){
            Sq = 0;
            q = k * dq;
            for(i=0; i<N; ++i){
                for(j=(i+1); j<N; ++j){
                    double rx = x[i]-x[j];
                    double ry = y[i]-y[j];
                    double rz = z[i]-z[j];
                    double r2 = rx*rx+ry*ry+rz*rz;
                    double r = Math.sqrt(r2);
                    Sq += Math.sin(q*r) / (q*r);
                }
            } // End of the i loop
            Sq *= (2. / (double) N);
            Sqvalues[k] += Sq;
        }
    }

    public String outputData(){
        StringBuffer data = new StringBuffer();
        updateSQ();
        for(int k=10; k<kMax; k++){
            Sqvalues[k] /= count;
            Sqvalues[k] += 1;
            data.append(k*dq + ” ” + Sqvalues[k] + ”\n”);
        }
        String finalData = data.toString();
        return finalData;
    }
}
```

**The C++ code below initializes the chains on rectangular lattices on two flat plates (with an offset between lattices on opposing plates), at fixed ligand coverage.**

```
#include <cmath>
#include <fstream>
#include <iostream>
#include <cstdlib>
#include <string>
using namespace std;

/*
 This script generates chains on two flat plates which are parallel to the xy-plane.
     The two plates can have a minimum surface separation of approximately 2.0,
    regardless of the number of segments per chain. The chains will loop around,
    therefore not passing through the walls. ”chains.xyz” is the output file
    containing particle types and coordinates, and the total number of atoms.
 Version 3.0: an extension of Version 2.0 that allows the user to specify a ligand
    coverage value which is fixed over all separation distances. As of now, the user
    should use 2chainsinit.cpp to determine what the maxmimum v should be for the
    system: for 64 chains per wall (assuming 8 rows and 8 columns) with 8 segments
    per chain, the minimum x,y-dimensions must be 32, 90 sigma (respectively).
    Theoretical calculation for arbitary nPerChain and nChain to be determined.
 */

# define nPerChain 8 // Number of monomers per chain
# define nChain 64 // number of chains per wall
# define R 8.25 // surface separation
# define v 0.022222222222 // ligand coverage
# define xMax 32.00000 // maximum x coordinate
# define dx 4.000000 // distance between columns of chains attached to one wall.
    Changeable, but it is recommended to have xMax = nPerChain*dx for easy setting of
     the PBC in the data file
int N = 2*nPerChain*nChain; // number of monomers
float separation = (nChain/v)/xMax; // determines the separation between rows in the
     y-direction
float dy = separation/sqrt(nChain); // determines the distance between rows of
    chains in the y-direction
```

```
int main(){
    ofstream f;
    f.open(”chains.xyz”);
    f << N << endl;
    f << ” linear chains of atoms” << endl;

    int types [N/2];
    float positions[N/2][3];

    float x = 0.000000;
    int i = 0; // keeps track of the monomer index
    for(int l = 0; l < sqrt(nChain); l++){ // loop over one dimension of chains on a
     wall
        float y = 0.000000;
        float yPrev = 0.0000000; // stores the position of the start of the previous
     chain in the column
        for(int j=0; j<sqrt(nChain); j++){ // loop over second dimension of chains
    on a wall
            float z = 0.000000;
            for(int k = 0; k<nPerChain; k++){ // loop over beads in one chain
                if(k==0){
                    yPrev = y; // stores the position of the start of the chain
                }
                positions[i][0] = x;
                positions[i][1] = y;
                positions[i][2] = z;
                if(k % nPerChain == 0){ // fixed atom
                    types[i] = 2;
                }
                else{
                    types[i] = 1;
                }
                if((int(y-yPrev) % 4) == 0){
                    z += 1.0000000;
                }
                else if((int(y-yPrev) % 4) == 2){
                    z -= 1.0000000;
                }
                else{
                    y += 1.0000000;
                }
                if(z>=R){ // next particle will overlap the wall on the right side
                    z -= 1.0000000;
                    y += 1.0000000;
                }
                else if(z<=0.000000){
                    z += 1.0000000;
                    y += 1.0000000;
                }
                i += 1;
            }
            y = yPrev+dy; // determines the y-coordinate of the fixed atom for the
    next chain in the column using the ligand coverage value and the start of the
    previous chain
        }
        x += dx; // start a second column

    }
    for(int i = 0; i<N/2; i++){ // left wall
        f << types[i] << '\t' << positions[i][0] << '\t' << positions[i][1] << '\t'
    << (positions[i][2]-R/2.) << endl;
    }
    for(int i = 0; i<N/2; i++){ // right wall, translate x-positions by 2.0
        f << types[i] << '\t' << (positions[i][0]+2.000000) << '\t' << positions[i
    ][1] << '\t' << (-1.*positions[i][2]+R/2.) << endl;
    }
    f.close();
}
```

**The C++ code below initializes the chains on rectangular lattices on two flat plates (with *no* offset between lattices on opposing plates), at fixed ligand coverage.**

```
#include <cmath>
#include <fstream>
#include <iostream>
#include <cstdlib>
#include <string>
using namespace std;

/*
 This script generates chains on two flat plates which are parallel to the xy−plane.
     The two plates can have a minimum surface separation of approximately 5.5,
    regardless of the number of segments per chain. The chains will loop around,
    therefore not passing through the walls. ”chains.xyz” is the output file
    containing particle types and coordinates, and the total number of atoms.
 Version 4.0: An extension of Version 3.0. This version initializes the chains so
    that there is no offset between the chains on the two flat plates.
 */

# define nPerChain 8 // Number of monomers per chain
# define nChain 64 // number of chains per wall
# define R 10.5 // surface separation
# define v 0.022222222222 // ligand coverage
# define xMax 32.00000 // maximum x coordinate
# define dx 4.000000 // distance between columns of chains attached to one wall.
    Changeable, but it is recommended to have xMax = nPerChain*dx for easy setting of
     the PBC in the data file
int N = 2*nPerChain*nChain; // number of monomers
float separation = (nChain/v)/xMax; // determines the separation between rows in the
     y−direction
float dy = separation/sqrt(nChain); // determines the distance between rows of
    chains in the y−direction

int main(){
    ofstream f;
    f.open(”chains.xyz”);
```

```
f << N << endl;
f << ” linear chains of atoms” << endl;

int types [N/2];
float positions[N/2][3];

float x = 0.000000;
int i = 0; // keeps track of the monomer index
for(int l = 0; l < sqrt(nChain); l++){ // loop over one dimension of chains on a
 wall
    float y = 0.000000;
    float yPrev = 0.0000000; // stores the position of the start of the previous
 chain in the column
    for(int j=0; j<sqrt(nChain); j++){ // loop over second dimension of chains
on a wall
        float z = 0.000000;
        for(int k = 0; k<nPerChain; k++){ // loop over beads in one chain
            if(k==0){
                yPrev = y; // stores the position of the start of the chain
            }
            positions[i][0] = x;
            positions[i][1] = y;
            positions[i][2] = z;
            if(k % nPerChain == 0){ // fixed atom
                types[i] = 2;
            }
            else{
                types[i] = 1;
            }
            if((int(y-yPrev) % 4) == 0){
                z += 1.0000000;
            }
            else if((int(y-yPrev) % 4) == 2){
                z -= 1.0000000;
            }
            else{
                    y += 1.0000000;
                }
                if(z>=((R/2.)-1.5)){ // next particle will overlap the wall on the
    right side
                    z -= 1.0000000;
                    y += 1.0000000;
                }
                else if(z<=0.000000){
                    z += 1.0000000;
                    y += 1.0000000;
                }
                i += 1;
            }
            y = yPrev+dy; // determines the y-coordinate of the fixed atom for the
    next chain in the column using the ligand coverage value and the start of the
    previous chain
        }
        x += dx; // start a second column

    }
    for(int i = 0; i<N/2; i++){ // left wall
        f << types[i] << '\t' << positions[i][0] << '\t' << positions[i][1] << '\t'
    << (positions[i][2]-R/2.) << endl;
    }
    for(int i = 0; i<N/2; i++){ // right wall, translate  x-positions by 2.0
        f << types[i] << '\t' << positions[i][0] << '\t' << positions[i][1] << '\t'
    << (-1.*positions[i][2]+R/2.) << endl;
    }
    f.close();
}
```

**The C++ code below initializes the chains on hexagonal lattices on two flat plates (with an offset between lattices on opposing plates), at fixed ligand coverage.**

```
#include <cmath>
#include <fstream>
#include <iostream>
```

```
#include <cstdlib>
#include <string>
using namespace std;

/*
 This script generates chains on two flat plates which are parallel to the xy-plane.
     The two plates can have a minimum surface separation of approximately 2.0,
    regardless of the number of segments per chain. The chains will loop around,
    therefore not passing through the walls. ”chains.xyz” is the output file
    containing particle types and coordinates, and the total number of atoms.
 Version 5.0: an extension of Version 3.0 that allows the user to initialize the
    chains on a hexagonal lattice. Users should use 2chainsinit.cpp to determine what
     the maxmimum v should be for the system: for 64 chains per wall (assuming 8 rows
     and 8 columns) with 8 segments per chain, the minimum x,y-dimensions must be 32,
     90 sigma (respectively). Theoretical calculation for arbitary nPerChain and
    nChain to be determined.
 */

# define nPerChain 8 // Number of monomers per chain
# define nChain 64 // number of chains per wall
# define R 10. // surface separation
# define v 0.022222222222 // ligand coverage
# define xMax 32.00000 // maximum x coordinate
# define dx 4.000000 // distance between columns of chains attached to one wall.
    Changeable, but it is recommended to have xMax = nPerChain*dx for easy setting of
     the PBC in the data file
int N = 2*nPerChain*nChain; // number of monomers
float separation = (nChain/v)/xMax; // determines the separation between rows in the
     y-direction
float dy = separation/sqrt(nChain); // determines the distance between rows of
    chains in the y-direction

int main(){
    ofstream f;
    f.open(”chains.xyz”);
    f << N << endl;
```

```
f << ” linear chains of atoms” << endl;

int types [N/2];
float positions[N/2][3];

float x = 0.000000;
int i = 0; // keeps track of the monomer index
for(int l = 0; l < int(sqrt(nChain)/2); l++){ // loop over one dimension of
chains on a wall
    float y = 0.000000;
    float yPrev = 0.0000000; // stores the position of the start of the previous
 chain in the column
    for(int j=0; j<sqrt(nChain); j++){ // loop over second dimension of chains
on a wall
        float z = 0.000000;
        for(int k = 0; k<nPerChain; k++){ // loop over beads in one chain
            if(k==0){
                yPrev = y; // stores the position of the start of the chain
            }
            positions[i][0] = x;
            positions[i][1] = y;
            positions[i][2] = z;
            if(k % nPerChain == 0){ // fixed atom
                types[i] = 2;
            }
            else{
                types[i] = 1;
            }
            if((int(y-yPrev) % 4) == 0){
                z += 1.0000000;
            }
            else if((int(y-yPrev) % 4) == 2){
                z -= 1.0000000;
            }
            else{
                y += 1.0000000;
            }
            if(z>=R){ // next particle will overlap the wall on the right side
                z -= 1.0000000;
                y += 1.0000000;
            }
            else if(z<=0.000000){
                z += 1.0000000;
                y += 1.0000000;
            }
            i += 1;
        }
        y = yPrev+2*dy; // determines the y-coordinate of the fixed atom for the
 next chain in the column using the ligand coverage value and the start of the
previous chain
    }
    x += dx; // start a second column
}

for(int i = 0; i<N/4; i++){ // left wall first row
    f << types[i] << '\t' << positions[i][0] << '\t' << positions[i][1] << '\t'
<< (positions[i][2]-R/2.) << endl;
}
for(int i = 0; i<N/4; i++){ // left wall, second row
    f << types[i] << '\t' << (positions[i][0]+dx/2.) << '\t' << (positions[i
][1]+dy) << '\t' << (positions[i][2]-R/2.) << endl;
}
for(int i = 0; i<N/4; i++){ // right wall, translate  x-positions by 2.0, first
row
    f << types[i] << '\t' << (positions[i][0]+dx/2.) << '\t' << positions[i][1]
<< '\t' << (-1.*positions[i][2]+R/2.) << endl;

}
for(int i = 0; i<N/4; i++){ // right wall, translate  x-positions by 2.0, second
 row
    f << types[i] << '\t' << positions[i][0] << '\t' << (positions[i][1]+dy) <<
'\t' << (-1.*positions[i][2]+R/2.) << endl;
```

```
    }
    f.close();
}
```

**The C++ code below initializes the chains on square lattices on two flat plates (with an offset between lattices on opposing plates), at fixed ligand coverage.**

```
#include <cmath>
#include <fstream>
#include <iostream>
#include <cstdlib>
#include <string>
using namespace std;

/*
 This script generates chains on two flat plates which are parallel to the xy-plane.
     The two plates can have a minimum surface separation of approximately 2.0,
    regardless of the number of segments per chain. The chains will loop around,
    therefore not passing through the walls. "chains.xyz" is the output file
    containing particle types and coordinates, and the total number of atoms.
 Version 5.0: an extension of Version 3.0 that allows the user to specify a ligand
    coverage value which is fixed over all separation distances. The chains are
    initialized in a square lattice on one wall, and another (offset) square lattice
    on the second wall.
 */

# define nPerChain 8 // Number of monomers per chain
# define nChain 64 // number of chains per wall
# define R 10. // surface separation
# define v 0.022222222222 // ligand coverage
int N = 2*nPerChain*nChain; // number of monomers
float separation = sqrt(nChain/v); // determines the side length (both x and y)
float ds = separation/sqrt(nChain); // determines the distance between rows of
    chains in the y-direction

int main(){
    cout << "Separation " << separation << endl;
cout << "ds " << ds << endl;
ofstream f;
f.open("chains.xyz");
f << N << endl;
f << " linear chains of atoms" << endl;

int types [N/2];
float positions[N/2][3];

float x = 0.000000;
int i = 0; // keeps track of the monomer index
for(int l = 0; l < sqrt(nChain); l++){ // loop over one dimension of chains on a
 wall
    float y = 0.000000;
    float yPrev = 0.0000000; // stores the position of the start of the previous
 chain in the column
    for(int j=0; j<sqrt(nChain); j++){ // loop over second dimension of chains
on a wall
        float z = 0.000000;
        for(int k = 0; k<nPerChain; k++){ // loop over beads in one chain
            if(k==0){
                yPrev = y; // stores the position of the start of the chain
            }
            positions[i][0] = x;
            positions[i][1] = y;
            positions[i][2] = z;
            if(k % nPerChain == 0){ // fixed atom
                types[i] = 2;
            }
            else{
                types[i] = 1;
            }
            if((int(y-yPrev) % 4) == 0){
                z += 1.0000000;
            }
            else if((int(y-yPrev) % 4) == 2){
                    z -= 1.0000000;
                }
                else{
                    y += 1.0000000;
                }
                if(z>=R){ // next particle will overlap the wall on the right side
                    z -= 1.0000000;
                    y += 1.0000000;
                }
                else if(z<=0.000000){
                    z += 1.0000000;
                    y += 1.0000000;
                }
                i += 1;
            }
            y = yPrev+ds; // determines the y-coordinate of the fixed atom for the
    next chain in the column using the ligand coverage value and the start of the
    previous chain
        }
        x += 2.*ds; // start a second column

    }
    for(int i = 0; i<N/2; i++){ // left wall
        f << types[i] << '\t' << positions[i][0] << '\t' << positions[i][1] << '\t'
    << (positions[i][2]-R/2.) << endl;
    }
    for(int i = 0; i<N/2; i++){ // right wall, translate x-positions by 2.0
        f << types[i] << '\t' << (positions[i][0]+(0.5*ds)) << '\t' << positions[i
    ][1] << '\t' << (-1.*positions[i][2]+R/2.) << endl;
    }
    f.close();
}
```

**The C++ code below initializes the chains on square lattices on two flat plates (with *no* offset between lattices on opposing plates), at fixed ligand coverage.**

```
#include <cmath>
#include <fstream>
#include <iostream>
#include <cstdlib>
#include <string>
using namespace std;

/*
 This script generates chains on two flat plates which are parallel to the xy−plane.
     The two plates can have a minimum surface separation of approximately 5.5,
    regardless of the number of segments per chain. The chains will loop around,
    therefore not passing through the walls. ”chains.xyz” is the output file
    containing particle types and coordinates, and the total number of atoms.
 Version 6.0: An extension of Version 5.0. This version initializes the chains so
    that there is no offset between the chains on the two flat plates.
 */

# define nPerChain 8 // Number of monomers per chain
# define nChain 64 // number of chains per wall
# define R 10. // surface separation
# define v 0.022222222222 // ligand coverage
int N = 2*nPerChain*nChain; // number of monomers
float separation = sqrt(nChain/v); // determines the side length of the lattice
float ds = separation/sqrt(nChain); // determines the distance between rows and
    columns of chains

int main(){
    cout << ”Separation ” << separation << endl;
    cout << ”ds ” << ds << endl;
    ofstream f;
    f.open(”chains.xyz”);
    f << N << endl;
    f << ” linear chains of atoms” << endl;

```

```
int types [N/2];
float positions[N/2][3];

float x = 0.000000;
int i = 0; // keeps track of the monomer index
for(int l = 0; l < sqrt(nChain); l++){ // loop over one dimension of chains on a wall
    float y = 0.000000;
    float yPrev = 0.0000000; // stores the position of the start of the previous chain in the column
    for(int j=0; j<sqrt(nChain); j++){ // loop over second dimension of chains on a wall
        float z = 0.000000;
        for(int k = 0; k<nPerChain; k++){ // loop over beads in one chain
            if(k==0){
                yPrev = y; // stores the position of the start of the chain
            }
            positions[i][0] = x;
            positions[i][1] = y;
            positions[i][2] = z;
            if(k % nPerChain == 0){ // fixed atom
                types[i] = 2;
            }
            else{
                types[i] = 1;
            }
            if((int(y-yPrev) % 4) == 0){
                z += 1.0000000;
            }
            else if((int(y-yPrev) % 4) == 2){
                z -= 1.0000000;
            }
            else{
                y += 1.0000000;
            }
```

```
                if(z>=((R/2.)-1.5)){ // next particle will overlap the wall on the
    right side
                    z -= 1.0000000;
                    y += 1.0000000;
                }
                else if(z<=0.000000){
                    z += 1.0000000;
                    y += 1.0000000;
                }
                i += 1;
            }
            y = yPrev+ds; // determines the y-coordinate of the fixed atom for the
    next chain in the column using the ligand coverage value and the start of the
    previous chain
        }
        x += ds; // start a second column

    }
    for(int i = 0; i<N/2; i++){ // left wall
        f << types[i] << '\t' << positions[i][0] << '\t' << positions[i][1] << '\t'
    << (positions[i][2]-R/2.) << endl;
    }
    for(int i = 0; i<N/2; i++){ // right wall, translate x-positions by 2.0
        f << types[i] << '\t' << positions[i][0] << '\t' << positions[i][1] << '\t'
    << (-1.*positions[i][2]+R/2.) << endl;
    }
    f.close();
}
```

**The TCL script below was used to create the bonds between beads that were closer than a certain threshold value. This script was adapted by Professor Alan Denton from a tutorial written by Axel Kohlmeyer that explains the use of LAMMPS and VMD to build bead-spring models of polymers.**

```
#!/usr/bin/tclsh


# use topotools to generate topology data from coordinate data
# for linear chains of model particles.
##############################################

# load coordinates, but don't automatically compute bonds.
mol new chains.xyz autobonds no waitfor all

# set atom name, type and radius for all atoms
set sel [atomselect top all]
$sel set radius 0.85
$sel set name A
$sel set type A
$sel set mass 1.0

# bonds are computed based on distance criterion: 0.6 * (r_A + r_B) > r_AB.
# with radius 0.85 the cutoff is 1.02, so particles 1.0 unit apart will be bonded.
mol bondsrecalc top

# now recompute bond types.
# by default a string label: <atom type 1>-<atom type 2>
# we have all atoms of type A, so there should be only
# one bond type, A-A
topo retypebonds
vmdcon -info "assigned [topo numbondtypes] bond types to [topo numbonds] bonds:"
vmdcon -info "bondtypes: [topo bondtypenames]"


# now derive angle definitions from bond topology.
# every two bonds that share an atom yield an angle.
topo guessangles
vmdcon -info "assigned [topo numangletypes] angle types to [topo numangles] angles:"
vmdcon -info "angletypes: [topo angletypenames]"

# now let VMD reanalyze the molecular structure
# this is needed to detect fragments/molecules
# after we have recomputed the bonds
mol reanalyze top
```

```

# now set box dimensions and write out the result as a lammps data file.
pbc set {100.0 100.0 100.0 90.0 90.0 90.0}
topo writelammpsdata data.chains angle

# done. now exit vmd
quit
```

**The Python code below was used to change the types of the beads that were fixed to the walls to distinguish them from beads that could move. It generated a data file that was used in place of the one generated by the above TCL script.**

```
# This code changes the types of the fixed atoms to 2 from 1, for chains covering 2
    surfaces
# Users must take the "Atoms" part of the data.chains file and copy all of the atom
    numbers, types, and coordinates into a file called 'chainspositions'. Start from
    the first line containing atom data.
# This code somehow produces atom 1 twice in the new data.chains file, be sure to
    delete the extra one before running your simulation!

import numpy as np

nPerChain = 8 # number of atoms per ligand
N = 1024 # number of atoms

f = open('chainspositions', 'r') # read in the file for coordinates and atom types
data = f.readlines()
f.close()
positions = list()
for line in data:
    positions.append(line.split())

for i in range(len(positions)):
    if (int(positions[i][0])-1)%nPerChain == 0:
        positions[i][2] = '2' # change the atom types of the respective atoms

f = open('data.chains', 'r') # open the file for reading
```

```
datachains = f.readlines()
f.close()
chainsFile = list()
for line in datachains: # take the original data.chains file and copy the data
    chainsFile.append(line.split())
counter = 1
for i in range(34, len(chainsFile)): # Append the coordinates with new atom types
    chainsFile[i] = positions[counter-1]
    counter += 1
    if(counter > N):
        break
f = open(’data2.chains’, ’w’) # Write data.chains with correct atom types to a new
    file
for i in range(len(chainsFile)):
    for j in range(len(chainsFile[i])):
        f.write(str(chainsFile[i][j]) + ” ”)
    f.write(’\n’)
f.close()
```

**The script below is a sample LAMMPS input script, used to run the simulation. Using this particular input script, we are able to run 4 simulations sequentially without having to submit the input script 4 times. Because of how large one data file (needed to initialize the system) is, we have not included one in this thesis. However, these data files are generated using the TCL script, so one can easily generate their own data file.**

```
# input script for bead-spring chains
units lj
boundary p p f
atom_style angle
variable r index 2.5 3. 3.5 4.
variable w index 1.25 1.5 1.75 2.
log log.$r
read_data data.chains$r

# interaction styles
```

```
pair_style lj/cut $r
bond_style harmonic
angle_style harmonic

group 1 type 1 # atoms not at ends of chains will be allowed to move
group 2 type 2 # atoms at ends of chains will be held fixed

# don't compute non-bonded interactions where we have bonded potentials
special_bonds lj/coul 0.0 0.0 1.0

# force field parameters
pair_coeff   * *   1. 1.
bond_coeff   1     100. 1.
angle_coeff  1    10. 120.



# run a few steps of MD to break symmetries.
velocity all create 3.0 4324324 rot yes

compute 1 all temp # computes temperature
compute 2 all pair lj/cut

fix 1 1 nvt temp 3.0 3.0 100.0 # move atoms of type 1 (all atoms except end atoms)
#fix 2 1 wall/lj93 zlo -3.0 0.5 1.0 0.50 zhi 3.0 0.5 1.0 0.50 # walls with lj93
    potential (testing)
#fix_modify 2 energy yes

fix 2 1 wall/reflect zlo -$w zhi $w
fix 4 all ave/time 10000 100 1100000 c_1 c_2 file avg$r.out mode scalar

# dump data file of atomic coordinates
#dump            1 all atom 10000 dump$r.chains
dump            1 all custom 10000 dump2$r.chains id type x y z

thermo_style custom step temp press pe etotal epair emol
thermo 10000
```

```
run 1100000


clear
next r
jump in.chains
```

**Finally, the Python script below is used to calculate the force per unit area between each wall and take the ensemble average over a simulation.**

```
import math
import numpy as np

# Computes the force directly using the exact equations for the LJ force between
    beads and the steric potential between a bead and the wall
# Version 2.0: accounts for equilibration time, not used to compute forces anymore
# Update late Feb. 5, 2018: fixed the atom numbering issue
# Update Feb. 6, 2018: Fixed the x/y component calculations: the azimuthal angle was
     defined incorrectly in the update from Feb. 5

# bead-bead interaction parameters
R = 4.

sigma = 1.0 # LJ sigma
rcut = R # cutoff distance
epsilon = 1. # LJ-epsilon parameter

# LJ bead-wall interaction parameters

sigmaW = 1.0
rcW = R
epsilonW = 1.0

# System parameters

wallPosLeft = R/2. # absolute value of the z-position of the left wall
wallPosRight = R/2.
N = 1024 # number of beads
```

```
nPerChain = 8 # number of atoms per chain
nChains = 128 # number of chains
nt = 111 # number of timesteps output to the dump files
nt2 = 11 # number of timesteps used for equilibration

# box dimensions, taken from data.chains
xMin = 0.0
xMax = 53.6656 # xMax from data file
yMin = 0.0
yMax = 53.6656 # yMax from data file
tolerance = R # determines whether or not to check the interaction between a
    particle and a periodic image of another particle

timeStep = np.zeros(nt,dtype=int)
startOfData = np.zeros(nt,dtype=int)
wallLeft = np.zeros((nt,N/2,4)) # stores bead information for each chain for each
    timestep on the lefthand wall
wallRight = np.zeros((nt,N/2,4)) # same as wallLeft but for the righthand wall
forces = np.zeros((nt,2,4)) # stores the force values for each timestep: the middle
    index has 0 for interactions on the left wall from the right wall, 1 on right
    from left
F = 0.0 # THE TOTAL FORCE AVERAGED OVER TIMESTEPS TO BE OUTPUT

# Temporarily stores x,y-positions for the particles
# on the left wall and the right wall, to account for PBC
xL = 0.0 # xPos for bead on the left-hand wall
yL = 0.0 # yPos for bead on the left-hand wall
xR = 0.0 # xPos for bead on the right-hand wall
yR = 0.0 # yPos for bead on the right-hand wall



# METHODS

# calculate bead-wall interactions
def leftCalculateBeadWallForce(k,atomIndex): # beads on the left wall
    fz = 0.
    rw = np.abs(wallLeft[k][atomIndex][3]-wallPosRight) # right wall
    if rw < rcW:
        fz += 3.*epsilonW/rw*(0.4*(sigmaW/rw)**9-(sigmaW/rw)**3)
    return fz

# calculate wall-bead interactions
def leftCalculateWallBeadForce(k,atomIndex):
    fz = 0.
    rw = np.abs(wallPosRight - wallLeft[k][atomIndex][3])
    if rw < rcW:
        fz -= 3.*epsilonW/rw*(0.4*(sigmaW/rw)**9-(sigmaW/rw)**3)
    return fz

# calculate bead-wall interactions
def rightCalculateBeadWallForce(k,atomIndex): # beads on the right wall
    fz = 0.
    rw = np.abs(wallRight[k][atomIndex][3]+wallPosLeft)
    if rw < rcW:
        fz -= 3.*epsilonW/rw*(0.4*(sigmaW/rw)**9-(sigmaW/rw)**3)
    return fz

# calculate wall-bead interactions
def rightCalculateWallBeadForce(k, atomIndex):
    fz = 0.
    rw = np.abs(-1.*wallPosLeft - wallRight[k][atomIndex][3])
    if rw < rcW:
        fz += 3.*epsilonW/rw*(0.4*(sigmaW/rw)**9-(sigmaW/rw)**3)
    return fz

def oppositeBeadsInteractionsLR(timeNumber,fixedAtom,variableAtom, xRight, yRight):
   # forces between beads on opposite walls
    bForce = 0.
    fn = [0.,0.,0.]
    rx = xRight-wallLeft[timeNumber][fixedAtom][1]
    ry = yRight-wallLeft[timeNumber][fixedAtom][2]
    rz = wallRight[timeNumber][variableAtom][3]-wallLeft[timeNumber][fixedAtom][3]
    r = np.sqrt(rx*rx+ry*ry+rz*rz)
    if r < rcut:
        bForce = 24.*epsilon/r*(2.*(sigma/r)**12-(sigma/r)**6)
    if(bForce == 0.):
        return fn # don't compute components
    cosTheta = np.dot([rx,ry,rz],[0.,0.,1.])/r
    fn[2] = bForce*cosTheta # fz
    theta = np.arccos(cosTheta)
    phi = np.arctan2(ry,rx)
    fn[0] = bForce * np.sin(theta) * np.cos(phi)
    fn[1] = bForce * np.sin(theta) * np.sin(phi)
    if r > 2**(1./6.)*sigma:
        fn[0] *= -1.
        fn[1] *= -1.
        fn[2] *= -1.
    return fn

def oppositeBeadsInteractionsRL(timeNumber,fixedAtom,variableAtom, xLeft, yLeft): # forces between beads on opposite walls, by switching the order of the beads
    bForce = 0.
    fn = [0.,0.,0.]
    rx = xLeft-wallRight[timeNumber][fixedAtom][1]
    ry = yLeft-wallRight[timeNumber][fixedAtom][2]
    rz = wallLeft[timeNumber][variableAtom][3] - wallRight[timeNumber][fixedAtom][3]
    r = np.sqrt(rx*rx+ry*ry+rz*rz)
    if r < rcut:
        bForce += 24.*epsilon/r*(2.*(sigma/r)**12-(sigma/r)**6)
    if bForce == 0.:
        return fn
    cosTheta = np.dot([rx,ry,rz],[0.,0.,1.])/r
    fn[2] = bForce*cosTheta # fz
    theta = np.arccos(cosTheta)
    phi = np.arctan2(ry,rx)
    fn[0] = bForce * np.sin(theta) * np.cos(phi)
    fn[1] = bForce * np.sin(theta) * np.sin(phi)
    if r > 2**(1./6.)*sigma: # attractive force
```

```
        fn[0] *= -1.
        fn[1] *= -1.
        fn[2] *= -1.
    return fn

# the execution of the program

f = open('dump.chains', 'r')
dump = f.readlines()
f.close()
j = 0 # index for timeStep array when determining start of data for each timestep
for i in range(len(dump)):
    if dump[i].find('NUMBER OF ATOMS') != -1:
        timeStep[j] = int(dump[i-1])
        startOfData[j] = i+7 # stores index of start of the data for the timestep
        j += 1

# Extract data of the beads and place them in arrays according to which wall they
    are attached to
for i in range(nt2,nt):
    leftCount = 0
    rightCount = 0
    k = startOfData[i]
    while dump[k].find('ITEM') == -1:
        atomInfo = dump[k].split(' ')
        numberOfAtom = int(atomInfo[0])
        if numberOfAtom <= (N/2):
            wallLeft[i][leftCount][0] = float(atomInfo[0]) # atom number
            wallLeft[i][leftCount][1] = float(atomInfo[2]) # x positions
            wallLeft[i][leftCount][2] = float(atomInfo[3]) # y positions
            wallLeft[i][leftCount][3] = float(atomInfo[4]) # z positions
            leftCount += 1
        else:
            wallRight[i][rightCount][0] = float(atomInfo[0]) # atom number
            wallRight[i][rightCount][1] = float(atomInfo[2]) # x positions
            wallRight[i][rightCount][2] = float(atomInfo[3]) # y positions
            wallRight[i][rightCount][3] = float(atomInfo[4]) # z positions
            rightCount += 1
        k += 1
        if k == len(dump):
            break

print(”Extracted particle coordinates, now computing forces” + ’\n’)
average = 0.0
g = open(’forces ’ + str(R),’w’)

for k in range(nt2,nt): # loop over timesteps computed
    for i in range(N/2): # loop over particles on the left−hand wall
        #if int(wallLeft[k][i][0]) % nPerChain != 1: # not a fixed bead on the left−
   hand wall
            #forces[k][1][2] += leftCalculateBeadWallForce(k,i)
            #forces[k][0][2] += leftCalculateWallBeadForce(k,i)
        for j in range(N/2):
            xR = wallRight[k][j][1]
            yR = wallRight[k][j][2]
            oppBeadsForces = oppositeBeadsInteractionsLR(k,i,j,xR,yR)
            for l in range(3):
                forces[k][0][l] += oppBeadsForces[l]
            if(np.abs(xR-xMax)<tolerance): # check whether particle j is near the
   plus x boundary
                oppBeadsForces = oppositeBeadsInteractionsLR(k,i,j,xR-xMax,yR)
                for l in range(3):
                    forces[k][0][l] += oppBeadsForces[l]
            elif(np.abs(xR-xMin)<tolerance): # check near the minus x boundary
                oppBeadsForces = oppositeBeadsInteractionsLR(k,i,j,xR+xMax,yR)
                for l in range(3):
                    forces[k][0][l] += oppBeadsForces[l]
            if(np.abs(yR-yMax)<tolerance): # check whether particle j is near the
   plus y boundary
                oppBeadsForces = oppositeBeadsInteractionsLR(k,i,j,xR,yR-yMax)
                for l in range(3):
                    forces[k][0][l] += oppBeadsForces[l]
            elif(np.abs(yR-yMin)<tolerance): # check near the minus y boundary
                oppBeadsForces = oppositeBeadsInteractionsLR(k,i,j,xR,yR+yMax)
                for l in range(3):
                    forces[k][0][l] += oppBeadsForces[l]
    for i in range(N/2): # consider interactions on the atoms on the right hand wall from the left hand wall
        #if int(wallRight[k][i][0]) % nPerChain != 1: # not a fixed bead on the right-hand wall, DIFFERENT FROM THE ORIGINAL WALL CHAINS (WAS 0)
            #forces[k][0][2] += rightCalculateBeadWallForce(k,i)
            #forces[k][1][2] += rightCalculateWallBeadForce(k,i)
        for j in range(N/2):
            xL = wallLeft[k][j][1]
            yL = wallLeft[k][j][2]
            oppBeadsForces2 = oppositeBeadsInteractionsRL(k,i,j,xL,yL)
            for l in range(3):
                forces[k][1][l] += oppBeadsForces2[l]
            if(np.abs(xL-xMax)<tolerance): # check whether particle j is near the plus x boundary
                oppBeadsForces2 = oppositeBeadsInteractionsRL(k,i,j,xL-xMax,yL)
                for l in range(3):
                    forces[k][1][l] += oppBeadsForces2[l]
            elif(np.abs(xL-xMin)<tolerance): # check near the minus x boundary
                oppBeadsForces2 = oppositeBeadsInteractionsRL(k,i,j,xL+xMax,yL)
                for l in range(3):
                    forces[k][1][l] += oppBeadsForces2[l]
            if(np.abs(yL-yMax)<tolerance): # check whether particle j is near the plus y boundary
                oppBeadsForces2 = oppositeBeadsInteractionsRL(k,i,j,xL,yL-yMax)
                for l in range(3):
                    forces[k][1][l] += oppBeadsForces2[l]
            elif(np.abs(yL-yMin)<tolerance): # check near the minus y boundary
                oppBeadsForces2 = oppositeBeadsInteractionsRL(k,i,j,xL,yL+yMax)
                for l in range(3):
                    forces[k][1][l] += oppBeadsForces2[l]
    forceMag = 0.0
    forceMag2 = 0.0
```

```
        for l in range(3):
            forceMag += (forces[k][0][l])**2
            forceMag2 += (forces[k][0][l])**2
        forces[k][0][3] = np.sqrt(forceMag)
        forces[k][1][3] = np.sqrt(forceMag2)
        if forces[k][0][2] < 0.: # signifies fz away from the right wall a.k.a. repulsive
            forces[k][0][3] *= -1. # keep directions consistent with the coordinate system used in the simulation
        if forces[k][1][2] < 0.: # attractive force on the right wall from the left wall, pointing in the negative-z direction
            forces[k][1][3] *= -1.
        print ('timestep ' + str(k) + ' on the left wall ' + str(forces[k][0]))
        print ('timestep ' + str(k) + ' on the right wall ' + str(forces[k][1]))
        average += forces[k][0][3]
        g.write(str(timeStep[k]) + ' ' + str(forces[k][0][3]) + '\n')
        F += 0.5*(forces[k][0][3]-forces[k][1][3]) # as given in Marla et. al. 2005

average /= float(nt-nt2)
F /= float(nt-nt2)
F /= ((xMax-xMin)*(yMax-yMin)) # force per unit area
print('Force ' + str(F))
g.write('Force ' + str(F))
g.close()
```