STRUCTURE AND DYNAMICS IN COLLOID–POLYMER MIXTURES OUT OF EQUILIBRIUM

Michael Patrick Howard

A Dissertation
Presented to the Faculty
of Princeton University
in Candidacy for the Degree
of Doctor of Philosophy

RECOMMENDED FOR ACCEPTANCE
by the Department of
CHEMICAL AND BIOLOGICAL ENGINEERING
Adviser: Athanassios Z. Panagiotopoulos

JUNE 2018
Abstract

Colloidal dispersions out of equilibrium have wide-ranging applications in science and engineering, including biotechnology, consumer products, and the chemical industry. Examples include high-throughput particle separation using microfluidic devices, the deposition of multilayer or functional coatings, and the assembly of colloidal crystals for optics and photonics. Colloidal dispersions are typically carefully engineered and often have multiple components, e.g., hard particles, polymers, and surfactants. A key challenge is to connect the microscopic properties of the colloidal dispersion to the macroscopic properties that emerge such as large-scale assembled structures or the suspension rheology. Predicting this relationship is complicated by the intimate coupling between structure and dynamics in colloidal suspensions at the microscopic level, which is the focus of this dissertation.

Massive-scale molecular simulations are applied to study the structure and dynamics of colloids, polymers, and their mixtures out of equilibrium. The simulations are accelerated by algorithms and software developed for the massively parallel architectures of graphics processing units. I first investigate the cross-stream migration of colloids in microfluidic channels. A distinct focusing into the channel center is obtained for colloids in dilute polymer solutions, but significant scatter around this focused position is observed due to conformational fluctuations in the polymer solution. I next consider the flow-induced axial dispersion of colloids in microfluidic channels. I develop a complete theoretical framework for analyzing this dispersion, finding that it is critical to consider microscopic structuring due to confinement when the colloid diameter is comparable to the channel size. Nonequilibrium structure formation in drying films is then explored. I demonstrate how mixtures of colloids and polymers vertically segregate (stratify) into layers and propose a model for this process based on dynamical density functional theory. I finally consider the evaporation-induced assembly of colloidal crystals, showing how solvent evaporation influences
early stages of crystal formation but that structural rearrangements compete with the drying dynamics at later times to determine the final morphology. In all cases, hydrodynamic interactions are shown to play an essential role in determining the colloid dynamics.
Acknowledgements

It’s interesting to look back after reaching a milestone like finishing a degree and see how you came to be there. The path is rarely, if ever, straight, with many unexpected twists and detours. Five or ten years ago, I can honestly say that I would not have seen myself completing the dissertation that you are hopefully about to read. It’s clear to me that I only made it to this point because of the help, support, and influence of many important people in my life along the way. It’s a pleasure and an honor to be able to properly thank them here.

When I was in high school and getting ready to apply to college, I thought that I wanted to be a physician (a “real doctor”) or a math teacher. My mom, a teacher herself, pushed me to apply for an engineering major instead, and my dad, an engineer, of course supported that too. I didn’t know which one I should try for, but I liked my chemistry courses and so chemical engineering seemed like a good choice. Admittedly, you only find out after you get your degree that chemical engineers are often more concerned with plumbing than chemistry, but I think the pipes and fluid flow turned out to be more fun than reactions in the end.

During my first year as an engineer at Penn State, I was still completely unsure if I had chosen the right major. This was when the second of two bits of good luck came my way that year (more on the first at the end). One day in my first thermodynamics course, my professor, Scott Milner, uttered the line universally dreaded by students, “Could Michael Howard please come see me after class?” I approached him after the lecture, worrying that something was amiss with one of my exams or assignments. I was completely caught off guard when he instead asked if I had thought about doing research and if I would like to stay for the summer to work with him. This turned out to be an experience that completely changed my outlook on engineering and what ultimately motivated me to go to graduate school. I am grateful to Scott
for his mentorship and setting an example for how expressing individual interest in a student can make a huge difference in their life.

After choosing to come to Princeton for graduate school, I was trying to find a dissertation topic, and my adviser, Thanos Panagiotopoulos, pointed me at the intentionally vague area of “colloid and polymer mixtures”. While visiting graduate schools, I remember having a conversation with someone, who hopefully has no memory of this conversation themselves, in which I said that I wasn’t the least bit interested in colloids because I didn’t know what you could do with them, so I was a bit skeptical. Nonetheless, I did a quick literature search that turned up over 1 million results—I think it is safe to say that I was somewhat (woefully?) mistaken. I share this story because I think it highlights many of Thanos’s greatest strengths as a mentor. During my time at Princeton, Thanos has helped me grow significantly as a researcher, teaching me the importance of being open to new problems, asking the right scientific question, and using or developing the right tools to answer it. He gave me the independence to explore a wide range of topics, and I learned as much, if not more, from my failures and mistakes as my successes. His expertise in molecular simulations and thermodynamics has been invaluable, as has his mentorship and advice over the years.

I am also lucky to have had Arash Nikoubashman as a second unofficial adviser for my research. Arash was a postdoctoral fellow in the AZP group when I joined the lab. Anyone who knows Arash can confirm that he has a contagious energy, especially for research and soft matter physics but also for video games and softball. Together, we brainstormed and debated (at times more fervently than at others) many of the problems that came to form the core of this dissertation, which was always an enjoyable process. Arash also taught me the multiparticle collision dynamics method, my first exposure to simulating hydrodynamic interactions, and how to administer
a Linux cluster during our many trips to the Forrestal Campus. I’m fortunate to count him as a mentor, collaborator, and friend.

When visiting the department, prospective graduate students invariably ask, “Why did you choose Princeton?” For me, the answer has always been straightforward—our department is incredibly supportive and collaborative. Initially, this meant that I felt welcome from the first day that I got here, in no small part thanks to the efforts of Karen Oliver. However, as time progressed, conversations around the coffee machine gradually and unexpectedly found their way into my work. I am particularly grateful to Howard Stone, who was always able to shed light on a fluids problem and inspired the chapter on axial dispersion of colloids, and to Sujit Datta, Brian Pethica, and Bob Prud’homme for taking the time to chat and attempt to explain experiments to a theorist.

I have been lucky to work with an amazing group of graduate students in the AZP lab during the last five years: Kevin Daly (GPU and Linux wizard), Nate Mahynski (expert on all things colloid- or bacon-related), Joey Vella (with whom I probably ate one too many lunch specials), Andrew Santos (poet and second-loudest American on our trip to Varenna), Wes Reinhart, Nannan Li, Jeffrey Young, and Shuwen Yue. I would like to especially thank Wes for being a great conference companion (despite his travel curse) and for developing the analysis technique that made the colloid-crystallization project possible. I am grateful for his infinite patience as we puzzled through that problem. The AZP group has also had an outstanding ensemble of postdoctoral researchers during my time: Hao Jiang, Filipe Lima, Zoltan Mester, Gustavo Orozco, and Antonia Statt. Many thanks are owed in particular to Antonia for several fruitful collaborations, including the chapter on stratification with hydrodynamic interactions, and for allowing me to move her mouse to the “wrong” side of the keyboard from time to time. I was also privileged to mentor
Kevin Silmore and Felix Madutsa as they worked on their senior thesis projects in the AZP group—I hope that they learned as much from me as I did from them.

My dissertation research was initially funded by a Department of Defense, Air Force Office of Scientific Research, National Defense Science and Engineering Graduate (NDSEG) fellowship. I was then fortunate to receive a Blue Waters Graduate Fellowship through the Blue Waters sustained-petascale computing project, supported by the National Science Foundation and the state of Illinois. I am grateful to both for giving me the freedom to pursue the different projects in this dissertation and for the conversations they facilitated with outstanding graduate students and researchers across many disciplines. In particular, I am indebted to Josh Anderson at the University of Michigan, with whom I developed the neighbor list algorithm that enabled many of the simulations in this dissertation. Josh has set an excellent example for sustainable scientific software development upon which I have modeled my own work.

Going to graduate school at Princeton has been intellectually rewarding, but what really made it fun and worthwhile was the time I got to spend with friends. Sara Chuang (.com and her colorful coat) has been a part of my Princeton experience since our first graduate school visits and co-founded Terrible Movie Night with me. Lian Zhu gets all the credit for representing our class on the graduate student committee and was great to work with, as was Logan Matthews. Brian Wilson (a man of many names) has been a great friend, even if he is a Giants fan. (Go Birds!) Brian and I also currently hold the title of longest-tenured managers of the C-league summer softball team. Everyone that played with us, especially Wes, Dave Sroczynski, Vicki Lee, Becca Napolitano (+ clipboard), Jeni Sorli, Chet Markwalter, Mary Burroughs, and the all-AZP roster, made this a highlight of every year for me. Equally anticipated was pub trivia with Brian, Cody Nunno, Linda Quartner, and my wife Rachael, where the company was always much greater than our score.
Most importantly, I thank my family for a lifetime of love and support. To my parents, thank you for being my first teachers in school and in life. Everything that I am today is only because of what you did for us, and you are the best role models that a son could ask for. Mom, I promise that I finally understand why you get more when you divide by a fraction. Thank you for guiding me to find the things in life that I am passionate about and helping me stick to them. Dad, thank you for the time you spent checking over my math homework every evening. I might argue that differential equations turned out to be better puzzles than trigonometric proofs, but I think this is still up for debate. Thank you for every Boy Scout trip, offsides call or yellow card analysis, and PB&J packed lunch; it has all meant the world to me. To my brother Matthew, you have always pushed me to do better, probably because you are one step ahead of me on most things, maybe even in my own dissertation. We have only grown closer as we’ve gotten older, and I think that is the best thing a brother could ask for. To my wife Rachael, she knows what she did. This concludes my acknowledgements.

Just kidding.

The first bit of luck that I had in my first year at Penn State was meeting Rachael. She has been with me on every step of this journey, which is a long way from Simmons Hall. Over the years, we have driven more miles than we can add between Penn State, Delaware, and Princeton, but they have all been worth it. Rachael has opened up my thinking in ways that I didn’t realize I needed, and all with an incredible sense of humor that keeps a constant smile on my face. One professor asked “How happy are you?” at a recruiting event and wouldn’t believe me when I said “100%”. She is the reason why. I don’t think there is a model accurate enough to predict where our path will go next, but I look forward to wherever life takes us, together.
For my family,
always.
# Contents

Abstract ................................. iii

Acknowledgements ........................ v

List of Tables ............................ xvi

List of Figures ............................ xvii

1 Introduction ............................ 1

2 Inertial and viscoelastic forces on colloids in microfluidic channels 9
  2.1 Model and Methods .................... 13
  2.2 Results and Discussion ................. 18
      2.2.1 Effect of polymer length .......... 18
      2.2.2 Solution rheology ................ 22
      2.2.3 Sensitivity to flow rate and particle diameter 23
      2.2.4 Particle distributions .......... 26
  2.3 Conclusions .......................... 30

3 Axial dispersion of colloids in microfluidic channels 32
  3.1 Theoretical Model ..................... 35
  3.2 Simulation Model and Methods .......... 38
  3.3 Results and Discussion ............... 43
      3.3.1 Model characterization .......... 43
      3.3.2 Colloid distributions .......... 47
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.3.3 Velocity profiles</td>
<td>48</td>
</tr>
<tr>
<td>3.3.4 Diffusion coefficients</td>
<td>51</td>
</tr>
<tr>
<td>3.3.5 Axial dispersion</td>
<td>53</td>
</tr>
<tr>
<td>3.4 Conclusions</td>
<td>59</td>
</tr>
<tr>
<td>3.A Moment Analysis</td>
<td>60</td>
</tr>
<tr>
<td>4 Stratification in drying colloid mixtures</td>
<td>64</td>
</tr>
<tr>
<td>4.1 Model and Methods</td>
<td>68</td>
</tr>
<tr>
<td>4.2 Results and Discussion</td>
<td>72</td>
</tr>
<tr>
<td>4.2.1 Equilibrium</td>
<td>72</td>
</tr>
<tr>
<td>4.2.2 Effect of size ratio on stratification</td>
<td>75</td>
</tr>
<tr>
<td>4.2.3 Effect of drying rate on stratification</td>
<td>78</td>
</tr>
<tr>
<td>4.2.4 Comparison to theoretical models</td>
<td>80</td>
</tr>
<tr>
<td>4.3 Conclusions</td>
<td>86</td>
</tr>
<tr>
<td>4.A Additional Density Profiles</td>
<td>87</td>
</tr>
<tr>
<td>5 Stratification in drying polymer and colloid–polymer mixtures</td>
<td>89</td>
</tr>
<tr>
<td>5.1 Simulation Model and Methods</td>
<td>90</td>
</tr>
<tr>
<td>5.2 Theoretical Model</td>
<td>94</td>
</tr>
<tr>
<td>5.3 Results and Discussion</td>
<td>97</td>
</tr>
<tr>
<td>5.3.1 Polymer–polymer mixtures</td>
<td>97</td>
</tr>
<tr>
<td>5.3.2 Colloid–polymer mixture</td>
<td>102</td>
</tr>
<tr>
<td>5.4 Conclusions</td>
<td>105</td>
</tr>
<tr>
<td>5.A Colloid–Polymer Mixture Model</td>
<td>106</td>
</tr>
<tr>
<td>5.B Numerical Solution of Model</td>
<td>107</td>
</tr>
<tr>
<td>6 Influence of hydrodynamic interactions on stratification</td>
<td>109</td>
</tr>
<tr>
<td>6.1 Models and Methods</td>
<td>110</td>
</tr>
</tbody>
</table>
A Efficient neighbor list calculation for molecular simulation of colloidal systems using graphics processing units 165

A.1 Algorithms ................................. 169
  A.1.1 Stenciled cell list .......................... 169
  A.1.2 Bounding volume hierarchy .................. 171

A.2 Implementation ................................ 175
  A.2.1 Stenciled cell list .......................... 175
  A.2.2 Building LBVHs .............................. 177
  A.2.3 Traversing LBVHs ............................ 179
  A.2.4 Memory management ........................... 181

A.3 Performance .................................. 182
  A.3.1 Lennard-Jones binary mixture ................. 182
  A.3.2 Colloid solution .............................. 186

A.4 Conclusions ................................. 188

B Efficient mesoscale hydrodynamics: multiparticle collision dynamics with massively parallel GPU acceleration 190

B.1 Algorithm ................................. 193

B.2 Implementation ............................... 196
  B.2.1 Framework ................................. 197
  B.2.2 Particle data ............................... 198
  B.2.3 Domain decomposition ........................ 199
  B.2.4 Cell list ................................. 201
  B.2.5 Cell properties ............................. 203
  B.2.6 Stochastic rotation dynamics .................. 206
  B.2.7 Andersen thermostat .......................... 209
  B.2.8 Particle sorting ............................. 209
B.3 Performance ............................................... 211
B.3.1 Precision ............................................... 211
B.3.2 Scaling ................................................... 215
B.3.3 Polymer solution benchmark ....................... 220
B.4 Conclusions ............................................... 222

Bibliography .................................................. 224
List of Tables

3.1 Simulation parameters and diffusion coefficients. .................. 41
3.2 Theoretical models for axial dispersion. ............................ 55

6.1 Equilibrium densities of polymer mixtures. ......................... 126
6.2 Diffusion coefficients for short polymer in explicit solvent. ........ 128
6.3 Diffusion coefficients for long polymer in explicit solvent. ......... 128
6.4 Diffusion coefficients for short polymer in implicit solvent. ........ 128
6.5 Diffusion coefficients for long polymer in implicit solvent. ......... 128

A.1 Benchmark timings for Lennard-Jones binary mixture. ............ 184
A.2 Colloid solution benchmark parameters and LBVH performance. ... 187

B.1 Summary of MPCD benchmarks. .................................... 212
## List of Figures

2.1 Snapshot of colloids in dilute polymer solution under flow. . . . . . 16
2.2 Average force on a colloid for varied polymer chain lengths. . . . . 19
2.3 Average elastic force on a colloid for varied polymer chain lengths. . 21
2.4 Chain length dependence of average elastic force. . . . . . . . . . . 22
2.5 Average force on a colloid for varied flow rates. . . . . . . . . . . . 24
2.6 Average force on a colloid for varied colloid diameters. . . . . . . . 26
2.7 Colloid distribution in channel for varied polymer chain lengths. . . 28
2.8 Focused and unfocused colloid trajectories. . . . . . . . . . . . . . . 30

3.1 Schematic of parallel plate channel. . . . . . . . . . . . . . . . . . . 35
3.2 Snapshot of colloidal dispersion in parallel plate channel. . . . . . . 39
3.3 Time derivatives of average suspension mean-squared displacement. . 42
3.4 Solvent velocity profile. . . . . . . . . . . . . . . . . . . . . . . . . . . 44
3.5 Single colloid mean-squared displacement. . . . . . . . . . . . . . . 45
3.6 Colloid distribution for varied concentrations. . . . . . . . . . . . . 49
3.7 Suspension velocity profile for varied concentrations. . . . . . . . . . 50
3.8 Comparison of models at varied flow rates. . . . . . . . . . . . . . . 56
3.9 Comparison of models at varied concentrations. . . . . . . . . . . . . 56
3.10 Comparison of theoretical model to all simulation results. . . . . . . . 58

4.1 Schematic of stratification in a colloid mixture. . . . . . . . . . . . . 65
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.2</td>
<td>Equilibrium colloid density profiles.</td>
<td>73</td>
</tr>
<tr>
<td>4.3</td>
<td>Snapshots of stratifying film.</td>
<td>75</td>
</tr>
<tr>
<td>4.4</td>
<td>Colloid density profiles for 8:1 size ratio.</td>
<td>76</td>
</tr>
<tr>
<td>4.5</td>
<td>Distance of stratification peak below drying interface.</td>
<td>77</td>
</tr>
<tr>
<td>4.6</td>
<td>Relative center-of-mass displacements during drying.</td>
<td>78</td>
</tr>
<tr>
<td>4.7</td>
<td>Colloid density profiles for different drying rates.</td>
<td>79</td>
</tr>
<tr>
<td>4.8</td>
<td>Colloid velocity profiles for 8:1 size ratio.</td>
<td>82</td>
</tr>
<tr>
<td>4.9</td>
<td>Migration velocity rescaled according to Eq. 4.9.</td>
<td>83</td>
</tr>
<tr>
<td>4.10</td>
<td>Migration velocity rescaled according to Eq. 4.12.</td>
<td>85</td>
</tr>
<tr>
<td>4.11</td>
<td>Colloid density profiles for 6:1 size ratio.</td>
<td>87</td>
</tr>
<tr>
<td>4.12</td>
<td>Colloid density profiles for 4:1 size ratio.</td>
<td>87</td>
</tr>
<tr>
<td>4.13</td>
<td>Colloid density profiles for 2:1 size ratio.</td>
<td>88</td>
</tr>
<tr>
<td>5.1</td>
<td>Single-chain radius of gyration and hydrodynamic radius.</td>
<td>93</td>
</tr>
<tr>
<td>5.2</td>
<td>Snapshot of stratified polymer–monomer mixture.</td>
<td>98</td>
</tr>
<tr>
<td>5.3</td>
<td>Density profiles of polymer–monomer mixtures with varied polymer lengths.</td>
<td>99</td>
</tr>
<tr>
<td>5.4</td>
<td>Density profiles of polymer–polymer mixtures with varied size ratios.</td>
<td>100</td>
</tr>
<tr>
<td>5.5</td>
<td>Density profiles of polymer–monomer mixtures with varied drying speeds.</td>
<td>101</td>
</tr>
<tr>
<td>5.6</td>
<td>Snapshots of stratified colloid–polymer mixtures.</td>
<td>103</td>
</tr>
<tr>
<td>5.7</td>
<td>Colloid density profiles with varied polymer chain lengths.</td>
<td>104</td>
</tr>
<tr>
<td>6.1</td>
<td>Density profiles of drying polymer mixtures with and without hydrodynamic interactions.</td>
<td>115</td>
</tr>
<tr>
<td>6.2</td>
<td>Temperature and density profiles with and without thermostat.</td>
<td>117</td>
</tr>
<tr>
<td>6.3</td>
<td>Exchange chemical potentials for explicit- and implicit-solvent models.</td>
<td>120</td>
</tr>
<tr>
<td>6.4</td>
<td>Relative diffusion coefficients for explicit- and implicit-solvent models.</td>
<td>122</td>
</tr>
</tbody>
</table>
6.5 Fraction of monomers diffusing into right half of box.  
6.6 Internal structure of single polymer chain.  
6.7 Internal structure of polymer chains in mixtures.  
7.1 Trajectories of evaporation-induced crystallization for explicit and implicit models.  
7.2 Solvent and crystal interface positions.  
7.3 Snapshots of surface structures.  
7.4 Diffusion map for characterizing crystal structures.  
7.5 Evolution of local crystal structures.  
7.6 Effective potential between colloids in implicit model.  
7.7 Radial distribution function for colloids at varied concentrations.  
7.8 Diffusion coefficients for colloids at varied concentrations.  
7.9 Snapshots of surface structure at Pe = 29.  
7.10 Snapshots of surface structure at Pe = 58.  
7.11 Snapshots of surface structure at Pe = 118.  
7.12 Snapshots of fully crystallized suspensions at Pe = 29.  
7.13 Snapshots of fully crystallized suspensions at Pe = 58.  
7.14 Snapshots of fully crystallized suspensions at Pe = 118.  
7.15 Snapshots of crystals with equal film height at Pe = 29.  
7.16 Snapshots of crystals with equal film height at Pe = 58.  
7.17 Snapshots of crystals with equal film height at Pe = 118.  
7.18 Snapshots of final crystal structure with explicit solvent at Pe = 58.  
7.19 Snapshots of final crystal structure with implicit solvent at Pe = 58.  
A.1 Schematic of cell list for colloidal system.  
A.2 Schematic of stenciled cell list for colloidal system.  
A.3 Schematic of bounding volume hierarchy for colloidal system.
Chapter 1

Introduction

Colloidal dispersions [1] are mixtures we interact with everyday as consumer products, food, and household items that also have wide-ranging applications in science and engineering from biotechnology [2, 3] to enhanced oil recovery [4] and materials science [5]. Colloidal dispersions consist of nanometer to micrometer sized particles (“colloids”) dispersed in a liquid solvent in which they undergo Brownian motion [1]. Colloidal particles are often spherical and rigid, but aspherical or deformable particles like cells [6, 7], micelles [8], and polymers [9] can also be considered “soft” colloids. The effective interactions between colloids are comparable in strength to thermal energy and can be tuned by adding, e.g., salt to modify electrostatic interactions [10] or polymers to induce a depletion attraction [11], making colloids promising candidates for functional materials.

Colloidal dispersions are typically carefully engineered for desired properties and specific applications. For example, latex paints [12, 13] are complex formulations containing particles of different sizes and materials [14], surfactants [15], polymers [16], and other additives. The latex particles serve as binders for the pigments in the paint, surfactants stabilize the dispersion and extend its shelf life, while the polymers modify the viscosity of the paint for applying the coating. It is a key
challenge to connect the microscopic properties of the colloids (e.g., size, interactions, composition) to the emerging macroscopic properties of the entire dispersion (e.g., structure, rheology). Indeed, an early triumph of computer simulations was to show that hard spherical colloids form crystals at sufficiently high densities [17]. It was recently shown that polymers incorporated into the voids of a hard-sphere crystal enhance the stability of one of these competing crystal lattices [18], highlighting an unexpected interplay between microscopic and large-scale structure in mixtures.

To date, much work has focused on predicting equilibrium properties of colloidal dispersions, such as thermodynamic phase diagrams [19]. Here, the goal is usually to understand how colloids self-assemble and to manipulate the interactions between colloids for this purpose [20–22]. However, despite careful design of properties at equilibrium, it is often the case that colloidal dispersions are out of equilibrium during at least part of their usage or processing. In the case of self-assembly, colloids must dynamically form the equilibrium structure (e.g., a crystal) from an initial condition (e.g., a disordered state). This process usually does not give the thermodynamically favored structure but instead produces defective, polymorphic assemblies [23, 24] when there are competing structures with similar free energies [25–27].

Predicting relationships between microscopic and macroscopic properties out of equilibrium is considerably complicated by the fact that structure and dynamics in colloidal dispersions are intimately linked. One natural example is when a dispersion is driven in a flow, e.g., colloids are pumped through a reservoir for enhanced oil recovery or cells are transported in the blood stream. In these cases, colloids experience additional forces that are not present at equilibrium. For example, shear flow alters the structure of a polymer chain, which transitions from a random coil at low shear rates to a stretched state at high shear rates [28]. The motion and dynamics of the polymer will be influenced by the conformation, which in turn affects how it interacts with other polymers and colloids. Changes in macroscopic properties of the
suspension, such as shear-thinning of the viscosity, also accompany this transition in the polymer conformations [29].

Another process where coupled structure and dynamics play an important role is the formation of a coating by drying the solvent from a dispersion. This is, for example, the process by which latex paints are applied to surfaces: drying concentrates the colloids until they form a close-packed structure, after which an interdiffusion process creates a continuous coating [12, 13]. Drying colloidal dispersions can also be found in processes such as ink jet printing [30, 31], manufacturing polymer LED displays [32, 33], spraying pesticides [34], or even simply in the ring that is deposited by a spilled drop of coffee or tea [35–40]. In all these cases, drying modifies the distribution of colloids in dispersion depending on their dynamics, creating internally structured regions [41]. The dispersion may form a different structure near the drying interface than in the bulk, presenting an opportunity to engineer the structure of the dried coating through the colloid dynamics and processing conditions, which are themselves altered by the structure.

In this dissertation, I study colloids, polymers, and their mixtures out of equilibrium with a focus on two types of processes: flow in microfluidic channels (Chapters 2–3) and evaporation of films (Chapters 4–7). In both cases, I connect the microscopic structure and dynamics of the colloids to emerging large-scale properties. To this end, it is critical to be able to monitor both structure and dynamics concurrently, which is often challenging to do with sufficient spatial or temporal resolution in experiments. Theoretical modeling of such problems is complicated by the presence of hydrodynamic interactions, fluctuating microscopic details, and complex geometries. These difficulties motivate the numerical approach employed in this dissertation, where I apply massive-scale molecular simulations to model these processes with microscopic detail of the structures of interest and faithfully resolved dynamics. The rest of this dissertation is organized as follows.
The behavior of spherical colloids dispersed in a dilute polymer solution is investigated in Chapter 2. Under pressure-driven flow in a microfluidic channel, a distinct migration of the colloids into the channel center is obtained for specific polymer and flow conditions. This effect, known as viscoelastic focusing [42–47], can be exploited for high-throughput cell sorting or particle filtration. In agreement with well-established results for millimeter-sized particles [42–44], simulations reveal that the migration originates from a competition between viscoelastic forces exerted by the polymer on the colloids and hydrodynamically-induced inertial lift forces. However, at the colloidal scale, Brownian motion leads to significant fluctuations in the colloid positions around the channel center. Moreover, a complex interplay between the colloids and polymer chains enhances these fluctuations compared to when such interactions are suppressed, suggesting an important coupling between structure and dynamics for flow-induced focusing of colloids.

The axial dispersion of colloids under pressure-driven flow in a microfluidic channel is then considered in Chapter 3. Such dispersion is relevant for controlling the distribution of colloids in geological fractures for oil recovery or wastewater treatment and for the design of microfluidic devices. A classical analysis for this problem [48–50] is extended to model colloids having diameters comparable to the channel width. This confinement establishes a non-uniform colloid structure across the channel and profoundly influences the colloid dynamics. It is shown that this microscopic structure must be incorporated into theoretical models in order to reliably predict the dispersion dynamics.

In Chapters 4–7, I shift focus from colloids in microfluidic devices to the case of drying films. It was observed in experiments [51–53] that mixtures of small and large colloids dynamically segregate (stratify) into layers based on their size. At fast drying rates, the small colloids were unexpectedly enriched near the drying interface, while the large colloids were pushed down into the film. This stratification can also
be found in other drying mixtures, for example, during the processing of polymer nanocomposites [54]. Stratification may be advantageous, e.g., to create a multilayer coating during a single drying step, or undesirable, e.g., if a uniform coating is required. As will be shown, the stratification process is a purely nonequilibrium effect where large-scale stratified structures form as a result of the microscopic diffusive dynamics.

Stratification of mixtures of spherical colloids is systematically studied over a range of colloid sizes and drying rates in Chapter 4. The stratified layer of small colloids is found to grow faster and to larger thicknesses for larger size ratios and drying rates. A model based on dynamical density functional theory is proposed to explain the stratification. Chapter 5 extends this analysis to the case of drying polymer–polymer and colloid–polymer mixtures. The polymer–polymer mixtures stratify analogously to the colloids with the shorter polymers enriched near the drying interface. The colloid–polymer mixtures stratify into a polymer-on-top structure when the polymer radius of gyration is comparable to or smaller than the colloid diameter, and a colloid-on-top structure otherwise. These results are rationalized theoretically using an extension of the dynamical density functional theory of Chapter 4. The important role of hydrodynamic interactions, a fundamentally dynamic effect, in determining the stratified structure is demonstrated in Chapter 6.

I finally consider a different type of structure formation in a drying film in Chapter 7: the evaporation-induced assembly of colloidal crystals. Colloidal crystals are often prepared by evaporation from solution, and there is considerable interest to link the processing conditions to the crystal morphology and quality for applications in optics [55] and photonics [56]. Simulations reveal that although the quality of initial surface structures is sensitive to the evaporation rate, the final crystal morphology is surprisingly independent of it due to collective rearrangements that relieve defects from the bulk crystal. Competing equilibrium and nonequilibrium growth mecha-
nisms can be controlled through the initial colloid concentration and evaporation rate to rapidly process colloidal crystals while also annealing defects, highlighting an important interplay between structure and dynamics in setting the crystal morphology. As in the case of the stratifying films, solvent-mediated interactions are shown to play a critical role in the crystallization kinetics.

The results presented in this dissertation were obtained by massive scale simulations that often required the development of new algorithms and software. In particular, the massively parallel architectures of graphics processing units (GPUs) were heavily utilized to accelerate the simulations. Appendix A describes an efficient algorithm for performing molecular dynamics simulations on GPUs when there are mixtures of colloids of different sizes. Appendix B presents a GPU implementation of the multiparticle collision dynamics algorithm for simulating hydrodynamic interactions at the mesoscale. Both of these advances, which have been released as open-source software, will enable the scientific community to push the boundaries of molecular modeling of colloidal dispersions in the future.

The contents of this dissertation are based on the following publications with the corresponding chapter denoted:


I additionally contributed to the following publications during my dissertation work, some of which are referenced within this document:


* denotes equal contribution.
Chapter 2

Inertial and viscoelastic forces on colloids in microfluidic channels

The reliable sorting and filtration of colloidal particles based on their size, shape, and rigidity is a challenging problem which is relevant for a wide variety of applications, ranging from water purification [57] to biotechnology [58–60]. For example, colloids can be sorted by size using sedimentation, since particles of different weight settle at different rates and thereby separate. However, such an approach has several shortcomings. The buoyancy of the solute particles has to be tuned carefully [61]. Moreover, thermal motion plays a significant role on colloidal length scales, further impeding the partition into monodisperse samples.

In the past decade, microfluidic technologies have emerged as promising candidates for manipulating colloidal particles, since they allow for high-throughput automation of physical, chemical, and biological processes by linking multiple functional elements into a pipeline to form a “lab on a chip” [62]. In principle, it is possible to spatially separate the flowing colloids by imposing external stimuli, such as

as electric [63, 64] or magnetic fields [65]. In practice, however, such active methods are often impractical as they rely on an intrinsic particle property or careful doping of the solute particles. Alternatively, the lateral displacement of solute particles can be controlled by exploiting non-linear flow effects [2, 66]. The advantage of these passive techniques is that they are independent of solute chemistry, and therefore can be applied to a wide range of materials. Furthermore, these effects can be sustained or even amplified at high flow rates, guaranteeing high throughputs.

There have been multiple experimental, theoretical, and simulation studies of particle segregation under flow. Specifically, Segré and Silberberg observed the formation of annular distributions of neutrally buoyant 0.8 mm – 1.6 mm (non-colloidal) spheres in cylindrical channels, when the flow was dominated by inertial effects [67]. Under these conditions, the Navier–Stokes equation describing the fluid dynamics is not time-reversible anymore, and the parabolic velocity profile of the solvent results in a net lift on the solute particles that pushes them away from the channel center. At the same time however, the wall-induced asymmetry in the wake vorticity field of the solute particles leads to a net repulsion away from the walls [66–69]. Recent simulations have demonstrated that this mechanism holds down to the colloidal regime [70], although thermal fluctuations can lead to strongly scattered solute distributions around the annulus.

For most practical filtering and sorting applications it is however more desirable to achieve spatial focusing of the solute particles onto a single line or plane, since this better facilitates the extraction and purification of the target species. One promising pathway for accomplishing this goal is through viscoelastic focusing [42–47]. Early experiments [42–44] revealed the migration of non-colloidal solute particles to the region of lowest shear rate (i.e., the channel center in the case of Poiseuille flow) when the solute was dispersed in a viscoelastic medium and exposed to non-inertial flow. The lateral migration of solute particles has been described by continuum fluid
mechanics as an imbalance of non-linear compressive elastic forces in the viscoelastic medium [44, 45].

In inertial flow, the elastic force from the viscoelastic medium $F_E$ is in direct competition with the outward hydrodynamic lift forces $F_L$ near the channel centerline. Scaling laws for these forces have been formulated in the limit of macroscopic solute particles. Di Carlo et al. [71] showed that the lift force on non-colloidal particles far from the walls in square channels scales as

$$F_L \sim \frac{\rho_s v_{\text{max}}^2 a^3}{L_x},$$

(2.1)

where $\rho_s$ is the solvent density, $v_{\text{max}}$ is the maximum velocity in the channel (directly related to the Reynolds number Re), $a$ is the particle diameter, and $L_x$ is the channel width. The elastic force $F_E$ has been attributed to a normal stress gradient in the viscoelastic medium across the rigid particle [45]. Assuming a negligible second normal stress difference [72], this leads to

$$F_E \sim a^3 \nabla N_1,$$

(2.2)

where $N_1$ is the first normal stress.

For dilute solutions of high molecular weight polymer, $N_1 \sim \dot{\gamma}^\beta$ where $\dot{\gamma}$ is the local shear rate and $1 < \beta \leq 2$ [73]. At low shear rates, $N_1$ is related to the storage modulus $G'$ in the low frequency limit [45],

$$\frac{G'}{\omega^2} = \frac{N_1}{2\dot{\gamma}^2},$$

(2.3)

with oscillatory frequency $\omega$. In most cases, the storage modulus of a viscoelastic medium scales as $G' \sim \omega^2$ in the low frequency regime [74], and so $N_1 \sim \dot{\gamma}^2$ near the centerline. For the upper convected Maxwell model, the first normal stress can be
calculated analytically as \( N_1 = 2\eta \tau_R \dot{\gamma}^2 \) [72], where \( \tau_R \) is the characteristic relaxation time of the polymer. The shear rate across the channel can be characterized by \( \dot{\gamma} \sim v_{\text{max}}/L_x \), leading to

\[
F_E \sim \eta \tau_R v_{\text{max}}^2 (a/L_x)^3.
\] (2.4)

These scaling laws have been derived in the continuum limit where the rigid solute is significantly larger than the molecular constituents of the viscoelastic medium, which allows for a mean-field approximation of the polymer solution. Interestingly, \( F_E \) exhibits a similar scaling behavior as the hydrodynamic lift force \( F_L \) (Eq. 2.1), which sets in at \( \text{Re} \gg 1 \) and pushes the solute away from the channel centerline. Hence, we anticipate an intermediate flow regime, where the leading terms in \( F_L \) and \( F_E \) cancel each other out, and the dynamics are dominated by other terms. In a recent publication [75], D’Avino et al. employed a continuum description of the viscoelastic medium in conjunction with computational fluid dynamics (CFD) to study the effect of inertia in two-dimensional systems, and successfully identified a regime where \( F_L \) and \( F_E \) are competing.

It is unclear, however, whether such a description is valid in the colloidal limit, where the solute particles are of comparable size to the polymer. This is certainly the case in recent experiments [46], where sub-micron polystyrene spheres were immersed in a poly(ethylene oxide) (PEO) solution with a radius of gyration of roughly 100 nm [76]. Additionally, thermal fluctuations play a significant role at these length scales and cannot be neglected anymore [70]. In order to study focusing on the colloidal scale, we recently performed mesoscale molecular dynamics (MD) simulations with multiparticle collision dynamics (MPCD) where we explicitly modeled the constituent polymers of the viscoelastic medium [77, 78]. We identified a broad parameter range in which the flow-induced focusing of the solute particles was possible, demonstrating the general feasibility to sort and filter colloidal particles. As in experiments, we measured the fraction of colloids focused to the channel centerline.
In the chapter at hand, we provide a more quantitative and rigorous analysis of the forces responsible for viscoelastic focusing in the inertial regime. From the measured forces, we extract information about the expected quality of focusing, and sensitivity of focusing forces to relevant design parameters like polymer molecular weight and flow rate. Unlike previous computational efforts to study inertial viscoelastic focusing [75], our simulations are three-dimensional and incorporate thermal fluctuations and microscopic detail of the viscoelastic medium. We demonstrate here that these fluctuations and details impart unique characteristics to colloidal focusing that are not present for macroscopic particles and must be considered when designing microfluidic devices.

2.1 Model and Methods

Viscoelastic focusing has been studied experimentally for both macroscopic and colloidal particles dispersed in aqueous solvents with low polymer concentrations [46, 47, 60, 79]. Typically, the rigid colloids are made of polystyrene, while the polymers are linear PEO or poly(vinylpyrrolidone) (PVP) chains with molecular weights in the range of $10^2 - 10^3$ kg/mol. Accordingly, we modeled our systems as rigid spherical particles dispersed in a dilute solution comprised of the polymers and an underlying Newtonian solvent. As in the experiments, the polymer concentration can be varied to control the rheology of the solution. Because of the inherent disparity of length and time scales between the Newtonian solvent, the polymer, and the colloids, we employed a multiscale hybrid simulation approach based on multiparticle collision dynamics [80–82].

We adopt a similar molecular model as in Ref. 77, which we briefly summarize here. Polymers were represented explicitly as bead–spring chains consisting of $M$ monomers. We modeled the steric interactions between the beads through the purely
repulsive Weeks–Chandler–Anderson (WCA) pair potential [83], which corresponds to good solvent conditions. We defined length scales relative to the monomer diameter $a_m = 1$ and energy scales relative to the monomer–monomer interaction energy $\varepsilon_m = k_B T = 1$. Bonds between monomers were modeled with the finitely extensible nonlinear elastic (FENE) potential [84]. We adopted the standard Kremer–Grest parameters for the springs [85] in order to prevent unphysical bond crossing. Spherical colloids with diameter $a$ were modeled using the WCA potential parameterized as in Ref. 77 so that they had only excluded volume interactions with the polymers. Both the colloid and polymer interactions with the channel walls were modeled through a purely repulsive 9–3 potential [86]. We set the wall interaction energy to $\varepsilon_w = 0.1$ to prevent the wall from excluding too much volume and narrowing the effective channel size accessible to the colloids and polymer. The equations of motion for the colloids and polymers were integrated using the standard velocity Verlet scheme with a step size of $\Delta t_{MD} = 2 \times 10^{-3}$ in MPCD reduced units.

The underlying Newtonian solvent was modeled on the mesoscopic scale using the MPCD technique [80–82]. In MPCD, the solvent is modeled explicitly as an ensemble of ideal point particles with unit mass $m = 1$ that propagate through a series of alternating streaming and collision steps (see Appendix B for an overview). In contrast to classical MD algorithms, the solvent–solvent and solvent–solute interactions are not governed by pairwise potentials, but instead through stochastic collisions. This approach leads to a significant computational speedup, while still preserving the correct hydrodynamic interactions [81, 82].

During each streaming step, all solvent particles move ballistically over a period $\Delta t$ and are reflected off the channel walls using modified bounce-back collision rules [87] to achieve no-slip boundary conditions. The solvent particles interact with the rigid colloids through momentum exchanges, where solvent particles are reflected off the colloid surfaces and impart linear and angular momentum to the colloids, fol-
lowing the algorithm described in Ref. 88. In the MPCD collision step, all solvent particles and monomers are first sorted into cubic cells of edge length \(a_m\), and then undergo a stochastic collision with particles in the same cell where the particle momenta are rotated around a randomly chosen unit vector by a fixed angle \(\alpha\) [80–82]. At each collision step, the cells are shifted by a random three-dimensional vector with components drawn uniformly on \([-a_m/2, +a_m/2]\), to ensure Galilean invariance [89, 90]. We applied cell-level velocity rescaling as a thermostat in order to maintain an isothermal system, which is necessary in non-equilibrium simulations or when viscous heating is significant [81].

The viscosity and diffusivity of the solvent are dictated by the solvent density \(\rho_s\), rotation angle \(\alpha\), and collision time \(\Delta t\) [81, 89, 90]. For our simulations, we chose \(\rho_s = 5\), \(\alpha = 130^\circ\), and \(\Delta t = 0.1\), which results in a dynamic viscosity of \(\eta = 3.96\) for the pure Newtonian solvent. We chose a lower solvent density and higher collision time than in Ref. 88, where \(\rho_s = 10\) and \(\Delta t = 0.04\), for computational expediency. The choice of MPCD parameters in the present work still corresponds to a liquid-like solvent. The mesoscopic character of the MPCD scheme makes it difficult to directly map the reduced viscosity back to physical units [82], and it is instead better to characterize the fluid by dimensionless numbers. A reliable measure of the importance of hydrodynamics is the Schmidt number \(Sc = \nu/D\), which is the ratio between the kinematic viscosity \(\nu\) and the diffusivity \(D\) of the fluid. We find \(Sc \approx 12\) for the MPCD parameters employed in this work, which is sufficiently large for achieving liquid-like behavior [91, 92].

Polymers and colloids were density matched to the solvent according to the corresponding rules for our embedding schemes: \(m_m = \rho_s a_m^3\) for the monomer mass and \(m_c = \pi \rho_s a_m^3/6\) for the colloid mass. We maintained a constant monomer number density \(\rho_m\) in all simulations. We define this density in terms of a volume fraction \(\phi_m = \pi \rho_m a_m^3/6\), which we fixed at \(\phi_m = 0.05\). This leads to a polymer mass frac-
tion of 9%, which is comparable to typical experiments with PVP as the viscoelastic agent [60, 79]. As we will show in Section 2.2.2, this monomer concentration creates a nearly constant viscosity elastic fluid, which agrees well with experimental observations for aqueous PVP solutions [79].

Figure 2.1: Simulation snapshot of multiple colloids in a viscoelastic medium, where \( M = 40 \) and only every fifth polymer has been colored for the sake of clarity. The velocity profile from Poiseuille flow and the coordinate system have been schematically drawn to the right.

All simulations were conducted in a slit channel as schematically shown in Figure 2.1. We set the dimensions to \( L_x = 50 \) in the gradient direction \( x \), \( L_y = 40 \) in the vorticity direction \( y \), and \( L_z = 150 \) in the flow direction \( z \). The channel was periodic in \( y \) and \( z \), and we checked that it was sufficiently large in these dimensions to avoid artificial self-interactions with the periodic images. We generated Poiseuille flow by applying a body force in the flow direction to all solvent particles, as is the case for gravity driven flow in a pipe. For the pure Newtonian solvent, this procedure gives rise to a parabolic velocity profile \( v_z(x) \), which in a slit channel takes the form

\[
v_z(x) = \frac{g}{2\nu} (L_x - x) x,
\]

where \( g \) is the acceleration associated with the body force and sets the flow rate. We quantify the strength of the flow in terms of the channel Reynolds number \( \text{Re} = \frac{v_{\text{max}} L_x}{\nu} \), defined by the slit diameter and the theoretical maximum velocity \( v_{\text{max}} = \frac{gL_x^2}{8\nu} \) reached at the channel center. In non-equilibrium MPCD simulations, it is recommended to keep the solvent velocities well below the medium’s speed of sound \( c_s \) to avoid artificial compression effects [70, 87]. Accordingly, we restricted
our simulations to Mach numbers $Ma = v_{\text{max}}/c_s < 0.5$. Given the isothermal speed of sound $c_s = 1$ in the MPCD liquid, this leads to a maximum Reynolds number of $Re \approx 32$.

In order to test the scaling of the forces on a colloidal particle, we directly computed the effective (mean) force on the particle using a method similar to Prohm et al. [70]. They calculated the solvent-mediated forces on a single colloid $F_s$ in a purely Newtonian liquid using MPCD by fixing the lateral position of the colloid and measuring the time-averaged momentum exchanged from the solvent to the colloid,

$$F_s = \left\langle \frac{\Delta p}{\Delta t} \right\rangle,$$

(2.6)

where $\Delta p$ is the momentum exchange during each MPCD streaming step. $F_s$ is then the average lift force on the particle during this time interval $\Delta t$. This method was shown to give a potential of mean force correctly describing the particle probability distribution (position of colloids in the channel).

We extended the method of Ref. 70 to measure the additional polymer-mediated force exerted by the pairwise repulsions between the colloid and polymer $F_p$,

$$F_p = \left\langle \sum F_{i,c} \right\rangle,$$

(2.7)

where $F_{i,c}$ is the force exerted by polymer $i$ on the colloid, and the sum is taken over all polymers. The average effective force on the particle is then given by $F = F_s + F_p$. The direction of $F$ controls the direction in which the particle will migrate, and so can be used to identify focusing positions in the channel.

We measured $F$ as a function of displacement from the channel centerline. We fixed the lateral position of the colloid in the $xy$-plane, but allowed it to flow freely along the $z$ direction. Because the force should be antisymmetric around the channel centerline, we displaced the colloid only in the positive $x$ direction. We ran extended
simulations with the colloid fixed at each position to generate initial configurations, and confirmed that a steady state had been reached by monitoring the polymer density profiles and average gyration tensors as well as the solvent momentum and drift in the average force on the colloid. We selected three statistically independent configurations at each lateral position once the steady steady state was reached, and restarted the MPCD simulations with a 50,000 MPCD collision warm-up period for the solvent flow. The stochastic nature of the MPCD algorithm randomizes the polymer velocities during the warm-up, and so the three simulations are statistically independent. We then calculated the effective forces on the colloid during 500,000 MPCD collision intervals. We computed the autocorrelation time of the total gradient force on the colloid, and found that for our parameters this force was effectively decorrelated between MPCD collisions. Accordingly, we sampled the total forces every MPCD collision, averaged these forces over each independent simulation, and estimated uncertainties based on the averages from the three simulations.

2.2 Results and Discussion

2.2.1 Effect of polymer length

Figure 2.2 shows the effective force in the gradient direction $F_x$ versus the lateral center of mass displacement for a colloid of diameter $a = 16$. We measured $F_x$ for a purely Newtonian solvent as well as polymer solutions of increasing chain length at flow rate $Re = 32$. We found that a single run was sufficient to calculate the effective forces for the Newtonian solvent with reasonable uncertainty, and so error bars are based on a single run for this case. The displacement from the centerline is measured in reduced coordinates $\xi = (2x - L_x)/L_x$, defined so that $\xi = 0$ when the colloid center of mass is at the channel centerline and $\xi = 1$ when it is at the wall. Due to its rigid nature, the colloid cannot move all the way to $\xi = 1$. The contact
distance for the colloid and wall is \( \xi = 1 - a/L_x \), and so for \( a = 16 \), the colloid touches the wall at \( \xi = \pm 0.68 \). Forces very close to the wall are sensitive to the choice of wall potential used to approximate the interaction between a hard sphere and a hard surface. Moreover, we found it challenging to obtain reliable statistics on the forces very near the wall, and so we restricted the colloid to \( |\xi| \leq 0.6 \).

![Figure 2.2: Average effective force on a single colloid versus displacement of its center of mass from the channel centerline for varying polymer chain lengths \( M \). The colloid diameter is \( a = 16 \), and the flow rate is \( Re = 32 \). Solid curves are polynomial fits to the data. At \( \xi = 0.68 \), the colloid contacts the wall.

Because displacements from the centerline were made for \( \xi > 0 \), a positive force corresponds to an outward motion of the colloid, while a negative force pushes the colloid towards the centerline. For the Newtonian solvent (no polymer), \( F_x \) is positive near the channel centerline, and then becomes strongly negative near the wall. These two regimes correspond to the lateral lift force driving the colloid outward and the inward hydrodynamic wall repulsion. The point where \( F_x \) crosses zero (\( \xi \approx 0.45 \)) gives the location of the equivalent of the Segré–Silberberg annulus [67] in slit flow. The value from our simulations is slightly smaller than the experimentally measured one of \( \xi \approx 0.6 \), which has been determined for non-colloidal particles with \( a \ll L_x \) in channels with circular [67] and square [71] cross sections. This discrepancy is most likely due to the rather large blockage ratio in our simulations, which is \( a/L_x = 0.32 \).
for $a = 16$, and is consistent with the results of Ref. 70 in a cylindrical channel for large blockage ratios.

For short chains of length $M = 10$ and $M = 20$, the same general trend is observed as for the Newtonian solvent, but the magnitude of the lift force is somewhat decreased by the presence of the polymer. This decrease can be attributed to the elastic force induced by the polymer, not yet large enough to overcome the hydrodynamic lift. Hence, the $M = 10$ and $M = 20$ chains do not lead to viscoelastic focusing on the centerline. However, the outward force is considerably smaller for the $M = 20$ chains compared to the $M = 10$ case. Increasing the chain length to $M = 40$ leads to a significant shift in $F_x(\xi)$, and this shift becomes more sizable for $M = 80$. At these longer chain lengths, the elastic force from the polymer is sufficiently large to overcome the inertial lift force, and the particle focuses on the centerline from all positions ($F_x(\xi) \leq 0$). These trends qualitatively match results from two-dimensional CFD calculations of discs in viscoelastic flow that have been conducted at comparable flow rates [75], as well as our earlier simulations [77, 78].

The total force on the colloid is a combination of a hydrodynamic contribution from the solvent and an elastic contribution from the polymer. The hydrodynamic force exerted by all the polymers should be approximately the same provided that the different polymer chain lengths similarly perturb the flow field in the vicinity of the colloid. Based on macroscopic models (e.g., upper convected Maxwell model), the elastic force should be proportional to the relaxation time of the chain $\tau_R$. At rest, the relaxation time scales with the radius of gyration $\tau_R \sim R_g^3$ [93], and $R_g$ in turn is proportional to $M^{3/5}$ [94]. Hence, the elastic force is expected to scale as $M^{9/5}$.

We can separate the elastic component of the force by subtracting the hydrodynamic contribution to the total force. The most convenient way to do this is to subtract the total force exerted on the colloid by a solution of monomers ($M = 1$),
which is essentially a purely hydrodynamic force, from the total force in the polymer solutions. In doing so, we assume that the presence of bonds in the polymers does not significantly perturb the flow field near the colloid compared to the monomer flow field. Then, all remaining forces must be solely due to the polymer elasticity.

Figure 2.3 shows the elastic contribution of the force $F_E$ as a function of colloid position for the different chain lengths. For $\xi \lesssim 0.5$, the elastic force drives the colloid towards the center of the channel for all chain lengths, and the strength of this focusing force increases as $M$ increases. Figure 2.4 shows these forces replotted against $M$ at a fixed channel position. From these data it is clear that the elastic force has a considerably weaker scaling than $M^{9/5}$. In fact, fitting the data through a power law suggests that $F_E$ scales in a sublinear manner with $M$.

![Figure 2.3: Average elastic force on a single colloid versus displacement of its center of mass from the channel centerline for differing polymer chain lengths $M$. Conditions are the same as in Figure 2.2. Solid curves are a guide to the eye.](image)

Figures 2.2 and 2.3 show that there is a crossover in the force curves near the walls ($\xi \gtrsim 0.5$), where the elastic force is starting to push the colloid away from the centerline to the channel walls. This effect becomes more pronounced with increasing chain length, and could potentially lead to the surface pinning of colloids at sufficiently low Re. However, at the investigated flow rates, the hydrodynamic wall repulsion outweighs the elastic contribution close to the walls, leading to focusing irrespective of the initial colloid position. The distance $\xi$ at which the inward and outward forces on the colloid are perfectly balanced is often referred to as the
Figure 2.4: Average effective force on a single colloid versus chain length $M$ at varied positions in the channel for the conditions of Figure 2.2. Solid lines are power law fits $aM^b$ with $b = 0.89 \pm 0.10, 0.74 \pm 0.06, 0.68 \pm 0.12, 0.88 \pm 0.10$ for each channel position, respectively. Error bars are omitted for clarity, but are the same as in Figure 2.3.

“neutral surface”. D’Avino et al. confirmed its existence for shear thinning fluids at $\text{Re} \ll 1$ using CFD and microfluidic experiments, but found no neutral surface for viscoelastic solvents with constant shear viscosity [79].

2.2.2 Solution rheology

In order to better understand the potential role of shear thinning solvent viscosity, we measured $\eta(\dot{\gamma})$ for all investigated polymer lengths $M$ by applying steady shear with the Müller-Plathe reverse perturbation method [82, 95, 96]. In this method, momentum flux along the gradient direction of the flow is induced through momentum exchanges, and the emerging shear rate of the solution is then measured. In our implementation, momentum swaps were only applied to the MPCD solvent particles, which then exchanged momentum with the embedded polymers through the stochastic collision step in the MPCD algorithm. We verified the validity of this approach by conducting additional simulations at selected state points, where we applied the momentum swaps to both monomers and MPCD solvent particles. All simulations were conducted in the bulk to eliminate wall-induced effects [97]. The range of $\dot{\gamma}$ was chosen to cover typical values of the Poiseuille flow simulations in the slit geometry.
To determine the degree of shear thinning, we considered the polymer solutions as power-law fluids and fit the shear stress through $K\dot{\gamma}^n$. In this model, the viscosity of the fluid is quantified by the flow consistency index $K$, while the dimensionless index $n$ characterizes the type of the fluid ($n < 1$ pseudoplastic; $n = 1$ Newtonian; $n > 1$ dilatant). For the pure solvent, we computed a value of $K = 3.98$ and $n = 1$, which is in excellent agreement with theoretical predictions [81, 90] and viscosity measurements from our Poiseuille flow simulations. In the case of the polymer solutions, we found $K = 4.80$ and $n = 0.98$ for the shortest investigated polymers ($M = 10$), and $K = 4.93$ and $n = 0.97$ for the longest ones ($M = 80$). As expected, the addition of polymers leads to an increase of the overall viscosity, where longer polymers have a slightly larger effect. Furthermore, we can conclude from these data that the polymer solutions exhibit no significant shear thinning in the investigated $\dot{\gamma}$ regime.

Hence, our simulations suggest that a neutral surface can in principle exist for viscoelastic solvents with constant shear viscosity. Such a neutral surface exists in an intermediate flow regime where chains are somewhat shear-aligned (based on the trends in Figure 2.3), but the flow rate is not high enough for significant wall repulsions. This finding is in contrast to previous predictions from CFD calculations [79], and might be due to microscopic details of the polymer–colloid interactions which cannot be captured in mean-field treatments of the polymer solution.

2.2.3 Sensitivity to flow rate and particle diameter

To investigate the effect of flow rate on focusing, we took the $M = 40$ polymers that focused for $a = 16$ and $Re = 32$, and systematically decreased the flow rate to essentially non-inertial conditions ($Re = 1$). Figure 2.5 shows the effective force $F_x$ as a function of channel position for various flow rates; it is immediately apparent that the driving force $F_x$ steadily decreases with decreasing flow rate. For $Re = 1$,
there is essentially no focusing force near the channel centerline, but instead there is a noticeable attractive force close to the channel walls. We surmise that this effective wall attraction is the result of polymer depletion effects, akin to the situation at equilibrium: the conformational degrees of freedom of the polymers are significantly restricted in the immediate vicinity of the walls, and therefore prefer to move to the channel center. This purely entropic effect leads to the displacement of the colloids from the channel center, which then manifests itself in the observed attraction [11].

Varying the flow rate affects the forces acting on the colloid in two ways. First, the outward hydrodynamic lift force $F_L$ on the colloid decreases with decreasing flow rate. For a particle Reynolds number $Re_p = Re a / L_x \ll 1$, there is essentially no hydrodynamic lift force and the colloid is confined on a streamline [98], whereas in the inertial regime, $F_L$ increases with the square of the flow rate near the channel centerline (see Eq. 2.1). Second, while the outward lift force is weakening, the magnitude of the inward elastic force is also decreasing at the same time. Macroscopic scaling arguments (see Eq. 2.4) suggest that the elastic force should also be proportional to the square of the flow rate due to the dependence on the first normal stress gradient. From a microscopic point of view, sufficient shear rates are required to deform

Figure 2.5: Average effective force on a single colloid versus displacement of its center of mass from the channel centerline for various flow rates Re. The colloid ($a = 16$) is dispersed in a polymer solution with chain length $M = 40$. Solid curves are polynomial fits to the data.
and shear align the polymers in order to create this normal stress difference. The threshold shear rate \( \dot{\gamma}_c \) is characterized by the dimensionless Weissenberg number \( Wi = \dot{\gamma}_c \tau_R \), where shear alignment is expected when \( Wi \gtrsim 1 \).

For \( M = 40 \), we estimate \( Wi < 1 \) for \( Re = 1 \) and \( Wi \geq 1 \) for all other investigated flow rates, based on the average shear rate in the channel and a relaxation time estimated within the Zimm model using the equilibrium radius of gyration \([93, 99]\). Hence, the \( M = 40 \) polymers should be isotropic at \( Re = 1 \) and stretched otherwise. We confirmed this trend quantitatively by evaluating the time-averaged gyration tensor for the polymers under flow. We believe that because our \( M = 40 \) polymer chains do not stretch at \( Re = 1 \), no viscoelastic focusing is observed. For higher \( Re \) where the chains are stretched, \( F_x \) still does not show a strong dependence on \( Re \), and we conclude that the leading scaling terms for \( F_L \) and \( F_E \) roughly cancel each other out for this state point. The weak \( Re \) dependence observed in Figure 2.5 can then be attributed to prefactors, other terms, and the increased role of wall hydrodynamic repulsions.

We also computed \( F_x \) for varying colloid diameters at fixed flow rate \( Re = 32 \) and polymer length \( M = 40 \), shown in Figure 2.6. \( F_x \) depends only weakly on \( a \) near the centerline, with a stronger inward force for larger particles, suggesting that near the centerline the leading scaling terms of \( F_L \) and \( F_E \) (both predicted to be proportional to \( a^3 \)) have similar prefactors for this choice of parameters. The biggest effect on colloid size appears close to the walls (\( \xi \gtrsim 0.4 \)), where the apparent repulsion is much stronger for the largest colloid than for the smaller ones. This behavior is mainly due to the fact that the larger colloid has a much smaller wall separation compared to the smaller ones at a fixed position: for \( a = 8 \), wall contact does not occur until \( \xi = 0.84 \), whereas the largest colloid is already touching the wall at \( \xi = 0.68 \). Di Carlo et al. have shown that the strength of the hydrodynamic wall repulsion scales
as $\sim a^6$ [71], which is qualitatively consistent with our simulations (i.e., at a fixed position larger colloids have considerably stronger repulsions).

![Graph showing average effective force on a single colloid versus displacement of its center of mass from the channel centerline for various colloid diameters $a$. The colloid is dispersed in a polymer solution with chain length $M = 40$ at flow rate $Re = 32$. Solid curves are polynomial fits to the data.]

Figure 2.6: Average effective force on a single colloid versus displacement of its center of mass from the channel centerline for various colloid diameters $a$. The colloid is dispersed in a polymer solution with chain length $M = 40$ at flow rate $Re = 32$. Solid curves are polynomial fits to the data.

2.2.4 Particle distributions

In practice, it is not only important to know whether a colloidal particle will focus, but also to characterize the quality of focusing. Important characteristics are, for example, the speed with which focusing can be achieved, the width of the focused stream, and its stability. Due to the inherent Brownian motion of colloidal particles, there will be a distribution of particle positions at steady state, which intuitively should be more narrow for stronger focusing forces. Such particle distributions have been measured experimentally to qualitatively characterize focusing of colloidal particles [46]. Prohm et al. characterized the particle distribution of colloidal particles for tube Poiseuille flow in a Newtonian solvent [70]. Here, we apply their methods to characterize particle distributions for colloidal particles in dilute polymer solutions.

The motion of a colloid in the channel gradient direction can be described by a one-dimensional random walker in an external potential. Proceeding as in Ref. 70,
the Langevin equation describing the position of such a walker is

\[ m\ddot{\xi} = -\gamma \dot{\xi} + F(\xi) + \zeta(t), \quad (2.8) \]

where \( \gamma = 3\pi \eta a \) is the Stokes drag coefficient and \( \zeta(t) \) is white noise with zero mean and variance satisfying the fluctuation-dissipation relation \( \langle \zeta(t)\zeta(t') \rangle = 2\gamma k_B T \delta(t - t') \). The steady state distribution function \( f(\xi) \) for such a random walker is a standard Boltzmann distribution,

\[ f(\xi) \sim e^{-U(\xi)/k_B T}, \quad (2.9) \]

where \( U(\xi) \) is the potential of mean force obtained by integrating the calculated forces

\[ U(\xi) = -\frac{L_x}{2} \int_0^\xi d\xi' F_x(\xi') \quad (2.10) \]

with \( U(\xi = 0) = 0 \). In a recent publication [47], De Santo et al. proposed to model the forces exerted by the viscoelastic medium as an effective harmonic potential by considerations of the first normal stress on a Brownian particle. However, their model is restricted to the inertialess limit (Re \( \sim 10^{-8} \)), and is therefore not applicable to our simulations where inertia has a significant contribution to the total effective force.

In order to compute the particle distribution directly, we repeated our simulations starting from the same initial configurations as the previous force calculations in Figure 2.2, but this time allowed the colloids to move freely in all dimensions. Due to the symmetry of the channel geometry, we consider only the probability of finding the colloid at an absolute displacement \( |\xi| \) to improve sampling.

Figure 2.7 shows the probability distributions calculated from our simulations along with the expected distributions based on Eq. 2.9. The inset of Figure 2.7 shows the potentials corresponding to the forces calculated in Figure 2.2. As expected, the simulated particle distributions have a maximum around the minimum of the
potential. Brownian particles can fluctuate within a few $k_B T$ of the minimum, and so for the Newtonian solvent we expect and obtain a peak between $0.4 \lesssim \xi \lesssim 0.5$. The distribution for the Newtonian solvent is not perfectly symmetric, but slightly skewed towards the center of the channel because of the strong hydrodynamic repulsion near the wall.

Figure 2.7: Colloid distribution function for varying polymer chain lengths $M$. Symbols are calculated from simulation, while solid curves are the distributions obtained from Eq. 2.9. Error bars correspond to standard error from taking three subsets of the independent trajectories. Simulation parameters are the same as in Figure 2.2. Inset: potentials of mean force calculated from Figure 2.2 using Eq. 2.10.

In the case of the polymer solutions, we can discern a sizable discrepancy between the simulated particle distribution and the distribution predicted by Eq. 2.9, which becomes more pronounced as the polymer chain length is increased. For the shortest polymers ($M = 10$), the two distributions are still in excellent agreement, whereas for the longer chain lengths, the simulated distributions are much broader than the calculated potentials suggest.

We believe that this discrepancy originates from the method of calculating the force on the colloid and the microscopic relationship between the colloid and nearby polymers: in order to compute the force, we fixed the lateral position of the colloid so that it is unable to move in the gradient direction regardless of the nearby polymer environment. This allows us to sample the average potential of mean force
experienced by the colloid. However, when the colloid is allowed to freely move, it may experience and also induce fluctuations in the nearby polymer environment, e.g., gaps where there are no polymers. We hypothesize that these additional degrees of freedom could lead to colloid trajectories that are inconsistent with the average potential of mean force computed for colloids at fixed position.

This line of reasoning suggests that the discrepancy in the predicted distribution should be the largest near the channel centerline where small movements of the polymers can induce dramatic changes in their conformations. When a polymer is perturbed from the channel centerline (where there is no shear) to a region of finite shear rate, the previously coiled polymer may suddenly stretch if the shear rate exceeds the threshold rate $\dot{\gamma}_c$. This effect is more pronounced for longer polymers because they deform at smaller $\dot{\gamma}_c$. Away from the center of the channel, the polymer conformation is less sensitive to position because the chains are already stretched. We confirmed this trend from the gyration tensor of the polymers, for which we found a significantly stronger position dependence near the channel centerline, especially for longer chains ($M = 40$ and $M = 80$).

Figure 2.8 demonstrates this effect for single colloid trajectories in polymer solutions with chain lengths of $M = 10$ and $M = 80$, respectively. In both systems, the colloid positions fluctuate about the expected distribution peak. However, the fluctuations for the $M = 10$ case are much smaller than for the $M = 80$ case, although the opposite is predicted by Eq. 2.9. Moreover, for $M = 80$, several trajectories exhibit rare but noticeable veerings away from the centerline position, before returning to the middle in a similar fashion (e.g., between $t = 60,000$ and $t = 80,000$). We believe that these events correspond to fluctuations in the calculated average potential of mean force, which are not captured appropriately in $F_x$.

The fluctuation of the local polymer environment in response to the motion of the colloid suggests that the colloid and polymer solution cannot be decoupled from each
other. At these length scales, a mean-field treatment of the polymer as a density or stress field is no longer adequate, and a microscopic description of the polymers is necessary. Although the calculated forces do not quantitatively reproduce the exact particle distributions, we emphasize that our measured force curves still predict the qualitatively correct behavior: polymer chains with $M = 80$ provide better focusing than $M = 40$, whereas chains with $M \leq 20$ do not focus at all. Furthermore, we expect that many body effects, such as hydrodynamic wakes, may play an important role when focusing systems with multiple colloids. These additional terms may have a sizable effect in applications with multiple particles.

### 2.3 Conclusions

We studied the viscoelastic focusing of rigid colloids in slit-like channels using a hybrid molecular dynamics approach that faithfully takes into account hydrodynamic interactions. A bead–spring model was used for the constituent polymers of the viscoelastic medium in a way consistent with the colloidal nature of the solute particles. We systematically measured the forces exerted on the colloid through the surrounding solvent, and decomposed them into inertial and elastic contributions to better
understand the driving force behind the cross-stream migration of colloidal particles at high Reynolds numbers $\text{Re} \geq 1$.

For short polymer chains, the elastic forces were not sufficiently strong to outweigh the hydrodynamic lift forces at the channel center, leading to unfocused states. However, as the polymer length increased, elastic contributions became dominant, resulting in the focusing of the colloids. Further, we found that the total forces became stronger with increasing colloid size and flow rate. However, these parameters exhibited a significantly weaker scaling compared to the predictions within a continuum approximation of the viscoelastic medium. In part, this discrepancy stemmed from microscopic details of the polymer solution that became increasingly important at these length scales, and which manifested as wall attractions and pronounced fluctuations in the colloid trajectories. These microscopic effects cannot be captured in mean-field descriptions of viscoelastic focusing, and potentially need to be taken into account when designing and conducting microfluidic experiments on the colloidal scale.
Chapter 3

Axial dispersion of colloids in microfluidic channels

Having studied the cross-stream migration of colloids in Chapter 2, we now consider motion in the axial (flow) direction. Particles in non-uniform flow fields undergo an enhanced axial dispersion (spreading) compared to diffusion in the absence of flow [100]. Qualitatively, axial dispersion is enhanced by flow because particles diffuse across streamlines and advect at rates different from the average velocity, resulting in a net spreading relative to the mean. In his seminal papers [48, 49], Taylor calculated the asymptotic form of this dispersion coefficient for non-interacting point-like tracer particles in a cylindrical tube under the assumptions that the solute particles diffuse isotropically and explore all streamlines of the flow field uniformly. Aris subsequently performed a moment analysis that justified the assumptions in Taylor's original analysis and captured the transient evolution of the dispersion [50].

The Taylor–Aris description of dispersion works well for molecular solutions, whose components are considerably smaller than the channel diameter. However,

the physical picture becomes significantly more complicated for colloidal systems. Some volume of the channel is excluded to the colloids due to their finite diameters. Under sufficient confinement, the excluded volume between the colloids and channel walls can establish a non-uniform colloid distribution \[101–103\]. This means that the colloids do not explore all streamlines equally. Moreover, due to hydrodynamic interactions between the colloids and the channel walls, the colloid diffusion tensor can be anisotropic \[104\]; the diffusion coefficient normal to the channel walls may differ significantly from the axial diffusion coefficient even at rest. For sufficiently high colloid concentrations, secondary flows between colloids may also modify the effective flow field or diffusion coefficient.

Brenner and Gaydos have comprehensively treated the dispersion problem for a single colloid in a cylindrical pore \[105\]. Their theory accounts for anisotropic diffusion and an applied external field on the colloid. Silebi and DosRamos applied DLVO theory to compute the forces between colloids and the walls of a capillary tube, and also accounted for an effective external potential due to inertial lift on the colloids \[106\]. A simplified version of the Brenner and Gaydos theory assuming no external field and isotropic diffusion has also been developed for rigid colloids between parallel plates and amounts to the introduction of an additional geometric factor that reduces the actual dispersion from the Taylor–Aris value \[107\]. Bhattacharya et al. applied a perturbation method to analyze the dispersion of finite-size particles confined in cylindrical tubes subject to inertial lift \[108\]. However, these theories do not account for non-uniform colloid distributions that can arise even when the colloids have only hard-core volume exclusion with the channel walls. Moreover, classical dispersion theory results may not be sufficiently accurate to model dispersion in a confined suspension, where hydrodynamic coupling between many colloids and the channel walls can affect the dispersion. For example, Griffiths and Stone showed that shear-enhanced diffusion can cause a reduction of the axial dispersion \[109\].
The invention of microfluidic devices has fueled interest in Taylor–Aris dispersion in order to control dispersion [110–112], improve micromixing [62, 113], probe flow fields [114], and measure colloid [115] and nanoparticle [116] diffusion coefficients. Unfortunately, experimental validation of theoretical models for axial dispersion remains challenging due to the small diffusion coefficients of the colloids, which necessitate long channels for capturing the long-time dispersion [117], and the difficulty of performing particle-tracking measurements with sufficient spatial and temporal resolution. Computer simulations are an ideal tool to address these issues because they allow for the precise tracking of the colloidal particles, direct measurement of the flow fields, and systematic control over the relevant parameters, such as solute concentration and flow strength. Recently, two-dimensional mesoscale simulations were performed to study the axial dispersion of repulsive colloidal disks of varying diameter [118]. Significant deviations from the theoretical Taylor–Aris prediction were observed, but due to the challenge of obtaining reliable statistics the authors were unable to conclude whether or not existing theory was sufficient to describe this behavior.

In this chapter, we develop a theory for the axial dispersion of dilute Brownian colloids in a parallel plate geometry that explicitly takes into account the effects of colloid concentration and confinement on the colloid distribution, velocity profile, and diffusion coefficients. We describe a framework to estimate all of the necessary parameters to reliably predict the enhancement to the axial dispersion. We perform complementary molecular dynamics simulations to directly test our theoretical predictions. Our simulation approach fully takes into account hydrodynamic interactions and thermal fluctuations.

The chapter is organized as follows. In Section 3.1, we derive an expression for the axial dispersion of colloids in the parallel plate geometry. We introduce
our simulation model and methodology in Section 3.2. We present our results and findings in Section 3.3, and provide a brief summary and outlook in Section 3.4.

3.1 Theoretical Model

The analysis presented by Brenner and Gaydos [105] for a colloidal particle in a cylindrical pore can be straightforwardly applied to the parallel plate geometry, schematically drawn in Figure 3.1. The coordinate system has been chosen so that the principal axes coincide with the transverse \( x \), vorticity \( y \), and axial directions \( z \). Due to the channel walls, the colloid center of mass cannot cross the boundaries at \( x = \pm \ell/2 \). The accessible width \( \ell \) will be smaller than the actual channel width \( L_x \) because of the finite particle diameter \( a \).

![Figure 3.1: Schematic representation of flow in the parallel plate geometry, indicating the transverse \( x \), vorticity \( y \), and axial \( z \) directions. The colloid of diameter \( a \) diffuses in the accessible width \( \ell \) of the full channel width \( L_x \), and is advected by the velocity field \( u \).](image)

We model the time-dependent probability distribution \( p \) for the center of mass position of a single colloid. In doing so, we neglect correlations between the distributions of different particles and assume that the colloid has negligible inertia on the timescales of interest so that its velocity is Maxwell–Boltzmann distributed. Then, the time evolution of the distribution is governed by the Smoluchowski equation,

\[
\frac{\partial p}{\partial t} + \nabla \cdot \mathbf{j} = 0, \tag{3.1}
\]
with a probability flux $j$,

$$ j = pu - D \cdot \nabla p - p\beta D \cdot \nabla \phi \, . $$  \hfill (3.2)

The first term is the advection due to the flow field $u$, the second term is the diffusion due to Brownian motion with diffusion tensor $D$, and the third term represents an applied force due to some external potential $\phi$, where $\beta = 1/(k_B T)$ is the inverse temperature.

Concentration and confinement effects of the suspension are included in a mean-field treatment through the effective flow field, diffusivity, and external potential. In the parallel plate geometry, the flow field is expected to be one-dimensional in nature: $\mathbf{u} = u(x) \mathbf{e}_z$. Moreover, in the absence of an applied field, the external potential should only be due to the effects of confinement between the walls so that $\phi = \phi(x)$. We neglect any shear-enhanced diffusion so that $D$ has only diagonal elements. This is justified for sufficiently small particles, low concentrations, or slow flow rates, where Brownian motion dominates over advective transport. Simplifying and applying a product rule to Eq. 3.2 yields the governing equation

$$ \frac{\partial p}{\partial t} + u(x) \frac{\partial p}{\partial z} = \frac{\partial}{\partial x} \left( w(x) D_x(x) \frac{\partial}{\partial x} \left( \frac{p}{w(x)} \right) \right) + D_z(x) \frac{\partial^2 p}{\partial z^2} \, , \hspace{1cm} (3.3) $$

where $D_x$ is the diffusion coefficient in the transverse direction, $D_z$ is the diffusion coefficient in the axial direction, and $w$ is the normalized Boltzmann weight for the particle

$$ w(x) = \frac{e^{-\beta \phi(x)}}{\int_{-L/2}^{L/2} e^{-\beta \phi(x)} \, dx} \, . \hspace{1cm} (3.4) $$

In order to close the problem, we apply no-flux boundary conditions at the walls, $j_x(x = \pm \ell/2, z, t) = 0$, and assume an infinite channel in order to neglect entrance effects, $p(x, z \to \pm \infty, t) \to 0$. The colloid’s initial position is assumed to be $(x_0, z_0)$,
so \( p(x, z, t = 0) = \delta(x - x_0)\delta(z - z_0) \) to make \( p \) properly normalized. Under these
conditions, it can be shown that \( w \) is the steady-state marginal probability distribution
for the colloid across \( x \) (see Appendix 3.A). In Eq. 3.3, we have assumed that \( D_z \)
is independent of the colloid’s \( z \) position, which is reasonable for an infinite channel.
However, hydrodynamic interactions with the channel walls significantly influence
diffusion as a function of transverse position \([104]\), and so \( D_x \) and \( D_z \) are taken to
be functions of \( x \).

We seek the long-time dispersivity \( K \) that quantifies the average spreading of
the particle distribution, which is expected to have a contribution due to Brownian
motion and an enhancement due to flow. We define \( K \) using the Einstein relation
for a Brownian particle in a moving frame of reference

\[
K = \frac{1}{2} \lim_{t \to \infty} \frac{d}{dt} \langle (z - \langle z \rangle)^2 \rangle, \tag{3.5}
\]

where the brackets indicate the total expectation taken using \( p(x, z, t) \). Performing
a moments analysis in the spirit of Aris \([50]\) and Brenner and Gaydos \([105]\) (see
Appendix 3.A for details) yields the dispersivity

\[
K = \overline{D}_z + \int_{-\ell/2}^{\ell/2} \frac{dx}{w(x)D_z(x)} \left[ \int_{-\ell/2}^{x} (u(\hat{x}) - \overline{u})w(\hat{x})d\hat{x} \right]^2, \tag{3.6}
\]

where \( \overline{D}_z \) is the average axial diffusion coefficient

\[
\overline{D}_z = \int_{-\ell/2}^{\ell/2} D_z(x)w(x)dx, \tag{3.7}
\]

and \( \overline{u} \) is the mean velocity experienced by the colloid

\[
\overline{u} = \int_{-\ell/2}^{\ell/2} u(x)w(x)dx. \tag{3.8}
\]
The dispersivity $K$ is then fully determined by knowledge of only a few important parameters: (1) the steady-state colloid distribution $w$, (2) the velocity profile $u$, and (3) the diffusion tensor $D$. Theoretical methods for estimating these parameters will be discussed in detail in Section 3.3.

### 3.2 Simulation Model and Methods

In order to test the theoretical prediction of Eq. 3.6, we performed molecular dynamics (MD) simulations of colloidal suspensions under flow. Explicit-solvent molecular dynamics incorporates hydrodynamic effects, energetic interactions between particles, and thermal fluctuations. The solvent and colloids were modeled as spherical particles interacting through the Lennard-Jones potential

$$U_{ij}(r) = 4\varepsilon_{ij} \left[ \left( \frac{\sigma_{ij}}{r} \right)^{12} - \left( \frac{\sigma_{ij}}{r} \right)^{6} \right],$$  

where $r$ is the distance between particles of types $i$ and $j$, $\varepsilon_{ij}$ sets the interaction strength, and $\sigma_{ij}$ sets the range of the interaction. Unless stated otherwise, the potential was truncated at $r_{\text{cut}} = 3\sigma_{ij}$ with a smoothing polynomial applied from $2.5\sigma_{ij}$ so that both the energy and force were zero at $r_{\text{cut}}$. In what follows, all quantities are reported in reduced units derived using the solvent–solvent interaction energy scale, $\varepsilon_{ss} = 1$; solvent–solvent interaction length scale, $\sigma_{ss} = 1$; and solvent particle mass, $m_s = 1$, as the fundamental units. For example, the derived unit of time is $\sqrt{m_s \sigma_{ss}^2/\varepsilon_{ss}}$ in this system of units.

Neutrally buoyant colloids were density-matched to the solvent by setting their mass $m_c = \pi \rho_s a^3/6$, where $\rho_s$ is the solvent density and $a$ is the effective diameter of the colloids (see Section 3.3.1). The interactions between colloids were modeled through the Weeks–Chandler–Anderson (WCA) potential \[83\], which is obtained by truncating Eq. 3.9 at its minimum, $r_{\text{cut}} = 2^{1/6} \sigma_{ij}$, and shifting it by $\varepsilon_{ij}$. The
colloid–colloid interaction length was set to $\sigma_{cc} = 3$, and the interaction strength was set to $\varepsilon_{cc} = 10$ to reduce possible overlap between colloids. Colloid–solvent cross-interactions were modeled using the Lennard-Jones potential with $\sigma_{cs} = (\sigma_{cc} + \sigma_{ss})/2 = 2$ and $\varepsilon_{cs} = 1$.

We confined our colloidal suspension between two planar walls in the $xy$-plane using a similar model to that of Khare and co-workers [119, 120]. The walls were constructed by placing particles into one layer of a face-centered cubic crystal with lattice spacing 1.3 at $x = \pm 12$ and fixing their positions with stiff harmonic springs (spring constant 500). Wall particles interacted with each other through the Lennard-Jones potential with $\sigma_{ww} = 1$ and $\varepsilon_{ww} = 1$. Wall–solvent interactions were similarly treated with $\sigma_{sw} = 1$ and $\varepsilon_{sw} = 1$. The cross-interactions with the colloid were modeled using the purely repulsive WCA potential with $\sigma_{cw} = 2$ and $\varepsilon_{cw} = 1$. With this choice of parameters, the lattice spacing gave a nearest-neighbor distance smaller than $\sigma_{ww}$, which achieved two important effects: (1) the suspension could not penetrate the walls, and (2) the walls became microscopically rough, which is essential to establish no-slip boundary conditions. A representative simulation snapshot is shown in Figure 3.2.

![Figure 3.2](image_url)

Figure 3.2: Simulation snapshot [121] of the parallel plate channel for colloid volume fraction $\Phi = 0.1$. The coordinate axis is oriented as in Figure 3.1.

Despite the purely repulsive colloid–wall interactions, we observed that the colloids would sometimes “stick” to the channel walls, which limited our ability to
statistically sample the entire colloid distribution during the accessible simulation
time. We attribute this effect to structuring of the Lennard-Jones solvent and de-
pletion forces that resulted in a weak net attraction of the colloids to the walls. In
a physical suspension, there may be additional repulsive forces between the channel
walls and the colloids, e.g., electrostatics, which counteract these attractions. The
proposed theory is general and is able to accommodate such repulsive external fields
imposed on the colloids. Accordingly, we introduced an additional purely repulsive
9–3 potential acting only on the colloids

\[ U_r(x) = \varepsilon_r \left[ \frac{2}{15} \left( \frac{1}{|x-x_r|} \right)^9 - \left( \frac{1}{|x-x_r|} \right)^3 \right], \]

(3.10)

where \( x \) is the transverse position of the colloid, \( \varepsilon_r = 144 \) is the interaction strength,
and the walls have been placed at \( x_r = \pm 10 \). The potential was truncated at its
minimum, \( x_{\text{cut}} = (2/5)^{1/6} \), and shifted by \( \varepsilon_r \sqrt{10}/3 \). This potential effectively reduced
the width of the channel that the colloid could explore by about 10%. We emphasize
that this potential has been chosen as a matter of simulation convenience, and other
wall interactions (including attractive surfaces) can also be incorporated into the
theory through \( \phi \) in Eq. 3.4.

We performed molecular dynamics simulations using the HOOMD-blue simula-
tion package \[122, 123\] with double-precision floating-point operations to minimize
momentum drift during long simulations. The equations of motion were integrated
using the velocity Verlet algorithm with a timestep \( \Delta t = 0.0025 \). Due to the size
asymmetry between the colloid and solvent particles, the pair force calculations were
accelerated with a binary tree neighbor search (see Appendix A) \[124\]. Pressure-
driven flow was generated by applying a constant gravitational body force \( F_g = m_i \cdot g \)
to all particles in the dispersion along the flow direction with acceleration constant \( g \).
The non-equilibrium flow of the suspension generates heat that must be dissipated
to maintain the fluid at a constant temperature. In order to minimally perturb the
dynamics of the suspension, a Langevin thermostat was applied to the wall particles
with friction factor $\gamma = 4$, essentially turning the walls into heat sinks. The suspen-
sion was not explicitly thermostatted. All values of $g$ investigated in this work were
sufficiently small so that the excess heat was efficiently dissipated through the walls
and the system remained at the targeted temperature $T = 2.5$.

The suspensions were initialized by placing the colloid and solvent particles ran-
domly on a lattice between the parallel plates in order to give a suspension density
of approximately $\rho = 0.62$. We applied periodic boundary conditions in the vorticity
($y$) and axial ($z$) dimensions of our simulations with box lengths $L_y = L_z = 29.9$.
This kept the suspensions at one effective average concentration throughout the
simulation, making it possible to systematically identify concentration effects on
the dispersion. We define the concentration from the colloid volume fraction $\Phi =
N_c \pi a^3/(6L_x L_y L_z)$, where $N_c$ is the total number of colloids in the system and $L_x$
is the effective channel width set by the atomistic walls (discussed in Section 3.3.1).
The number of colloid and solvent particles simulated for each concentration are
summarized in Table 3.1.

Table 3.1: Number of colloids, $N_c$, and solvent particles, $N_s$, in the channel for
targeted volume fraction, $\Phi$, and suspension density $\rho = 0.62$. $D$ is the simulated
bulk diffusion coefficient of the suspension extrapolated to infinite box size. $\overline{D}_z$ is the
simulated axial diffusion coefficient at rest in the parallel plate channel. Subscripts
on reported values indicate the measurement uncertainty in the last digit.

<table>
<thead>
<tr>
<th>$\Phi$</th>
<th>$N_c$</th>
<th>$N_s$</th>
<th>$D$</th>
<th>$\overline{D}_z$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.005</td>
<td>6</td>
<td>12822</td>
<td>0.126</td>
<td>0.116$_1$</td>
</tr>
<tr>
<td>0.025</td>
<td>30</td>
<td>12619</td>
<td>0.117</td>
<td>0.107$_1$</td>
</tr>
<tr>
<td>0.050</td>
<td>61</td>
<td>12356</td>
<td>0.110</td>
<td>0.097$_1$</td>
</tr>
<tr>
<td>0.075</td>
<td>91</td>
<td>12101</td>
<td>0.103</td>
<td>0.089$_0$</td>
</tr>
<tr>
<td>0.10</td>
<td>122</td>
<td>11838</td>
<td>0.096</td>
<td>0.083$_1$</td>
</tr>
</tbody>
</table>
We equilibrated the suspensions for $10^5$ MD steps at rest, and performed $5 \times 10^5$ MD steps to allow the flow to reach steady-state before performing measurements. We recorded the position and velocity of each colloid every 50 MD steps, and conducted between 4 and 16 independent simulations for every colloid concentration and flow rate considered, each running at least $1.5 \times 10^8$ MD steps. We measured the mean-squared displacement (MSD) of each colloid in the frame of reference of the average velocity of all colloids, $\langle (z - \bar{u}t)^2 \rangle$, because at long times the mean of the distribution advects with the average colloid velocity (see Appendix 3.A). The dispersivity can be computed from the first derivative of the MSD with respect to time using Eq. 3.5. Figure 3.3 shows the results of this procedure for $\Phi = 0.005$ at increasing $g$, where the first derivative of the MSD has clearly converged to a limiting value. We determined $K$ by fitting over this plateau region. We emphasize the need to perform multiple independent simulations, particularly for dilute concentrations, over long times in order to obtain a reliable measurement of $K$.

![Figure 3.3](image)

Figure 3.3: Time derivative of the MSD along the flow direction of the channel for a dilute colloid suspension with $\Phi = 0.005$ at increasing $g$. The dark lines are averages over multiple independent simulations, while the shaded regions correspond to the measurement uncertainty determined from the standard error of the independent simulations (less than 2% of average values).
3.3 Results and Discussion

The colloid distribution, velocity profile, and diffusion coefficients for our simulated model must be obtained to compare Eq. 3.6 against the direct simulation results. Ideally, these parameters could all be obtained theoretically so that predictions can be made independently of the simulations. In the present work, we estimate parameters for a dilute suspension of colloids with perfect slip because these are the conditions obtained in our simulations. In a physical system, the colloids are likely to have either no-slip or partial slip boundary conditions, which may modify the effective colloid velocity and the diffusion coefficients in confinement. These differences can be accommodated by appropriate modification of the theory presented here.

First, we characterize properties of our simulated system such as channel size, fluid properties, and colloid diameter. Then, we theoretically estimate the colloid distribution, flow profiles, and diffusion coefficients, and compare our predictions directly against the simulation results when possible. Finally, we incorporate these results into Eq. 3.6 to compare dispersion models of increasing detail with our simulation results.

3.3.1 Model characterization

We investigated the flow behavior of the pure solvent in the parallel plate channel. We placed \( N_s = 12873 \) solvent particles randomly into the channel. We performed simulations for different acceleration constants, \( 0.002 \leq g \leq 0.010 \), and measured the steady-state solvent velocity profile. Figure 3.4 shows a representative velocity profile of the pure solvent at \( g = 0.002 \), which clearly has the characteristic parabolic shape expected for pressure-driven flow of a Newtonian fluid between parallel plates,

\[
u(x) = \frac{3}{2} U \left[ 1 - \left( \frac{2x}{L_x} \right)^2 \right], \tag{3.11}\]
where \( U \) is the mean solvent velocity. We fit \( U \) and \( L_x \) to the simulated velocity profiles for the different flow rates, and found \( L_x = 23.4 \pm 0.1 \) independent of \( g \), from which we compute an effective average solvent density of \( \rho_s \approx 0.62 \). The measured \( L_x \) is slightly smaller than the nominal distance at which the walls were placed due to the roughness of the walls and the excluded volume between the solvent and wall particles. In order to determine the solvent viscosity, we used the relationship between \( U \) and the fluid properties in this geometry,

\[
U = \frac{\rho_s g L_x^2}{12 \eta_s},
\]

and found \( \eta_s = 0.99 \pm 0.02 \) in excellent agreement with the value of 1.00 \( \pm 0.03 \) interpolated from available equilibrium data for the Lennard-Jones fluid [125, 126].

![Figure 3.4: Velocity profile of the pure solvent at \( \rho_s = 0.62, T = 2.5, \) and \( g = 0.002 \).](image)

In order to density-match the colloid to the solvent, we needed to identify the effective colloid diameter \( a \). We tentatively set the mass of the colloid using \( a \approx \sigma_{cc} \) and \( \rho_s = 0.62 \). We then performed bulk simulations of a single colloid in a cubic simulation box of edge length \( L = 20 \) at the same effective density as in the parallel plate channel. In contrast to the suspensions in the parallel plate channels, isothermal conditions were achieved by applying a Nosé–Hoover thermostat to the suspension with time constant \( \tau_{NH} = 0.5 \). We measured the equilibrium diffusion coefficient
of the colloid by computing its three-dimensional MSD, $\langle \Delta r^2 \rangle$, as a function of time. The simulated diffusion coefficient $D_L$ is related to the measured MSD by $\langle \Delta r^2 \rangle \sim 6D_L t$ at sufficiently long times. We use the symbol $D_L$ to emphasize the fact that the diffusion coefficient measured in a simulation is system-size dependent and needs to be corrected for finite-size effects, as explained later in this section. Figure 3.5 shows $\langle \Delta r^2 \rangle$ averaged from four independent runs. We can see that the motion of the colloid is ballistic at short times and then becomes fully diffusive after $t \gtrsim 30$, where the MSD achieves a slope of 1. We extracted a diffusion coefficient of $D_L = 0.107 \pm 0.005$ from this diffusive regime.

![Figure 3.5: Mean-squared displacement $\langle \Delta r^2 \rangle$ of a single colloid versus time $t$.](image)

Due to the long-ranged nature of hydrodynamic interactions, the simulated $D_L$ must be corrected for finite size effects of the periodic simulation box in order to obtain the true bulk diffusion coefficient $D$. We employed the correction derived by Yeh and Hummer [127]

$$D = D_L + \frac{\xi k_B T}{6\pi \eta L},$$

(3.13)

where $\xi \approx 2.837297$ for a cubic simulation box and $\eta$ is the viscosity. We used $\eta \approx \eta_s$ for a single colloid, leading to a bulk diffusion coefficient $D = 0.126 \pm 0.005$. We verified the reliability of this correction by performing additional simulations in a simulation box with $L = 30$, and found $D_L = 0.114 \pm 0.004$ and $D = 0.127 \pm 0.004$. 45
For a single colloid, we also expect the diffusivity to be given by the Stokes–Einstein relation \( D = \mathcal{M} k_B T \), where \( \mathcal{M} \) is the mobility. Because the colloids are modeled as smooth spherical particles, the solvent is able to exert normal forces but no torques. Hence, we expect the colloids to have perfect slip boundary conditions, \( \mathcal{M} = 1/(2\pi\eta a) \). We can then estimate \( a \approx 3.2 \) using \( \eta \approx \eta_s \). This estimate is consistent with our simulation model because the Lennard-Jones potential should exclude slightly more volume than the sphere defined by \( \sigma_{ij} \) because the potential’s minimum is at \( 2^{1/6}\sigma_{ij} \). Accordingly, we set the mass of our colloids to \( m_c = 10.64 \) in all simulations using \( \rho_s = 0.62 \).

We then additionally measured the bulk diffusion coefficient \( D \) of our suspensions (edge length \( L = 30 \)). \( N_c \) and \( N_s \) were appropriately adjusted for the larger volume of the simulation box compared to the slit channels. Because the suspension viscosity is not known \textit{a priori}, we instead corrected our data for finite size effects of the periodic simulation box using the expression from Yeh and Hummer [127] for particles with slip boundary conditions that is independent of the suspension viscosity

\[
D \approx D_L \left(1 - \frac{\xi a}{3L}\right)^{-1}.
\]  

(3.14)

The suspension diffusion coefficients reported in Table 3.1 decrease as a function of concentration. We qualitatively expect that this is due to an increase in the suspension viscosity. Cichocki and Felderhof [128] showed that the viscosity for suspensions of slip colloids has an order \( \Phi^2 \) correction to the intrinsic viscosity:

\[
\eta(\Phi) = \eta_s \left(1 + 2.5\Phi + 1.911\Phi^2\right),
\]

(3.15)

which increases with concentration. We surprisingly noted that the intrinsic viscosity of our suspensions was best represented by the no-slip (Einstein) viscosity [129, 130],
even though the colloid diffusion was shown to be more consistent with slip boundary conditions. This point warrants further investigation.

We repeated the correction of the measured suspension diffusion coefficients using Eq. 3.13 with the viscosity given by Eq. 3.15. We found quantitative agreement with the results obtained by Eq. 3.14, indicating that Eq. 3.15 is a reasonable constitutive equation for our simulated suspensions. We also compared $D$ with the values that would be predicted by the Stokes–Einstein relation for the suspension, and found deviations of less than 4% over the range of concentrations we have considered.

### 3.3.2 Colloid distributions

Colloidal mixtures of sufficiently high concentration adopt non-uniform density profiles in confinement. It is well-known from classical statistical mechanics that this effect is present even for a simple purely repulsive hard sphere fluid against a structureless flat wall [101–103]. In this case, the fluid density is typically higher near the surface, and decays to the bulk density far away. The structuring near the wall occurs only due to the excluded volume interactions both within the hard sphere fluid and against the wall, and to our knowledge has not been accounted for in previous theories of axial dispersion. We can incorporate this inhomogeneous particle distribution into our current model through an “effective” external potential $\phi$ that modifies the distribution $w$ according to Eq. 3.4.

Various statistical mechanical methods are available to predict the inhomogeneous density profiles of simple liquids [131]. In particular, classical density functional theories [132, 133] and direct molecular simulations, e.g. Monte Carlo or MD, have been applied to predict the density profiles of fluids near surfaces [103, 134, 135] using only knowledge of the interparticle interactions. MD is obviously most expedient to determine the equilibrium particle distribution in the current work. However, we emphasize that this profile could also be determined theoretically by alterna-
tive computational methods or experimentally through appropriate particle tracking measurements.

The purely repulsive WCA colloids in our model can be approximately mapped onto the hard sphere fluid \([83]\), and so are expected to exhibit similar structuring near the walls in confinement. Figure 3.6 shows the steady-state colloid distribution along the transverse direction, \(w\), at increasing \(\Phi\) both at rest and at \(g = 0.010\). At the lowest concentration, \(\Phi = 0.005\), the colloids are nearly uniformly distributed between the walls, but then gradually start to aggregate at the edges as \(\Phi\) is increased.

As discussed in Chapter 2, particles can undergo cross-stream inertial migration in Poiseuille flow \([67]\), even at small but finite Reynolds number \([136]\). Inertial migration could lead to the focusing of particles onto planes near the walls in the parallel plate geometry, further enhancing the structuring observed in \(w\). It has been shown that inertial migration effects are significant when the product of the particle Reynolds number, \(\text{Re}_c\), and particle Péclet number, \(\text{Pe}_c\), is greater than unity \([108]\). For our suspensions, we estimate \(\text{Re}_c \sim 0.4\) and \(\text{Pe}_c \sim 0.4\) using the definitions in ref. 136, and so we expect that inertial effects should not be significant and any structuring should be due only to the colloid interactions in confinement. We confirmed this by repeating the measurement of \(w\) under flow at \(g = 0.010\) (the maximum flow rate considered) and found that the profiles are essentially indistinguishable. Accordingly, the equilibrium particle distribution may be applied even under flow for the concentrations and flow rates we have considered.

### 3.3.3 Velocity profiles

At steady state, the colloid velocity profile should approximately satisfy the one-dimensional Navier–Stokes equation for gravity-driven flow

\[
\frac{d}{dx} \left[ \eta(\Phi(x)) \frac{du}{dx} \right] + \rho(x)g = 0 \quad (3.16)
\]
with no-slip boundary conditions at the walls, \( u(x = \pm L_x/2) = 0 \). We neglect any additional flows between the colloids and the walls, and any lag in the colloid velocity relative to the solvent. Here, \( \rho(x) \) is the density of the suspension with contributions due to the solvent and the colloids, which may in principle depend on transverse position due to the confinement. Because the colloids are density-matched to the solvent, we can approximate the total suspension density as a uniform distribution

\[
\rho(x) \approx \rho = \frac{m_s N_s + m_c N_c}{L_x L_y L_z}.
\]  

The suspension viscosity \( \eta \) is a function of concentration, which depends on the colloid volume distribution. The colloid volume distribution can be computed by convolving \( w \), which effectively gives the distribution of the colloid center of mass, with the volume of a colloid in three dimensions,

\[
\Phi(x) = \frac{N_c}{L_y L_z} \int w(x - x') \Theta(a/2 - |r'|)dr',
\]  

Figure 3.6: Probability \( w \) to find a colloid at a given \( x \)-position at rest (lines) and for \( g = 0.010 \) (symbols) for increasing colloid concentrations \( \Phi \). Only every second point is drawn for \( g = 0.010 \). Error bars are omitted for clarity, but are no larger than \( \pm 10\% \) for all points.
where $x'$ is the $x$ component of $r'$, $|r'|$ is the magnitude of $r'$, and $\Theta$ is the Heaviside step function. The resulting distributions for $\Phi(x)$ are qualitatively similar to $w$, but are slightly smoothed near the channel walls.

We solved Eq. 3.16 numerically using Eq. 3.18 for $\Phi(x)$ with Eq. 3.15 for $\eta$. We emphasize that there are no free parameters used when solving Eq. 3.16. Figure 3.7 compares the steady-state colloid velocity profiles measured from flow simulations at $g = 0.010$ with the numerical solution. In all cases, the maximum velocities predicted by the numerical solution are in excellent agreement with the simulated data. At the highest concentrations, the curvature of the simulated velocity profiles is somewhat steeper than predicted by Eq. 3.16. This deviation in the near-wall region may be partially due to colloid–wall hydrodynamic interactions, which may reduce the colloid velocity. We will assess the importance of these deviations for computing the dispersion coefficient in Section 3.3.5. We did not observe any lag between the colloid velocity and that of the solvent in our simulations [100], consistent with numerical and experimental results for neutrally buoyant spherical particles of comparably sized particles in a parallel plate channel [137–139].

![Figure 3.7: Suspension velocity profiles $u$ at $g = 0.010$ for increasing colloid concentrations $\Phi$. Symbols are measured from simulations, and lines are calculated from Eq. 3.16. Error bars for the simulated data are smaller than the symbol size.](image)

50
In the limit that the colloids explore the channel uniformly, the velocity profile given by Eq. 3.16 and Eq. 3.18 should be essentially parabolic. However, it is evident that the velocity profiles in Figure 3.7 are not parabolic due to the spatially varying viscosity of the suspension. Hence, it is important to consider the effects of confinement on the velocity profile when the density profile is highly non-uniform or when there are strong repulsions between the colloids and the channel walls that exclude the colloid from accessing significant regions of the channel.

3.3.4 Diffusion coefficients

Considerable theoretical effort has been undertaken to predict the diffusion coefficient of bulk colloidal suspensions [128, 140, 141]. Incorporating the effects of confinement on the suspension is an even greater theoretical challenge. In the present work, we approximate the diffusion coefficient in confinement from results for the Stokes flow of a single colloid. Wacholder and Weihs exactly solved the enhancement to the drag on a spherical fluid droplet moving perpendicular to a single planar wall [142], which affects the observed transverse diffusion coefficient $D_x$. From their work, the mobility of a spherical particle with perfect slip boundary conditions is reduced by a factor $1/\lambda_1$, where

$$
\lambda_1(d) = \sinh \alpha \sum_{n=1}^{\infty} \frac{n(n + 1)}{(2n - 1)(2n + 3)} \left[ \frac{(2n + 3)e^{2\alpha} + 4e^{-(2n+1)\alpha} - (2n - 1)e^{-2\alpha}}{2 \sinh(2n + 1)\alpha - (2n + 1) \sinh 2\alpha} \right],
$$

(3.19)

$d$ is the center-of-mass separation from the wall, and $\alpha = \cosh^{-1}(2d/a)$. In the limit of large separations, the unperturbed mobility is recovered. Using the method of reflections [143–145], Eq. 3.19 can be used to approximate the drag on a particle in a channel by superimposing the effects of two walls at $x = 0$ and $x = L_x$. The particle
mobility at center of mass position $x$ is reduced by $1/\lambda_2$, where

$$
\lambda_2(x) \approx \lambda_1(x) + \lambda_1(L_x - x) - 1.
$$

(3.20)

The transverse diffusion coefficient of the confined suspension is then approximately related to the bulk diffusion coefficient by $D_x(x; \Phi) \approx D(\Phi)/\lambda_2(x)$. $D$ can be determined by measurement or an appropriate theoretical model. In this work, we will use the measured bulk diffusion coefficients of Table 3.1, which are in good agreement with Stokes–Einstein predictions.

Chang and Keh have used a semi-analytical boundary colocation method to exactly solve the drag on a spherical fluid droplet moving perpendicular to two planar walls [146]. Equation 3.20 overestimates the drag they reported for a particle with perfect slip boundary conditions on the centerline by about 5%. However, their solution is considerably more computationally demanding to evaluate than Eq. 3.19. Moreover, we have already made a more significant approximation that the result for a single particle can be applied to the suspension of many particles. In reality, there are additional flows (reflections) between particles that may incur error at higher particle concentrations.

We attempted to validate Eq. 3.20, but found that it was challenging to measure $D_x$ directly in our simulations. We observed in the bulk that a colloid translated a significant fraction of the accessible channel width before it entered the diffusive regime (see Figure 3.5). During this time, the particle is not a purely Brownian walker, and so standard techniques for evaluating the diffusion coefficient are unsuitable. We accept Eq. 3.20 as a reasonable approximation for lack of an alternative theory for the simultaneous effects of confinement and concentration on diffusion of a suspension, and will later validate its usefulness from the axial dispersion measurements.
The axial diffusion coefficient $D_z$ should also be affected by confinement to some extent [146, 147]. However, the periodic boundary conditions of our simulation box may also affect the observed diffusion, as they do in the bulk, making it challenging to theoretically predict this quantity for the simulated system. Since $D_z$ amounts only to a constant offset in the overall dispersivity and can be determined experimentally by appropriate extrapolation of $K$ as $g$ tends to zero, we simply measure the effective diffusion coefficient $D_z$ for the suspension at rest (see, for example, $g = 0.000$ in Figure 3.3), and report the diffusivity in Table 3.1. Typically, $D_z$ is roughly 85% to 90% of $D$ for our suspensions. If it is not possible to determine $D_z$, a theoretical prediction can be obtained in a manner analogous to Eq. 3.19 and Eq. 3.20 for motion parallel to a planar surface (see, for example, Ref. 147).

### 3.3.5 Axial dispersion

The results derived so far can be applied in Eq. 3.6 to develop models of increasing detail for the axial dispersion. We start from the classical Taylor–Aris case in the parallel plate channel, in which point particles access the entire channel uniformly so $\ell = L_x$ and $w(x) = 1/L_x$. The classical parabolic velocity profile of Eq. 3.11 is assumed. The diffusion coefficients are taken to be isotropic so that $D_x = D_z = D$.

Taking the integrals in Eq. 3.6 and Eq. 3.8 gives the well-known result [148]

$$K/D = 1 + \frac{\text{Pe}^2}{210},$$

(3.21)

where the Péclet number is defined as $\text{Pe} = U L_x/D$.

Using geometric considerations, James and Chrysikopoulos [107] derived an expression for the dispersivity of rigid colloids between parallel plates when the colloids are excluded from part of the channel but otherwise remain uniformly distributed. The assumptions related to the velocity profile and isotropic diffusion remain un-
changed. The dispersivity is then

\[ K/D = 1 + \frac{Pe^2}{210} \left( \frac{\ell}{L_x} \right)^6. \]  

(3.22)

We can relax the assumption of isotropic diffusion in Eq. 3.22 by replacing the transverse diffusion coefficient by its average value in the accessible channel width, \( D_x \). This assumption is reasonable when the variations in \( D_x \) across the accessible channel width are not too large. From this, we analogously obtain

\[ K/D_z = 1 + \frac{Pe^2}{210} \left( \frac{\ell}{L_x} \right)^6, \]  

(3.23)

where \( Pe = UL_x/(D_x D_z)^{1/2} \). We expect that these approximate models should be less accurate than the full prediction of Eq. 3.6.

Table 3.2 summarizes the equations and parameterizations used in these models in order to make direct comparison with simulation data. Model A is the classical Taylor–Aris result for the parallel plate channel, in which we set \( D = D_z \) so that \( K/D_z = 1 \) at rest. Model B incorporates the effects of the accessible channel width to Model A, but still assumes isotropic diffusion. Model C adds anisotropic diffusion through Eq. 3.23. Finally, Model D fully incorporates the non-uniform colloid distribution into Eq. 3.6 through \( w \) and \( u \), and so includes all the parameters and effects considered so far in this work. In Models A, B, and C, the characteristic velocity \( U \) is calculated using Eq. 3.11 with the suspension density and viscosity. In Models B and C, we set the accessible width \( \ell \approx 18.2 \), estimated from the accessible region shown in Figure 3.6 and the expected range of the repulsive wall potential acting on the colloids. We obtain \( D_x \) for Model C by taking an average of \( D(x) \) with the uniform distribution so that \( D_x \approx 0.75D \).

All models, including Model D, predict enhancements that increase quadratically with the acceleration constant \( g \) because \( K \sim U^2 \) in Eq. 3.6 and \( U \sim g \) from Eq. 3.16.
Table 3.2: Theoretical models for axial dispersion.

<table>
<thead>
<tr>
<th>model</th>
<th>reference</th>
<th>equation</th>
<th>parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>148</td>
<td>3.21</td>
<td>( D = \overline{D}_z )</td>
</tr>
<tr>
<td>B</td>
<td>107</td>
<td>3.22</td>
<td>( D = \overline{D}_z, \ell = 18.2 )</td>
</tr>
<tr>
<td>C</td>
<td>this work</td>
<td>3.23</td>
<td>( D_x = \overline{D}_x, \ell = 18.2 )</td>
</tr>
<tr>
<td>D</td>
<td>this work</td>
<td>3.6</td>
<td>( D_x = D_x(x), w(x) ) from Figure 3.6</td>
</tr>
</tbody>
</table>

Figure 3.8 compares the theoretical predictions of the different models to the simulated enhancements \( K/\overline{D}_z \) as a function of \( g \) for the most dilute concentration, \( \Phi = 0.005 \). At this concentration, the colloids are nearly uniformly distributed in the channel, and so Models B and C are expected to perform their best at these conditions. It is apparent that Model A vastly overpredicts the enhancement. This should not be surprising, based on the strong dependence of Eq. 3.22 and Eq. 3.23 on the accessible channel width, which predict an \( \approx 80\% \) reduction in the dispersivity due to the exclusion of the colloid from the slowest parts of the velocity profile. The remaining models are quantitatively much closer to the simulated enhancements. Model C differs from Model B by a factor of \( \overline{D}_z/\overline{D}_x \), which is typically greater than 1 for our simulated suspensions (and more generally for a single particle between two parallel plates), and so Model C consistently predicts a larger enhancement than Model B. For the dilute case, Model D gives the best quantitative prediction of the dispersion.

In order to assess which model performs best across multiple concentrations, we compare the predicted relative enhancements as a function of increasing concentration at fixed \( g = 0.010 \), shown in Figure 3.9. Model A clearly overpredicts the relative enhancement for all concentrations by a large factor. This overprediction is in agreement with the numerical results of Sané et al. [118]. The predictions of Models B and C are much closer to the simulated enhancements, but fail at capturing the correct concentration dependence of \( K/\overline{D}_z \). Only Model D is able to produce reason-
Figure 3.8: Comparison of predictions of different models for the enhancement of the axial diffusion, $K/D_z$, as a function of acceleration constant $g$ at $\Phi = 0.005$. Model predictions are given by labeled curves, while the simulated values are plotted as points. Error bars for the simulated values are smaller than the symbol size. See Table 3.2 for model details.

Figure 3.9: Comparison of predictions of different models for the enhancement of the axial diffusion, $K/D_z$, as a function of concentration $\Phi$ at fixed $g = 0.010$. Model predictions are given by labeled curves, while the simulated values are plotted as points. Error bars for the simulated data are smaller than the symbol size. The dashed line corresponds to prediction of Model D with the velocity profiles from Eq. 3.16 replaced by the simulated velocity profiles of Figure 3.7.

Although Model D captures the qualitative concentration dependence of $K/D_z$ best from the four models considered, it consistently underpredicts the dispersion by a small amount at the highest concentrations. We attribute this error to the incorrect curvature of the velocity profiles predicted by Eq. 3.16 at the higher concentrations.
in the near-wall regions. Qualitatively, we expect that the dispersion should increase if the velocity gradient is steeper because the colloids advect faster relative to the mean. This error is further amplified at the higher concentrations because the colloid distribution is biased towards the wall region, where the error in the velocity is largest. We confirmed this source of error by recomputing the dispersion coefficient using Model D with the velocity profiles from the simulations. The predicted enhancement is shown as a dashed line in Figure 3.9, and is in quantitative agreement with the simulated data. This indicates the sensitivity of the dispersion prediction to small errors in the velocity, especially near the channel walls, and the importance of reliably predicting the colloid velocity profile.

Although not shown in Figure 3.9, we also replaced $D_x = \overline{D}_x$ in Model D, and observed only a negligible difference in the predicted enhancements. Hence, the most significant difference between Model C and Model D is the colloid distribution. We conclude that the colloid distribution is the most significant factor controlling the concentration dependence of the enhanced dispersion. This makes physical sense given that the colloid distribution controls how the colloid explores the streamlines of the velocity profile that drive dispersion.

Consider two extreme cases. There is no enhancement to the dispersion when the colloid is confined to a single streamline because it always advects with its mean velocity. In the opposite case, the enhancement is maximized if the particle is distributed in such a way that it spends half its time advecting at the maximum possible velocity, and the other half advecting at the slowest possible velocity. As concentration increases, peaks are obtained towards the extrema regions of the velocity profile. It is then crucial to obtain a reliable estimate for the distribution of the particles in the channel.

Figure 3.10 concisely highlights the results of the present work by comparing the simulated enhancement of the axial dispersion to the predictions of Model D for all
five concentrations considered as a function of flow rate. The agreement between the simulations and the predictions is good in all cases, and can be made completely quantitative by correcting for the simulated velocity profile (not shown here). The enhancement increases monotonically as a function of $\Phi$. From Eq. 3.6, the enhancement qualitatively scales with the Péclet number, $K/D \sim (UL/D)^2$. But, $U$ and $D$ should both scale in the same way with concentration due to their inverse dependence on the viscosity. Hence, we attribute the increase in the relative enhancement as a function of concentration to the increasingly non-uniform colloid distribution, which biases the colloids to the streamlines that have the largest deviation from the mean velocity, and controls the exact scaling prefactor.

![Figure 3.10: Enhancement of the axial dispersion, $K/D_z$, as a function of applied body force per mass $g$ for colloidal suspensions of increasing concentration $\Phi$. Symbols are simulation results, while lines are predictions from Model D. Error bars for the simulated data are smaller than the symbol size.](image)

Ultimately, however, all the effects of confinement on the colloid distribution and diffusion are relevant. Although the qualitative concentration dependence of Model C is not significantly different from Model B, there is a considerable quantitative difference due only to the anisotropic diffusion. Indeed, Model D would also fail to quantitatively predict the dispersion if the bulk diffusion coefficient were used rather than $D_x$. The fact that $D_x$ could be reasonably replaced by $\overline{D_x}$ in Model D suggests that Eq. 3.20 provides a reasonably estimate of $\overline{D_x}$, but the exact functional
form for $D_x$ is not necessary. This is promising for experiments seeking to determine the transverse diffusion coefficient by fitting dispersion data rather than make predictions.

3.4 Conclusions

We developed a predictive model for the axial dispersion of Brownian colloids in confinement. We showed that non-uniform colloid distributions that arise in confinement due solely to interparticle interactions significantly influenced the effective axial dispersion. The axial dispersion was also found to be sensitive to the colloid velocity profile near the channel walls and the anisotropic diffusion tensor. Our model gave good predictions for the flow rate and concentration dependence of the dispersion coefficients measured from explicit molecular dynamics simulations that fully take into account hydrodynamic correlations and thermal fluctuations. This model should prove useful in many applications involving the axial dispersion of colloids, including extracting diffusion coefficients from microfluidic experiments and modeling the transport of colloids in geological fractures.

In this chapter, we have restricted ourselves to analyzing dilute suspensions of Brownian colloids with slip boundary conditions. The presented theoretical framework can be easily extended to no-slip or partial slip boundary conditions on the colloids by appropriate modification of the expressions for the colloid velocity and diffusion coefficients. At high concentrations, the mean-field approximation that the motion of individual colloids is independent may break down due to correlations between particles. Moreover, at high concentrations or flow rates, shearing between colloids may cause the effective diffusion coefficients to have an additional flow-rate dependence. The approximations for the colloid velocity profile and diffusion coefficients may become less accurate for sufficiently large particle diameters relative
to the channel width due to increased colloid–wall interactions. Unfortunately, the employed simulation approach is not suitable to access these conditions due to two challenges: (1) diffusion slows considerably as concentration or particle diameter increases and it becomes difficult to reliably measure the dispersion, and (2) more heat is generated at high flow rates than can be reasonably removed through the channel walls alone and thermal gradients develop. These difficulties might be overcome with a mesoscale simulation approach to determine the concentrations and particle sizes at which deviations from the presented theory are observed.

3.A  Moment Analysis

Our derivation is adapted from that of Brenner and Gaydos in a cylindrical tube [105], and accordingly we present it in similar form and notation to allow the interested reader to compare. We define axial moments $\mu_m$ and total moments $M_m$ of the distribution

$$\mu_m = \int_{-\infty}^{\infty} (z-z_0)^m p \, dz, \quad M_m = \int_{-\ell/2}^{\ell/2} \mu_m \, dx,$$

and assume that $(z-z_0)^m p \to 0$ as $z \to \pm \infty$ in order that $\mu_m$ remains finite. By the normalization condition, $M_0 = 1$ on $p$. Using Eq. 3.5, the dispersivity can be computed from from the total moments

$$K = \frac{1}{2} \lim_{t \to \infty} \left[ \frac{dM_2}{dt} - 2M_1 \frac{dM_1}{dt} \right].$$

The problem then reduces to computing the first and second moments of the distribution. Taking the axial moments of Eq. 3.3 and integrating by parts gives

$$\frac{\partial \mu_m}{\partial t} = \mathcal{L}[\mu_m] + mu(x)\mu_{m-1} + m(m-1)D_z(x)\mu_{m-2},$$

60
where $\mathcal{L}$ is a compact notation for the operator

$$
\mathcal{L}[p] = \frac{\partial}{\partial x} \left( w(x) D_x(x) \frac{\partial}{\partial x} \left( \frac{p}{w(x)} \right) \right). 
$$

(3.27)

Explicitly, the first three moments are:

$$
\frac{\partial \mu_0}{\partial t} = \mathcal{L}[\mu_0] 
$$

(3.28)

$$
\frac{\partial \mu_1}{\partial t} = \mathcal{L}[\mu_1] + u(x) \mu_0 
$$

(3.29)

$$
\frac{\partial \mu_2}{\partial t} = \mathcal{L}[\mu_2] + 2u(x) \mu_1 + 2D_z(x) \mu_0 . 
$$

(3.30)

The transformed no-flux boundary conditions at $x = \pm \ell/2$ are

$$
j_{x,m} = -w(x) D_x(x) \frac{\partial}{\partial x} \left( \frac{\mu_m}{w(x)} \right) = 0 . 
$$

(3.31)

The transformed initial conditions are $\mu_0(x, t = 0) = \delta(x - x_0)$ and $\mu_m(x, t = 0) = 0$ for $m > 0$.

We begin by solving for $\mu_0$, which is essentially the marginal distribution of the colloid along the transverse channel dimension. The steady state solution that satisfies the normalization condition is $w(x)$, or equivalently, the Boltzmann distribution in the external field. This moment is expected to relax exponentially to its steady state. Since we are interested in long-time behavior, we neglect these transient terms and adopt a quasi-steady state approximation for $\mu_0(x, t) \approx w(x)$. This approximation is valid after a characteristic diffusion time $\tau \sim \ell^2 / D_x$. (Note that using this approximation it is no longer possible to satisfy the initial condition.)

We substitute $\mu_0$ into Eq. 3.29 and guess a solution of the form

$$
\mu_1 \sim w(x) (\mathbf{u} t + B(x)) 
$$

(3.32)
so that
\[
\frac{d}{dx} \left( w(x) D_x(x) \frac{dB}{dx} \right) = -(u - \bar{u})w(x). \tag{3.33}
\]

At the boundaries, \(-w(x)D_x(x)dB/dx = 0\), so integrating across the channel yields Eq. 3.8, and demonstrates that \(\bar{u}\) is precisely the average velocity of the colloid. Integrating twice allows us to evaluate \(B(x)\)

\[
B(x) = B(0) - \int_{-\ell/2}^{x} \frac{dx'}{w(x')D_x(x')} \int_{-\ell/2}^{x'} (u(\hat{x}) - \bar{u})w(\hat{x})d\hat{x}. \tag{3.34}
\]

The existence of such a solution validates the original guess for the functional form. The first total moment is

\[
M_1(t) \approx \bar{u}t + C, \tag{3.35}
\]

where we defined the constant \(C\) for convenience

\[
C \equiv \int_{-\ell/2}^{\ell/2} B(x)w(x)dx. \tag{3.36}
\]

The time derivative \(dM_1/dt \approx \bar{u}\), showing that the mean of the distribution advects with the mean colloid velocity, as it should. For the second (and higher) order moment(s),

\[
\frac{dM_m}{dt} = \int_{-\ell/2}^{\ell/2} \frac{\partial \mu_m}{\partial t} dx \tag{3.37}
\]

\[
= m \int_{-\ell/2}^{\ell/2} u(x) \mu_{m-1} dx 
+ m(m - 1) \int_{-\ell/2}^{\ell/2} D_z(x) \mu_{m-2} dx, \tag{3.38}
\]
which is obtained by substituting Eq. 3.26 and applying the boundary conditions on \( \mu_m \). For \( m = 2 \), substituting \( \mu_0 \) and \( \mu_1 \) and simplifying gives

\[
\frac{dM_2}{dt} = 2 \left( \pi^2 t + \int_{-\ell/2}^{\ell/2} u(x)w(x)B(x)dx + D_z \right), \tag{3.39}
\]

where \( D_z \) is defined by Eq. 3.7. Multiplying Eq. 3.29 by \( B(x) \) and integrating by parts allows us to evaluate the integral to give

\[
\frac{dM_2}{dt} \approx 2\pi^2 t + 2\pi C + 2D_z + 2 \int_{-\ell/2}^{\ell/2} \frac{dx}{w(x)D_x(x)} \left[ \int_{-\ell/2}^{x} (u(\hat{x}) - \pi)w(\hat{x})d\hat{x} \right]^2. \tag{3.40}
\]

Substituting Eq. 3.35 and Eq. 3.40 into Eq. 3.25 gives Eq. 3.6.
Chapter 4

Stratification in drying colloid mixtures

The previous chapters discussed colloidal dispersions driven out of equilibrium by flow. I now shift focus to another nonequilibrium process: film drying. Drying colloidal dispersions [1, 41, 149] are encountered in many technological processes, including applying latex paints [12, 13], ink jet printing [30, 31], manufacturing polymer LED displays [32, 33], and spraying pesticides [34]. In these technologies, colloidal particles are typically initially dispersed in a liquid solvent such as water. The dispersion may have a complex formulation, containing particles of different sizes and materials [14], surfactants [15], or polymers [16]. The solvent is evaporated from the dispersion, the colloid concentration increases until a close-packed structure is formed, and a solid coating or film is deposited [12, 13].

The drying process can significantly influence the structure of the deposited solid. Key parameters of this process include the evaporation rate and drying angle with respect to the substrate. For example, in the well-known “coffee-ring” effect, capillary flows within a pinned drying droplet force colloids outward to form a concentrated

ring-like deposit \cite{35-40}. Complex dynamics are observed during drying parallel to a substrate, including the formation of solidification, cracking, and dewetting fronts, which significantly influence the final morphology of the film \cite{41, 150}. Fast drying perpendicular to a substrate leads to an accumulation of colloids at the drying interface \cite{151-155}, often referred to as skin-layer formation. In colloidal mixtures with multiple particle types, differential rates of accumulation at the interface can induce a separation into distinct stratified layers under certain conditions \cite{41}, as schematically illustrated in Figure 4.1 for a binary mixture of particles with different sizes.

![Schematic illustration of stratification of a binary colloid mixture as observed by Fortini et al. \cite{51}. The initially well-dispersed mixture (left) stratifies on drying into a layer enriched in small colloids on top of a layer of big colloids (right).](image)

Figure 4.1: Schematic illustration of stratification of a binary colloid mixture as observed by Fortini et al. \cite{51}. The initially well-dispersed mixture (left) stratifies on drying into a layer enriched in small colloids on top of a layer of big colloids (right).

Such stratification may be advantageous, e.g., to create a multilayer coating during a single drying step, or undesirable, e.g., if a uniform particle coating is required. Nikiforow et al. \cite{156} showed that mixtures of charged and neutral particles can undergo a stratification process that causes the neutral particles to accumulate in a layer on top of the film. Trueman et al. observed segregation due to stratification in binary mixtures of latex particles \cite{157}. Typically, the large particles accumulated at the interface, but it was found that the stratification behavior depended on the evaporation rate, initial volume fractions, and particle size ratio. It is an important
challenge to model drying colloidal mixtures in order to effectively engineer their properties.

There have been several prior theoretical and simulation studies of drying of colloidal suspensions. Routh and Zimmerman modeled skin-layer formation for a suspension of monodisperse colloidal particles [151]. Central to their model is the film Péclet number, \( \text{Pe} = \frac{H_0 v}{D} \), which quantifies the relative contributions of advection and diffusion to particle motion. Here, \( H_0 \) is the initial film height, \( v \) is the typical speed of the interface, and \( D \) is the particle diffusion coefficient. When \( \text{Pe} \ll 1 \), diffusion dominates, and particles remain uniformly distributed in the film. When \( \text{Pe} \gg 1 \), however, particles accumulate at the drying front faster than they diffuse away, and a skin layer forms. Routh and Zimmerman’s theory predicts that the magnitude of the particle concentration gradient in the film should scale with \( \text{Pe}^{1/2} \).

Ekanayake et al. tested this prediction experimentally using magnetic resonance profiling [155], and found modest agreement (the gradient scaled with \( \text{Pe}^{0.8} \)). Trueman et al. extended the theory of Routh and Zimmerman to binary mixtures of colloidal particles of different sizes [158]. Each component \( i \) was assigned a Péclet number, \( \text{Pe}_i \), that depended on the diffusion coefficient \( D_i \). Larger particles have larger \( \text{Pe}_i \) since \( D_i \) is expected to decrease with increasing particle diameter according to the Stokes–Einstein relation. A stratified layer of bigger colloids was predicted to form on top of a layer of smaller colloids, with optimal stratification occurring when the big particles had \( \text{Pe}_B > 1 \) and the small particles had \( \text{Pe}_S < 1 \). Atmuri et al. further extended this work to mixtures of charged particles [159]. Reasonable qualitative agreement was obtained between model predictions and experiments in this regime [157, 159].

Recently, Fortini et al. studied the stratification of a binary mixture of neutrally buoyant, purely repulsive spheres with \( \text{Pe}_S > 1 \) and \( \text{Pe}_B > 1 \) in both experiments and computer simulations [51]. They observed a peculiar inverted stratification with
a layer of the smaller colloids segregated to the top of the dried film and the larger colloids pushed to the bottom. The experiments of Trueman et al. suggested similar stratification behavior only for large particle size ratios (roughly 4:1 by diameter) [157]. Such a stratification of small colloids on top of big colloids was not predicted by their theoretical model [158], and Trueman et al. hypothesized that this effect may have stemmed from depletion forces and sedimentation in the experiments that were not accounted for in the theory [157]. Cheng and Grest observed a similar inverted stratification in molecular simulations of the dispersion of colloids into a polymer film, with a polymer skin layer formed on top of the colloids at fast drying rates [54]. In order to explain the observed inversion, Fortini et al. proposed an alternative model based on a force balance between the osmotic pressure gradient and the drag force on a colloid [51]. Their model predicts that the migration velocity away from the drying interface is faster for larger colloids, and that the difference between the migration velocities of big and small colloids in a binary mixture is a power law of the particle diameter ratio. Qualitatively, this model correctly predicts that the small colloids should be found on top of the larger colloids. However, no quantitative validations of the model’s predictions, such as the power-law scaling of the migration velocities with the particle diameter, were reported, and the mechanism for the observed inverted stratification remains an open question.

In this work, the dynamics of stratification were investigated for a binary colloid mixture of varying diameter ratios in order to provide insight into the inverted stratification observed by Fortini et al. [51]. Initially, size ratios, drying rate, and initial colloid concentration comparable to the work of Fortini et al. [51] were studied using implicit-solvent molecular dynamics simulations, and inverted stratification with a layer of small colloids segregated on top of a layer of big colloids was observed. The particle size ratio and drying rate were then systematically varied over a significantly larger range of values than studied by Fortini et al. [51]. Detailed dynamics of the
inverted stratification were quantified for the first time from the colloid density and velocity profiles measured in the simulations. The stratified layer of small colloids was found to grow faster and to larger sizes for larger diameter ratios. Inverted stratification was found to occur even at moderate Péclet numbers (slower drying rates), $\text{Pe}_S \ll 1$ and $\text{Pe}_B \sim 1$, although the extent of stratification was much less compared to larger Péclet numbers (faster drying rates), $\text{Pe}_S \gg 1$ and $\text{Pe}_B \gg 1$.

A model based on dynamical density functional theory is proposed to capture the observed phenomena.

4.1 Model and Methods

A binary mixture of small (S) and big (B) spherical colloids with diameters $d_S$ and $d_B$ and masses $m_S$ and $m_B$ was studied. The masses were set proportionally to the colloid volumes, $m_B/m_S = (d_B/d_S)^3$. Netural colloids were modeled using the purely repulsive Weeks–Chandler–Andersen (WCA) potential [83],

$$U_{ij}(r) = \begin{cases} 
4\varepsilon \left[ \left( \frac{\sigma_{ij}}{r} \right)^{12} - \left( \frac{\sigma_{ij}}{r} \right)^{6} + \frac{1}{4} \right] & r < 2^{1/6}\sigma_{ij} \\
0 & r \geq 2^{1/6}\sigma_{ij}
\end{cases}, \quad (4.1)$$

where $r$ is the distance between the centers of colloids of types $i$ and $j$, $\varepsilon$ sets the strength of the repulsion, and $\sigma_{ij} = (d_i + d_j)/2$. The mixture was supported on a smooth substrate modeled with a repulsive potential,

$$U_i^W(z) = \begin{cases} 
\varepsilon_W \left[ \frac{2}{15} \left( \frac{\sigma_i^W}{z} \right)^9 - \left( \frac{\sigma_i^W}{z} \right)^3 + \frac{\sqrt{10}}{3} \right] & z < (2/5)^{1/6}\sigma_i^W \\
0 & z \geq (2/5)^{1/6}\sigma_i^W
\end{cases}, \quad (4.2)$$

where $z$ is the height of the colloid’s center above the substrate, $\varepsilon_W$ sets the strength of the repulsion and $\sigma_i^W = (d_S + d_i)/2$. 

68
The solvent was modeled implicitly using Langevin dynamics simulations [160–162], which incorporate the effects of Brownian motion and drag from the solvent on the colloids. This simulation method neglects solvent flow and solvent-mediated hydrodynamic forces between colloids. The Langevin dynamics equation of motion for the \( k \)-th colloid is

\[
\dot{r}_k = F_k - \gamma_i \dot{r}_k + \eta_k(t),
\]

(4.3)

where \( r_k \) is the position of the colloid, \( F_k \) is the conservative force acting on the colloid, \( \gamma_i \) is the friction coefficient with the implicit solvent, and \( \eta_k \) is a random force with independent Gaussian-distributed components. Each component \( \alpha \) of the random force has zero-mean and variance

\[
\langle \eta_{k,\alpha}(t) \eta_{k',\alpha'}(t') \rangle = 2\gamma_i k_B T \delta_{k,k'} \delta_{\alpha,\alpha'} \delta(t - t').
\]

(4.4)

A planar solvent interface was modeled using a purely repulsive harmonic potential [51, 163, 164],

\[
U^E(z) = \begin{cases} 
0 & z < H \\
\kappa_i (z - H)^2 / 2 & H \leq z < H^c_i \\
\kappa_i (H^c_i - H)^2 / 2 - F^g_i (z - H^c_i) & z \geq H^c_i
\end{cases},
\]

(4.5)

where \( \kappa_i \) is a spring constant, \( H \) is the height of the interface above the substrate, and \( H^c_i \) is a cutoff height above which it is assumed that a gravitational force \( F^g_i \) dominates. The harmonic form of the potential arises from geometric considerations of the change in surface area (energy) when a colloid is at the interface [163]. This model assumes that both small and big colloids have a contact angle of 90° with the interface and neglects any capillary attractions between colloids or toward the interface from within the mixture. The spring constant \( \kappa_i \) indirectly plays the role of
surface tension and is also proportional to the surface area of the colloid [163], and so the spring constants were scaled accordingly, $\kappa_B/\kappa_S = (d_B/d_S)^2$. We nominally set $H_i^c = H + d_i/2$ so that the gravitational force acted only when the colloid was completely removed from the solvent and $F_{i}^{\sigma} = -\kappa_i(H_i^c - H)$ so that the force on the colloid was continuous at $z = H_i^c$. However, for sufficiently large $\kappa_i$, all colloids remained dispersed below the interface during the drying times considered, and this part of the potential was irrelevant.

Colloid mixtures with big colloid diameters $d_B = 4d_S$, $6d_S$, and $8d_S$ were investigated. The initial height of the mixture along the $z$-axis was $H_0 = 300d_S$. Periodic boundary conditions were applied along the $x$- and $y$-axis with edge length $L = 160d_S$. The composition of the mixture can be defined by a nominal initial volume fraction $\phi_i^0 = \pi d_i^3 N_i / 6V$, where $N_i$ is the number of colloids of component $i$ and $V = L^2H_0$ is the initial volume of the mixture. The volume fractions were fixed at $\phi_S^0 = 0.03$ and $\phi_B^0 = 0.07$, which roughly corresponds to the concentration studied by Fortini et al. for $d_B = 7d_S$ and $N_S/N_B = 151$ [51]. The small particle spring constant at the solvent interface was set to $\kappa_S = 50\varepsilon/d_S^2$, and the substrate interaction strength was set to $\varepsilon_W = 2.5\varepsilon$. All simulations were performed using the HOOMD-blue simulation package [122–124] (version 1.3.3) on general-purpose graphics processing units. The temperature was held constant at $T = 1.0\varepsilon/k_B$, where $k_B$ is Boltzmann’s constant. The simulation timestep was set to $\Delta t = 0.005\tau$, where $\tau = \sqrt{m_Sd_S^2/\varepsilon}$ is the unit of time.

Starting configurations were generated by first randomly dispersing the big colloids into the simulation box without overlaps. The small colloids were then placed randomly into sites of a face-centered cubic lattice surrounding the big colloids. To promote fast diffusion during the equilibration period, the friction coefficients of the colloids were set to $\gamma_S = \gamma_B = 0.1m_S/\tau$. With this choice of friction coefficient, the colloids were only weakly coupled to the implicit solvent, which then acted pri-
marily as a thermostat. Under these conditions, the typical time for a colloid to diffuse across the film was roughly \( H_0^2 \gamma_i/k_B T \approx 9000 \tau \). Accordingly, the initial mixture configuration was relaxed for 200000 \( \tau \) to equilibrate the initial configurations. We confirmed that equilibrium was reached by monitoring the convergence of the colloid density profiles. Subsequently, independent equilibrated configurations were generated for each colloid diameter by saving a configuration every 250 \( \tau \).

Drying simulations were conducted for a constant interface speed \( v \). The position of the interface at time \( t \) was then \( H = H_0 - vt \). The friction coefficient for the small colloids was set to \( \gamma_S = 2.5 m_S/\tau \). This choice gave a Stokes–Einstein diffusion coefficient \( D_S = k_B T/\gamma_S = 0.4 d_S^2/\tau \), which is the same value employed by Fortini et al. \[51\]. The friction coefficients of the big colloids were scaled proportionally to the colloid diameters, \( \gamma_B/\gamma_S = d_B/d_S \), consistent with the drag on a sphere in Stokes flow. Independent drying trajectories were generated from the initial configurations to a final film height \( H = 100 d_S \). (The pseudo-random number generator was also seeded uniquely for each trajectory to ensure statistical independence \[162\].) The number density profiles, \( n_i \), were computed during drying with a bin width 1.0 \( d_S \) for both colloid types. The profiles were subsequently averaged at a given time point over the independent trajectories.

Drying in the high Péclet number regime was studied with the interface speed set to \( v = 0.02 d_S/\tau \), using 200 trajectories for each big colloid diameter. This drying rate gives film Péclet numbers \( \text{Pe}_S = 15 \) for the small colloids and \( \text{Pe}_B = \text{Pe}_S (d_B/d_S) \), which is in the regime \( \text{Pe}_S \gg 1 \) and \( \text{Pe}_B \gg 1 \) studied by Fortini et al. \[51\]. An additional 301 trajectories were simulated for the earliest stages of drying (up to \( t = 2500 \tau \), film height \( H = 250 d_S \)) in order to collect high quality statistics of the colloid velocity profiles, which were computed using the same binning and averaging scheme as for the density profiles.
In order to assess the role of drying rate on stratification, the big colloid diameter was fixed at $d_B = 8d_S$, and the drying rate was systematically reduced. Ten simulations were conducted with the interface speed reduced by a factor of 10 ($v = 0.002d_S/\tau$), and 5 simulations were conducted with the speed reduced by a factor of 100 ($v = 0.0002d_S/\tau$). In the former case, $Pe_S \approx 1$ and $Pe_B \gg 1$, while in the latter case, $Pe_S \ll 1$ and $Pe_B \approx 1$. An interface speed where both $Pe_S \ll 1$ and $Pe_B \ll 1$ was beyond the reasonable drying times accessible computationally.

### 4.2 Results and Discussion

#### 4.2.1 Equilibrium

The equilibrium number density profiles, scaled by the particle diameters, are shown as a function of distance below the interface in Figure 4.2. Here, the interface is shifted to zero, and so the film extends to the left. In all cases, a sizable number of big colloids were located near the film interface. Below the interface, there was a layer with a width comparable to $d_B$ that was depleted of big colloids and filled mostly with small colloids. This effect is due to volume exclusion of the big colloids at the interface. Below this region, the number densities of both components converged to a bulk value. A similar structuring was observed near the substrate; however, these densities are not shown here in detail because stratification occurs near the solvent–air interface. This layering effect is well-known for fluids in contact with surfaces [101, 102], including hard-sphere fluids [103] and hard-sphere mixtures [165, 166].

Classical density functional theory [131–133] (DFT) provides a theoretical framework to predict the observed inhomogeneous density profiles through minimization...
of an appropriate free energy functional. The grand potential,

$$\Omega\left\{n_i(\mathbf{r})\right\} = F\left\{n_i(\mathbf{r})\right\} - \sum_i \mu_i \int \! d\mathbf{r} \, n_i(\mathbf{r}), \quad (4.6)$$

was minimized with respect to the number density profiles. In Eq. 4.6, $F\left\{n_i(\mathbf{r})\right\}$ is the Helmholtz free energy functional of the mixture that includes the substrate and interface external fields, and $\mu_i$ is the chemical potential for particles of type $i$. The White Bear version [167, 168] of Rosenfeld’s fundamental measure theory [169] for hard spheres was employed for $F$, which can reliably predict the structure of binary hard-sphere mixtures at moderate packing fractions near surfaces [133]. Because the WCA pair potential involves soft repulsion, the colloid mixture was mapped onto an effective hard-sphere mixture with diameters $d_i^{\text{HS}}$ by matching second virial
coefficients,
\[\tilde{d}_{i}^{HS} = \left[3 \int_{0}^{\infty} dr \, r^2 (1 - e^{-U_{ii}(r)/k_{B}T})\right]^{1/3}. \quad (4.7)\]

The chemical potentials were chosen to give an average number of particles in the mixture consistent with the simulations. The density profiles were discretized on a mesh of width \(\Delta z = 0.01 d_{S}\), convolutions were taken using fast Fourier transform methods, and the minimization was performed by Piccard iteration with 1\% mixing in most cases and an absolute tolerance of \(10^{-6} d_{S}^{-3}\) in the number density. The DFT-predicted number density profiles are shown as solid lines in Figure 4.2, and are in excellent quantitative agreement with the simulations, indicating two important points: (1) the starting configurations are well-equilibrated, and (2) fundamental measure theory gives a reliable equilibrium free-energy functional for the binary colloid mixture. We will revisit this second point when modeling the drying mixture.

The height of the \(n_{B}\) peak at the interface increased with increasing \(d_{B}\). This behavior corresponds to an increased volume fraction of big colloids at the interface. The local volume fraction, \(\phi_{i}(z)\), can be obtained by convolution of the density profile with the volume of a sphere,

\[\phi_{i}(z) = \int d\mathbf{r}' \, n_{i}(z - \mathbf{r}' \cdot \mathbf{e}_{z}) \Theta(d_{i}/2 - |\mathbf{r}'|), \quad (4.8)\]

where \(\mathbf{e}_{z}\) is the unit vector along the z-axis, \(|\mathbf{r}'|\) is the magnitude of \(\mathbf{r}'\), and \(\Theta\) is the Heaviside step function. The maximum big colloid volume fraction in the film, \(\phi_{B}^{*}\), is shown as an inset to Figure 4.2. This highest concentration region was always located at the solvent–air interface. As the diameter increases, \(\phi_{B}^{*}\) appears to approach close-packing. This layer of big colloids at the interface can also be observed visually in the simulations, as shown in Figure 4.3a for \(d_{B} = 8 d_{S}\).
4.2.2 Effect of size ratio on stratification

Inverted stratification was observed to some extent for all investigated big colloid diameters when \( v = 0.02 \frac{d_S}{\tau} \). For this drying rate, both small and big colloids are expected to accumulate at the interface because \( \text{Pe}_S \gg 1 \) and \( \text{Pe}_B \gg 1 \). Representative snapshots of the drying process are shown in Figure 4.3 for \( d_B = 8d_S \) at the initial, intermediate, and final film heights. Big colloids initially located in the top layer (Figure 4.3a) remained trapped at the surface during drying. At the same time, however, a layer of small colloids formed below the top surface (Figure 4.3b), and grew in size (Figure 4.3c), as indicated approximately by arrows. This finding is in agreement with the observations of Fortini et al. for a similar colloidal mixture and drying rate regime [51].

The time-resolved number density profiles, shown in Figure 4.4 and Figures 4.12 and 4.11 of Appendix 4.A for increasing values of \( d_B \), also clearly exhibit the formation of the stratified layer. For the small colloids, \( n_S \) monotonically increased near the interface and formed a gradient into the film. This accumulation was expected because \( \text{Pe}_S \gg 1 \). The height of the accumulating peak at the interface was roughly
independent of $d_B$. Below the interface, there was a sharp transition for $n_S$ between two regions of high and low density. The position of this transition appeared to correspond with a sudden increase in $n_B$. The region closer to the interface corresponded to the stratified layer of small colloids, while the bottom region was enriched in big colloids. For the big colloids, $n_B$ typically increased near the location of its first peak below the interface, $z^*$. This peak then slowly shifted further away from the interface, corresponding to an increase in the width of the stratified layer. The transition of $n_B$ between the stratified layer of small colloids and the layer enriched in big colloids near $z^*$ was sharper for larger $d_B$, indicating a stronger segregation.

Figure 4.4: Number density profiles during drying for $d_B = 8d_S$ and $v = 0.02d_S/\tau$. The first peak below the interface, $z^*$, is identified with an arrow for $t = 2500\tau$.

In order to quantify the emergence of stratification in the mixture, the position $z^*$ was extracted from the density profiles. The location of this peak below the interface is plotted as a function of time in Figure 4.5. Initially, $z^*$ was given by the equilibrium value from Figure 4.2. Once drying commenced, $z^*$ migrated away from the interface with approximately the same velocity for all big colloid diameters studied. However, it is apparent from visualization and the density profiles that
the height of the stratified layer and the extent of the stratification increases with increasing big colloid diameter, and so $z^*$ does not appear to be an effective proxy for the size of the stratified layer.

Figure 4.5: Distance of the stratification peak $z^*$ below the interface during drying at $v = 0.02 d_S/\tau$.

Accordingly, we also computed the average height migrated by the center-of-mass of each component, $\Delta \langle z_i \rangle$, as another measure of the stratification. The difference in this quantity for the two components is plotted in Figure 4.6. Because the average displacement is in the $-z$ direction during drying, $\Delta \langle z_S \rangle - \Delta \langle z_B \rangle > 0$ indicates that big colloids have travelled farther than small colloids, and suggests the formation of a stratified layer. We observed that, for the model considered here, $\Delta \langle z_S \rangle - \Delta \langle z_B \rangle$ correlated well with the apparent height of the stratified layer in the density profiles, and that the height of the stratified layer grew larger and faster for larger $d_B$. For a given diameter, the height of the stratified layer increased with time, and at a given time, the size of the stratified layer was larger for larger colloid diameters. This finding is in agreement with recent experiments and simulations performed by Martín-Fabiani et al. [52], who also observed an increase in the height of the stratified layer when they increased the size asymmetry from roughly $d_B/d_S = 4$ to $d_B/d_S = 7$. We also note that when we performed additional simulations for $d_B = 2 d_S$ (Figure 4.13 of Appendix 4.A), the height of the stratified layer further decreased to
the point that it became challenging to define the layer, which is consistent with our other findings and the results of Martín-Fabiani et al. [52].

Figure 4.6: Relative center-of-mass displacements during drying at \( v = 0.02 \frac{d_S}{\tau} \).

### 4.2.3 Effect of drying rate on stratification

For a given particle size ratio, reducing the drying rate is expected to decrease the extent of stratification because fewer colloids accumulate at the interface, and the colloids remain distributed closely to their equilibrium profiles. Our simulations and the work of Fortini et al. showed that inverted stratification is exhibited in the regime where \( \text{Pe}_S \gg 1 \) and \( \text{Pe}_B \gg 1 \). In our simulations for \( d_B = 8 \, d_S \) and \( v = 0.02 \frac{d_S}{\tau} \), the film Péclet numbers were \( \text{Pe}_S = 15 \) and \( \text{Pe}_B = 120 \). If the drying rate is reduced by a factor of 10 (\( v = 0.002 \frac{d_S}{\tau} \)), the film Péclet numbers are \( \text{Pe}_S = 1.5 \) and \( \text{Pe}_B = 12 \), and primarily big colloids are expected to accumulate at the interface. Reducing the drying rate further to \( v = 0.0002 \frac{d_S}{\tau} \) reduces the Péclet numbers to \( \text{Pe}_S = 0.15 \) and \( \text{Pe}_B = 1.2 \). For this slowest drying speed, essentially no accumulation of small colloids is expected at the interface, and only a small number of big colloids should accumulate. It is conceivable in these small Péclet number regimes that inverted stratification might not occur anymore because primarily big colloids are expected to accumulate at the interface.
In order to test this hypothesis, we performed additional simulations at the slower drying rates, and computed the number density profiles at the final film height $H = 100 \, d_S$, as shown in Figure 4.7. As the drying rate was reduced, the small colloids approached a more uniform distribution. For $v = 0.0002 \, d_S/\tau$, there was nearly no gradient in $n_S$, which is consistent with the small colloid film Péclet number being reduced from $\text{Pe}_S \gg 1$ to $\text{Pe}_S \ll 1$. Interestingly, inverted stratification was still observed for the big colloids at all drying rates. The extent of this stratification was noticeably lessened as the drying rate decreased and $\text{Pe}_B$ approached unity. For the fastest drying rate, there were essentially no big colloids in the stratified layer. At the slower drying rates, more big colloids were found closer to the interface. However, in all cases, there was still an apparent gradient in $n_B$, with the big colloid density decreased near the interface, suggesting that the mechanism driving inverted stratification at high Péclet numbers is also relevant even when $\text{Pe}_S \lesssim 1$.

![Figure 4.7](image-url)

Figure 4.7: Number density profiles at final film height $H = 100 \, d_S$ for $d_B = 8 \, d_S$ after drying at different interface speeds.
4.2.4 Comparison to theoretical models

As pointed out by Fortini et al., the observed stratification appears to be a purely nonequilibrium separation [51]. We can estimate from the nominal volume fractions of the final state that an equivalent bulk mixture would be a single, dispersed phase using the phase diagram for a binary hard-sphere mixture [170, 171]. We confirmed this estimate for our films by direct simulation. For \( v = 0.02 \, d_S/\tau \), drying was stopped at \( t = 5000 \, \tau \) (intermediate film height \( H = 200 \, d_S \)) and at \( t = 10000 \, \tau \) (final film height \( H = 100 \, d_S \)). The small and big colloids mixed in both cases and adopted density profiles qualitatively similar to those shown in Figure 4.2. Structuring was increased near the interfaces due to the increased confinement. However, no phase separation was observed on the length scales obtained in Figure 4.4.

Although the drying model of Routh and Zimmerman [151] shows reasonable agreement with experiments of a monodisperse suspension [155], the extension to binary mixtures by Trueman et al. [158] was unable to qualitatively capture the inverted stratification observed in their experiments [157] or our simulations. Fortini et al. proposed a model that qualitatively predicts inverted stratification [51]. Briefly reprised, the drying process is modeled using a balance of forces due to the osmotic pressure gradient and the drag force on a particle. At a given height in the film position with pressure gradient \( \partial P/\partial z \), the net force exerted on the particle is approximated as \( F_P \approx d_i^3(\partial P/\partial z) \). The drag force on the sphere was \( F_D = -K_i \gamma_i u_i \), where \( u_i \) is the colloid velocity along the \( z \)-axis, and \( K_i \) is the sedimentation coefficient that depends on the particle volume fractions and is equal to 1 in the dilute limit [1]. Balancing forces gives \( u_i \approx d_i^3(\partial P/\partial z)/K_i \gamma_i \), and so the difference in velocity \( \Delta u = u_B - u_S \) is

\[
\Delta u/u_S \approx \left( \frac{d_B}{d_S} \right)^2 \left( \frac{K_S}{K_B} \right) - 1,
\]

\[(4.9)\]
making use of the fact that $\gamma_B/\gamma_S = d_B/d_S$. The authors argue that at low densities, $K_S/K_B \approx 1$, while at high densities, $K_S/K_B \approx d_S/d_B$ [51]. Their model then predicts that $\Delta u/u_S + 1$ should scale as a power law of $d_B/d_S$ with an exponent between 1 and 2.

In order to test this prediction, the average colloid velocity profiles, $u_i$, were computed during the early stages of drying for $v = 0.02 d_S/\tau$. Representative velocity profiles are shown in Figure 4.8 for $d_B = 8 d_S$ for the initial ($t = 0 \tau$) and final ($t = 2500 \tau$) times for which the velocity profile was measured. Initially, the average velocities were essentially zero everywhere in the film, as expected at equilibrium. During drying, both small and big colloids at the interface had an average velocity consistent with the interface speed $v$, shown as a dotted line in Figure 4.8. Below the interface, the colloids also migrated downward as expected, and the migration speed decayed to zero farther into the film. This behavior at the boundaries was qualitatively expected because there was no flux of colloids through the substrate or drying interface. It is apparent that during drying the velocity of the big colloids was faster than the velocity of the small colloids. This behavior is qualitatively expected because, on average, the big colloids must migrate faster than the small colloids in order for inverted stratification to occur. Note that there is also a region where the big colloids move downward faster than the interface speed. This region was larger and had a faster average downward speed for larger colloid size asymmetries, enhancing the stratification.

The scaling prediction of Eq. 4.9 was tested by computing the difference in migration velocity $\Delta u$ after a given drying time for each particle diameter. For each set of velocity profiles, $\Delta u$ was averaged in a window $z^* \pm 5 d_S$. We confirmed that this choice of the averaging window had a negligible impact on the calculated $\Delta u$ by decreasing the size of the window by a factor of 2. For a given big colloid diameter, the measured migration velocity appeared to approach a steady-state value as the
Figure 4.8: Small and big colloid velocity profiles for \( d_B = 8 \, d_S \). The interface speed \( v = 0.02 \, d_S / \tau \) is shown as a dotted line. The velocity difference \( \Delta u \) is computed at \( z^* \) (Figure 4.5), and is indicated by an arrow.

drying time increased. For completeness, data are reported here for all time points measured.

The migration velocities were scaled according to Eq. 4.9, and are shown plotted on logarithmic axes in Figure 4.9. For a given drying time, the scaled migration velocity increased as the particle size ratio increased, which is qualitatively consistent with the model of Fortini et al. [51]. However, the measured \( \Delta u / u_S + 1 \) appears to scale with an exponent smaller than predicted by the theory. In fact, the scaling appears to be sublinear and to remain roughly constant for all drying times considered here; however, there is insufficient data to numerically quantify this exponent. Hence, although the model qualitatively predicts the correct inversion, it does not appear to quantitatively predict the rate of stratification.

One possible source of error in the models by Trueman et al. [158] and Fortini et al. [51] is that their approximations for the diffusive driving force (fundamentally, the chemical potential gradient) might not be sufficiently accurate. Since DFT accurately predicts the equilibrium density profiles (Figure 4.2), the equilibrium free-energy functional may also provide a reasonable approximation for the chemical potential gradients during out-of-equilibrium drying. We propose a model for stratification in the framework of dynamical density functional theory [172, 173], which
Figure 4.9: Measured migration velocity at $z^*$ scaled according to Eq. 4.9 for varied colloid size ratios and drying times with $v = 0.02 d_S/\tau$. Scaling exponent predicted by Eq. 4.9 in the high-density limit is also indicated for comparison.

can be considered a correction to the form of the chemical potential employed by Trueman et al. [158].

In the typical dynamical density functional theory formulation, the number density flux of component $i$, $\mathbf{j}_i$, is given by [174]

$$
\mathbf{j}_i = -\frac{n_i(r, t)}{\Gamma_i} \nabla \mu_i(r, t),
$$

(4.10)

where $\mu_i(r, t) = \delta F/\delta n_i(r, t)$ is the chemical potential obtained by functional differentiation of $F$, and $\Gamma_i$ is the effective friction coefficient. The number density profiles should then evolve according to the conservation law,

$$
\frac{\partial n_i}{\partial t} = -\nabla \cdot \mathbf{j}_i.
$$

(4.11)

For noninteracting particles, Eq. 4.10 and Eq. 4.11 reduce to the familiar diffusion equation.

The diffusive flux in Eq. 4.10 can be equivalently represented as a net velocity through $\mathbf{j}_i = n_i \mathbf{u}_i$. In one dimension, this gives a difference in migration velocity:

$$
\Delta u / u_S = \left( \frac{\partial \mu_B / \partial z}{\partial \mu_S / \partial z} \right) \left( \frac{\Gamma_S}{\Gamma_B} \right) - 1.
$$

(4.12)
The effective friction coefficient can be approximated by \( \Gamma_i \approx K_i \gamma_i \). The approximate scaling of \( K_i \) proposed by Fortini et al. [51] then predicts \( \Gamma_i \sim d_i \) in the low-density regime and \( \Gamma_i \sim d_i^2 \) in the high-density regime. When \( d_B > d_S \), there is then a competition between the relative magnitudes of the chemical potential gradients and the friction coefficients to determine which component migrates faster. Typically, the chemical potential gradient for the big colloids is larger than for the small colloids because more volume is excluded to the big colloids, and so the driving force for migration is larger. But, the big colloid friction coefficient is also larger, reducing the net migration velocity induced. The extent of inverted stratification should increase with \( d_B \) when \( \partial \mu_B/\partial z \) increases relative to \( \partial \mu_S/\partial z \) as a function of \( d_B \) faster than the size ratio increases.

We computed the chemical potential gradients at the peak position \( z^* \) using the density profiles obtained from the simulations and the equilibrium Helmholtz-free-energy functional, and found that Eq. 4.12 correctly predicted \( \Delta u > 0 \), which is necessary for inverted stratification. The measured migration velocities were then scaled by the computed chemical potential gradients in order to test Eq. 4.12 against the simulations. These data are shown in Figure 4.10 on logarithmic axes. It appears that the scaled migration velocities follow a power law of the size ratio with an exponent of approximately \(-2\), which is in agreement with the scaling proposed by Fortini et al. [51] for the high-density regime, \( \Gamma_S/\Gamma_B \sim (d_B/d_S)^{-2} \). This finding suggests that the proposed dynamical density functional theory may be able to quantitatively predict the evolution of the colloid density profiles, including the inverted stratification. Unfortunately, solution of Eq. 4.10 and Eq. 4.11 for direct comparison with the density profiles obtained in simulation is numerically challenging and is left as the subject of future work.

Zhou et al. [175] also concurrently proposed a similar stratification model based on chemical potential gradients. They employed a simpler form of the free-energy
Figure 4.10: Measured migration velocity at $z^*$ scaled according to Eq. 4.12 for varied colloid size ratios and drying times with $v = 0.02 \frac{d_S}{\tau}$. Scaling exponent predicted by Eq. 4.12 in the high-density limit is also indicated for comparison.

functional using a local-density approximation [132] and the second-order virial expansion for a hard-sphere mixture. This approximation is expected to be reasonable for a dilute mixture with weak variations in density [176]. Zhou et al. showed that their model is able to qualitatively capture the inverted stratification behavior, and their predictions for regimes where stratification can occur are in reasonable agreement with previous experiments and simulations [175]. However, the drying density profiles predicted by their model appear to be missing some qualitative features observed in our simulations, such as the trapping of colloids initially layered near the interface. The reason for these differences is again likely related to the approximate chemical potential gradients derived from the free-energy functional. The authors note that their model can be generalized to a wider range of densities with a more accurate equation of state. The model that we have proposed does precisely that. The White Bear version of fundamental measure theory is based on the Boublík–Mansoori–Carnahan–Starling–Leland equation of state [177, 178], a generalization of the accurate Carnahan–Starling hard-sphere-fluid equation of state [179] to hard-sphere mixtures. Moreover, our proposed model employs a nonlocal free-energy functional that should be more accurate near interfaces where density variations occur. The results of Zhou et al. [175] fully support our proposed model.
4.3 Conclusions

We performed implicit-solvent molecular dynamics of drying binary colloidal mixtures. In agreement with a recent study by Fortini et al. [51], we showed that an inverted stratification occurs at large film Péclet numbers, with a layer of small colloids segregated to the top of the film, and big colloids pushed toward the bottom. The size of this stratified layer depended on the particle size ratio, with larger stratified layers forming for larger size ratios. We also showed that a similar inverted stratification occurs for slower drying rates, but to a lesser extent. The inverted stratification effect appeared to persist even as the big colloid Péclet number approached unity.

We tested the drying model proposed by Fortini et al. [51], and found that the model did not predict the correct scaling of the particle migration velocities as a function of size ratio. We proposed an alternative model based on dynamical density functional theory (DDFT), which showed reasonable scaling agreement with the measured migration velocities. Qualitatively, inverted stratification occurs in this model because the chemical potential gradients driving diffusion of the big colloids are larger than for the small particles, and the magnitude of this driving force increases with particle diameter faster than the mobility decreases with diameter, resulting in a net faster migration velocity for larger particles.

In the chapter, we have neglected the influence of solvent-mediated hydrodynamics. Although experiments by Fortini et al. suggested that inverted stratification still occurs even in the presence of solvent [51], an investigation of the extent to which solvent flow and hydrodynamic forces influence stratification is warranted, and is discussed in Chapter 6. We have also neglected any hydrodynamic- or local-density-dependent contributions to the friction coefficients in the DDFT, the inclusion of which is left as a subject of future work.
4.A Additional Density Profiles

Number density profiles during drying for $d_B = 2d_S$, $4d_S$, and $6d_S$ at $v = 0.02d_S/\tau$.

Figure 4.11: Number density profiles during drying for $d_B = 6d_S$ and $v = 0.02d_S/\tau$.

Figure 4.12: Number density profiles during drying for $d_B = 4d_S$ and $v = 0.02d_S/\tau$. 
Figure 4.13: Number density profiles during drying for $d_B = 2d_S$ and $v = 0.02d_S/\tau$. 
Chapter 5

Stratification in drying polymer and colloid–polymer mixtures

Most previous work [51, 52, 175, 180] and Chapter 4 focused on stratification for hard spherical colloids, but only relatively little is known about how polymers and other soft materials stratify during drying. Stratification has been proposed as a key step in the mechanism for phase separation [181, 182] of spin-coated thin-films of immiscible polymer blends [183]. In polydisperse mixtures, stratification may modify the distribution of polymers at the drying interface, affecting the drying dynamics [184] and film structure [185, 186] through the polymer skin-layer [187–189]. Processing conditions can also be exploited to control the dispersion of nanoparticles in polymer nanocomposites [5, 190]. Cheng and Grest [54] showed that polymers can form a protective layer that prevents aggregation of colloids at the drying interface and increases colloid dispersion toward the substrate. However, their study was restricted to a regime in which the nanoparticles were larger than the polymers. In

---

short, there are many potential applications and implications of stratification for polymers in drying films, but additional understanding is first required.

In this chapter, we investigate stratification during drying of both polymer–polymer and colloid–polymer mixtures. We propose a predictive theoretical model for stratification in these mixtures, and perform computer simulations to validate our model. We demonstrate excellent agreement between the model predictions and the simulations for the polymer–polymer mixtures, which undergo an inverted stratification with the shorter polymer chains found on top. We show that colloid–polymer mixtures exhibit either colloid-on-top or polymer-on-top stratification, and that this stratification can be predicted theoretically based on the relative sizes of the polymers and colloids. Our results have implications for the engineering of multilayer coatings, polymer nanocomposites, and organic electronics.

The rest of the chapter is organized as follows. The physical problem is motivated and described in Section 5.1, and the corresponding simulation model and methods are presented. The theoretical model for the drying mixtures is developed in Section 5.2. The model predictions and simulation results are then compared and discussed in Section 5.3.

5.1 Simulation Model and Methods

Linear polymer chains in solution were modeled as $M$ beads connected by springs. Monomers of diameter $d$ and mass $m$ interacted with each other through the purely-repulsive Weeks–Chandler–Andersen potential [83],

$$U_{nb}(r) = \begin{cases} 
4\varepsilon \left[ \left( \frac{\sigma}{r} \right)^{12} - \left( \frac{\sigma}{r} \right)^{6} + \frac{1}{4} \right] & r < 2^{1/6}\sigma \\
0 & r \geq 2^{1/6}\sigma
\end{cases},$$

(5.1)
where $r$ is the distance between the centers of two monomers, $\varepsilon$ sets the strength of the repulsion, and $\sigma = d$. Bonds between monomers were modeled using the finitely extensible nonlinear elastic potential \[84],

$$U_b(r) = \begin{cases} 
-\frac{\kappa r_0^2}{2} \ln \left[1 - \left(\frac{r}{r_0}\right)^2\right] & r < r_0 \\
\infty & r \geq r_0
\end{cases}, \quad (5.2)$$

with the standard Kremer–Grest parameters $\kappa = 30 \varepsilon/d^2$ and $r_0 = 1.5 d$ \[85\. These parameters give an equilibrium bond length $b \approx 0.97 d$ at temperature $T = \varepsilon/k_B$ ($k_B$ is Boltzmann’s constant), which prevents any unphysical chain-crossing. In a dilute solution, this polymer model approximately corresponds to good-solvent conditions, for which the polymer radius of gyration $R_g$ is given by $R_g \sim bM^\nu$ with $\nu \approx 0.588$ in the large $M$ limit \[94\].

As in Chapter 4, interactions between colloids of diameter $d_c$ and mass $m_c$ were also modeled using Eq. 5.1 with $\sigma = d_c$ for colloid–colloid interactions and $\sigma = (d_c + d)/2$ for colloid–monomer interactions. The colloid mass was scaled relative to the monomer mass by volume, $m_c = (d_c/d)^3 m$. The systems were bounded below by a smooth substrate that was modeled with a purely repulsive potential. Because explicitly resolving the solvent would be highly computationally intensive, the solvent was instead represented implicitly using Langevin dynamics simulations [160–162]. This simulation method includes the effects of Brownian motion and solvent drag but neglects other hydrodynamic interactions. The solvent–air interface was represented using the purely repulsive part of a harmonic potential. Gravity did not act on any of the components of the mixture. Additional details of the model parameters (using the monomer diameter as the unit of length) can be found in Chapter 4.

For a single polymer chain, the Langevin equations of motion give rise to Rouse-like scaling \[191\] for the center-of-mass diffusion coefficient, $D = k_B T/M\gamma$, where
\( \gamma \) is the friction coefficient imposed on a monomer in the simulation. However, hydrodynamic interactions for polymer chains in dilute solution should instead result in Zimm-like scaling \([93, 94, 192, 193]\),

\[
D = \frac{k_B T}{M \gamma} + \frac{k_B T}{6 \pi \eta} \left\langle \frac{1}{R_h} \right\rangle,
\]

where \( \eta \) is the solution viscosity and \( R_h \) is the hydrodynamic radius of the polymer,

\[
\left\langle \frac{1}{R_h} \right\rangle = \frac{1}{M^2} \sum_{i \neq j}^M \left\langle \frac{1}{r_{ij}} \right\rangle,
\]

with \( r_{ij} \) being the distance between monomer \( i \) and monomer \( j \) in the polymer chain. The hydrodynamic radius is proportional to the radius of gyration \( (R_h \sim R_g \sim bM^{\nu}) \) \([194]\), and so from Eq. 5.3, the diffusion coefficient scales as \( D \sim k_B T/\eta bM^{\nu} \) in the large \( M \) limit. This Zimm-like scaling has a weaker \( M \) dependence \( (D \sim 1/M^{\nu}) \) than Rouse-like scaling \( (D \sim 1/M) \).

In order to approximately obtain Zimm-like scaling for \( D \) in our Langevin dynamics simulations, we adjusted \( \gamma \) to depend on \( M \):

\[
\gamma = \gamma_0 \left( 1 + M \left\langle \frac{d}{2R_h} \right\rangle \right)^{-1},
\]

where we used \( \gamma_0 = 3\pi \eta d \) for the low-Reynolds-number motion of a spherical monomer. The hydrodynamic radius was calculated for a single chain of length \( M \) at infinite dilution (Figure 5.1) in order to determine \( \gamma \). The polymer dynamics obtained by this procedure serve as an approximation of the true dynamics for two reasons: (1) the internal polymer dynamics in our Langevin simulations are still Rouse-like rather than Zimm-like \([85]\), and (2) the true polymer dynamics should cross over to Rouse-like dynamics at sufficiently high concentrations \([94]\). The first approximation is well-justified because the internal relaxation times of the polymers
in our model are much shorter than the drying time. The second approximation is appropriate provided that the polymer solution is sufficiently dilute. Accordingly, we conducted all simulations at concentrations below the estimated overlap concentration. Both of these points are discussed in more detail below.

![Figure 5.1: Radius of gyration (left axis) and hydrodynamic radius (right axis) for a single polymer chain of length $M$ at infinite dilution.](image)

All simulations were performed using the HOOMD-blue simulation package (version 2.1.5) on general-purpose graphics processing units \([122–124]\). The temperature was fixed at $T = \varepsilon/k_B$, and the simulation timestep was set to $0.005\tau$, where $\tau = \sqrt{md/\varepsilon}$ is the derived unit of time. The simulation box was periodic along the $x$- and $y$-axis with edge-length $L = 150d$. The initial interface height was $H_0 = 300d$.

We considered three types of mixtures: (1) a mixture of a polymer with its own monomers, (2) a mixture of two polymers, and (3) a mixture of colloids and polymers. The polymer–monomer and polymer–polymer mixtures consisted of 800000 total monomers, with the monomers divided equally between the two species. The colloid–polymer mixtures were prepared from 400000 monomers and 4000 colloids with diameter $d_c = 6d$. We considered polymers of length $M = 10$, 25, 40, and 80, giving between 40000 and 5000 polymer chains of a given length and spanning a factor of four in $R_g$ (Figure 5.1). For the colloid–polymer mixtures, the size ratio of the colloid to the polymer, $d_c/2R_g$, spans a factor of two with $d_c \approx 2R_g$ for $M = 25$. Starting configurations were generated from initially randomized states.
by equilibrating for 25000 $\tau$ with the friction coefficients set to $\gamma_c = \gamma = 0.1 \, m/\tau$ in order to promote fast diffusion. Initial equilibration was confirmed by monitoring the density profiles computed with bin width $1.0 \, d$.

Drying was simulated by moving the solvent–air interface downward with a constant speed $v$ to a final height $H = 100 \, d$. The friction coefficient of a free monomer was set to $\gamma_0 = 2.5 \, m/\tau$, which fixes the viscosity $\eta$ and the effective friction coefficient $\gamma$ of a polymer chain. The colloid friction coefficient $\gamma_c$ was scaled relative to $\gamma_0$ by the colloid diameter, $\gamma_c = (d_c/d)\gamma_0$, according to the Stokes–Einstein relationship. For this choice of parameters, the estimated Zimm relaxation time, $\tau_Z$, for the longest polymer chain ($M = 80$) was $\tau_Z \sim \eta R_g^3/k_B T \approx 71 \, \tau$. The solvent–air interface typically moved with a speed $v = 2 \times 10^{-2} \, d/\tau$, and so the time required for the interface to dry a distance $R_g$ was $324 \, \tau > \tau_Z$. The estimated overlap concentration of the longest polymer chain was $c^* \sim M/R_g^3 \approx 0.3/d^3$, and so the monomer concentrations $c$ were typically in or close to the dilute regime, $c/c^* \lesssim 1$, throughout the simulations for all chain lengths. Hence, our approximation of Zimm-like scaling for the polymer center-of-mass diffusion coefficient is reasonable.

5.2 Theoretical Model

As in Chapter 4, the drying mixture can be modeled theoretically by a diffusion equation within the framework of dynamical density functional theory (DDFT) [172–174] using Eq. 4.10 and Eq. 4.11. Under isothermal conditions, the diffusive flux $j_i$ is proportional to the chemical potential gradient [174],

$$j_i = -D_i n_i \frac{\partial (\beta \mu_i)}{\partial z},$$

where $D_i$ is the Stokes–Einstein diffusion coefficient and $\beta = 1/k_B T$. In DDFT, the chemical potential, $\mu_i = \delta F/\delta n_i({\bf r},t)$, is obtained by differentiation of a free-
energy functional $F$. (Fluxes induced by gradients of other species are neglected.) In Chapter 4, we approximated $F$ by the White Bear version [167, 168] of Rosenfeld’s fundamental measure theory [169], which gave excellent predictions for the mixture densities at equilibrium. However, obtaining the time evolution of the densities proved challenging due to nonlocal contributions to the free energy. Zhou et al. [175] modeled the same colloid mixture using a local-density approximation [132], $F[\{n_i\}] = \int \text{d}r f(\{n_i(r,t)\})$, where $f$ is the free-energy density. They approximated $f$ by a virial expansion for a bulk hard-sphere mixture, which is valid in the low-density limit. This approach proved more amenable to analysis and numerical solution; however, it is expected to incur errors at moderate densities or for large density variations.

In the chapter, we also adopt a local-density approximation for the free-energy functional. We will discuss the implications of this approximation later when comparing the model predictions to the simulations. The free-energy density $f$ is obtained from the equation of state for a hard-chain mixture derived by Chapman, Jackson, and Gubbins [195, 196] using Wertheim’s [197–200] first-order thermodynamic perturbation theory (TPT1):

$$
\beta f = \sum_i n_i \left[ (\ln n_i \Lambda_i - 1) + M_i \frac{\beta f_{\text{hs}}}{n_m} - (M_i - 1) \ln g(d_i^+) \right].
$$

(5.7)

Each species $i$ is considered to be a linear chain consisting of $M_i$ hard-spheres of equal diameter $d_i$. (Colloids and free monomers are chains of length $M_i = 1$.) The first term is the free-energy density for an ideal solution of chains with thermal wavelengths $\Lambda_i$. The second term incorporates volume exclusion between colloids and monomers with $f_{\text{hs}}$ being the excess free-energy density of a reference hard-sphere mixture with total density $n_m = \sum n_i M_i$. The last term accounts for chain formation within TPT1, where $g(d_i^+)$ is the value of the reference hard-sphere-mixture radial
distribution function at contact. We model this hard-sphere mixture using Boublík’s equation of state [177, 178, 201] (see Section 5.A), which reduces to the accurate Carnahan–Starling equation of state [179] for the hard-sphere fluid in the case of equal diameters.

The chemical potential obtained by differentiation of the free energy has corresponding ideal and excess contributions, \( \mu_i = \mu_i^{id} + \mu_i^{ex} \). The first term is the ideal-solution chemical potential, \( \beta \mu_i^{id} = \ln n_i \Lambda_i^3 \). The excess chemical potential, \( \mu_i^{ex} \), contains contributions from both volume exclusion and chain connectivity. A general expression can be derived for \( \mu_i^{ex} \) for the colloid–polymer mixture (see Section 5.A). Qualitatively, \( \mu_i^{ex} \) is proportional to chain length \( M_i \) and is larger for beads with larger diameters. For the polymer–polymer mixtures where all monomer beads have equal diameter, a particularly simple form for \( \mu_i^{ex} \) is obtained:

\[
\frac{\beta \mu_i^{ex}}{M_i} = \frac{8\eta - 9\eta^2 + 3\eta^3}{(1 - \eta)^3} \left( \eta - \frac{n_p \pi d^3}{6} \right) \frac{5 - 2\eta}{(2 \eta)(1 - \eta)} \frac{M_i - 1}{M_i} \ln \frac{1 - \eta/2}{(1 - \eta)^3},
\]

(5.8)

where \( \eta = \frac{n_m \pi d^3}{6} \) is the monomer packing fraction and \( n_p = \sum n_i \) is the total number of polymers. The flux for a polymer chain is then

\[
j_i = -D_i \left( \frac{\partial n_i}{\partial z} + n_i M_i \frac{\partial h}{\partial z} \right),
\]

(5.9)

where \( h(\eta, n_p) = \frac{\beta \mu_i^{ex}}{M_i} \) is a function that is essentially independent of \( M_i \) for sufficiently long chains. The first term in Eq. 5.9 is the familiar diffusive flux and is obtained from the gradient of the ideal-solution chemical potential. The second term captures all nonideal contributions from the excess chemical potential and is necessary for stratification to occur [175]. The flux generates an effective migration velocity \( u_i = j_i/n_i \) for a polymer chain. The derivative \( \partial h/\partial z \) is also independent of chain length, and so neglecting the ideal contribution, it is then clear that \( u_i \sim D_i M_i \).
For polymers obeying Zimm-like dynamics, \( D_i \sim 1/M_i^{1-\nu} \) and hence \( u_i \sim M_i^{1-\nu} \). Since \( \nu \approx 0.588 \) for good-solvent conditions, we then expect that \( u_i \) should increase as \( M_i \) increases, inducing an inverted stratification with shorter polymers on top of longer polymers.

5.3 Results and Discussion

5.3.1 Polymer–polymer mixtures

We first considered the drying of polymer–monomer and polymer–polymer mixtures. Initially, the mixtures were well-dispersed with an almost uniform density distribution through the film. Near the interface, there was a small depletion of large polymers on a length scale of roughly \( R_g \) due to the entropic penalty of placing a polymer chain near an impenetrable surface. Such an equilibrium density profile is consistent with previous simulations [202] and classical density functional theory predictions [203]. The mixture drying regime can be characterized by a dimensionless Péclet number, \( \text{Pe}_i = H_0 v / D_i \). When \( \text{Pe}_i \ll 1 \), diffusive motion dominates over advection, and a uniform density distribution is expected in the film; when \( \text{Pe}_i \gg 1 \), accumulation toward the drying interface is instead expected. The monomer Péclet number was \( \text{Pe}_m = 15 \) at the drying rate \( v = 2 \times 10^{-2} d/\tau \). Since \( D_i \) decreases with chain length, the polymer chains had larger \( \text{Pe}_i \) than the monomers, and so both the monomers and polymers were expected to accumulate at the drying interface and form a density gradient.

Our theoretical model then predicted that inverted stratification should occur for the polymer–monomer mixtures with a layer of monomers found on top of the polymers. Figure 5.2 shows a snapshot for the mixture with \( M = 80 \) taken at the end of the simulation (\( H = 100 d \)). A layer of monomers can clearly be seen on top of the polymers (Figure 5.2a), which have been pushed down (Figure 5.2b). This behavior
is analogous to drying colloid mixtures (Chapter 4), where the larger colloids were pushed below a top layer of small colloids. We confirmed that the effect was purely nonequilibrium by stopping the drying and reestablishing nearly-uniform equilibrium density profiles in the film.

Our theory predicts that the inverted stratification should be more pronounced for longer polymer chains because the migration velocity increases with chain length, assuming comparable density gradients in the different polymer–monomer mixtures. In order to compare the height of the stratified layer for the different mixtures, we computed the monomer density profiles, $n_{i,m}$, for both the polymers and monomers. Figure 5.3 shows the final density profiles obtained for the different mixtures. Consistent with Figure 5.2, the monomer density near the solvent–air interface was significantly higher than the polymer density, with nearly-zero polymer density for the longest chains.

The density profiles obtained by numerically solving our theoretical model for the polymer–monomer mixture (see Section 5.B) are in excellent agreement with the simulation results (Figure 5.3). We emphasize that there are no free parameters in our theoretical model: the results were obtained by inputting only the equilibrium diffusion coefficient of a polymer chain at infinite dilution. The density profiles
Figure 5.3: Final monomer density profiles for polymers of length $M$ (solid lines) in a 50 wt% mixture with their monomers (dashed lines) dried at $v = 2 \times 10^{-2} d/\tau$.

predicted by the theory exhibited the same trends as the simulations, with the largest stratified layer formed for $M = 80$. The transition between the monomer-enriched top layer and the polymer-enriched bottom layer sharpened as the polymer chain length increased in both the simulations and the model, which is similar to what we observed in colloid mixtures of increasing size ratio. The model also captured the sudden change in slope of the monomer density near $z \approx 80 d$ for the $M = 80$ mixture. The theoretical model had almost perfect quantitative agreement for the maximum monomer density at the drying interface, but gave a slight overprediction of the height of the stratified layer and, as a consequence, the polymer density in the bottom of the film. One likely cause of this discrepancy is that diffusion coefficients at infinite dilution are too large compared to their actual values at finite concentration, and so the migration speeds were overpredicted. Incorporating the concentration dependence of the diffusion coefficient would likely improve this prediction. We leave such additional refinements as a subject of future work.
Polymer–polymer mixtures are also expected to stratify, using the same theoretical reasoning as for the polymer–monomer mixtures. The height of the stratified layer should be smaller when the polymer chain lengths are more similar because the difference in migration velocities is smaller. The most stratification is expected for the polymer–monomer mixture \((M = 1)\), and no stratification should occur when the chain lengths were equal. Figure 5.4 shows the final monomer density profiles for polymer–polymer mixtures with one polymer length fixed at \(M = 80\). As expected, inverted stratification occurred in all cases, but to a lesser extent for smaller chain length differences. The theoretical model again gave good quantitative predictions for the density profiles.

![Figure 5.4: Final monomer density profiles for 80-bead polymers (solid lines) in a 50 wt% mixture with shorter polymers of length \(M\) (dashed lines) dried at \(v = 2 \times 10^{-2} d/\tau\).](image)

We expected from our theoretical model and Chapter 4 that reducing the drying speed should reduce the height of the stratified layer because the magnitude of the density gradients is reduced. To test this hypothesis, we conducted additional drying simulations for the polymer–monomer mixture with \(M = 80\) at slower drying speeds.
$v = 2 \times 10^{-3} \, d / \tau$ and $v = 2 \times 10^{-4} \, d / \tau$. These drying speeds corresponded to $\text{Pe}_m \approx 1$ and $\text{Pe}_m \ll 1$ for the monomers, respectively, and $\text{Pe}_{80} \gg 1$ and $\text{Pe}_{80} \approx 1$ for the polymers. Minimal stratification was then expected for the slowest drying rate. Figure 5.5 shows the monomer density profiles at the reduced drying speeds. As expected, the height of the stratified layer decreased with $v$, and almost completely disappeared for the slowest drying rate. The sharpness of the stratification also decreased with decreasing drying rate, which is indeed consistent with our model and the results of Chapter 4.

![Figure 5.5: Final monomer density profiles for polymers of length $M = 80$ (solid lines) in a 50 wt% mixture with their monomers (dashed lines) dried at different speeds $v$.](image)

Inverted stratification is predicted by our theoretical model for polymer mixtures because the driving force on the polymer increases linearly with $M$, while the mobility decreases like $1/M^\nu$ for Zimm-like diffusion. An immediate consequence of our theory is then that inverted stratification should be significantly less pronounced, or nonexistent, if the polymers instead exhibit Rouse-like diffusion, $D \sim 1/M$. We tested this prediction for the polymer–monomer mixtures by artificially setting all friction coefficients to $\gamma = \gamma_0$, which lead to Rouse-like scaling of the polymer diffusion coefficients. We found that all investigated mixtures developed similar density profiles with more polymers than monomers accumulated at the drying interface, i.e., no inverted stratification. This result highlights the important interplay between the
chemical potential driving force and the mobility in determining the stratification behavior.

5.3.2 Colloid–polymer mixture

We subsequently considered a mixture of colloids with diameter $d_c = 6d$ and polymers of different chain lengths. The polymer lengths were chosen to span a range of size ratios with the colloid. Making simple scaling predictions based on our theoretical model for the colloid–polymer mixture is challenging (see Section 5.A); however, we can make a short qualitative analysis. At a given polymer and colloid density, the chemical potential is proportional to the chain length and is larger for larger particle diameters. The exact scaling of the chemical potential with the particle diameter is nontrivial due to the lack of a leading-order term in the diameter, but it is approximately a polynomial in $d_i$ for a given set of volume fractions. Supposing that the scaling can be approximated as a power law of the diameter, $d_i^k$, the migration velocity is expected to roughly scale as $u_i \sim M_i^{1-\nu} d_i^{k-1}$ because the diffusion coefficients follow $D_i \sim 1/M_i^{\nu} d_i$ using Zimm-like scaling for the polymers or the Stokes–Einstein relation for the colloids. Provided that $k > 1$, which is reasonable for Eq. 5.11 and Eq. 5.12 in Section 5.A, either the polymer or the colloid will have a larger migration speed depending on the polymer chain length and the colloid diameter. When $M = 1$, the model reduces to the colloid mixture we and others [51, 175] previously analyzed, where the smaller colloids were always found on top. Hence, we expect that short polymers ($2R_g \ll d_c$) should also stratify in a polymer-on-top configuration. In the opposite limit where the polymers are large compared to the colloids ($2R_g \gg d_c$), the colloid-on-top configuration is instead predicted. There is an intermediate value of $M$, when the polymer is of comparable size to the colloid, that transitions between these two regimes.
We conducted drying simulations for colloid–polymer mixtures in the high Péclet number regime. Figure 5.6 shows snapshots of the final dried states for polymers of length $M = 10$, $M = 25$, and $M = 80$. These polymer sizes correspond to $2R_g < d_c$, $2R_g \approx d_c$, and $2R_g > d_c$, respectively. For $M = 10$, polymer-on-top stratification occurred, and most of the colloids were pushed below the polymers, with the exception of some colloids that were trapped at the solvent–air interface before drying. This size ratio between the polymers and colloids is in a comparable regime to that studied by Cheng and Grest ($M = 100$ and $d_c = 20d$) [54], who also observed a similar layer of polymers on top of a drying film when the polymer–colloid interactions were weak. Polymer-on-top stratification also occurred for $M = 25$, but the separation was weaker. For $M = 80$, colloids instead aggregated at the drying interface, undergoing a colloid-on-top stratification. Hence, the polymers appear to enhance the dispersion of colloids to the substrate only when $2R_g \lesssim d_c$.

![Figure 5.6: Snapshots of stratified colloid–polymer mixture with $M = 10$, 25, or 80 at the end of drying ($H = 100d$) at $v = 2 \times 10^{-2} d/\tau$. The top row shows the entire mixture of colloids (yellow) and polymers (blue), while the bottom row shows only the colloids.](image)

The transition from polymer-on-top to colloid-on-top behavior is most readily seen in the colloid density profiles, shown in Figure 5.7 for the average of 40 independent drying simulations. As expected from Figure 5.6, polymer-on-top stratification occurred for $M = 10$ and $M = 25$. The height of the stratified layer was larger and the interface between this layer and the colloid-enriched layer was sharper for
the shorter polymers. Colloid-on-top stratification occurred for both $M = 40$ and $M = 80$, and was more pronounced for the longer polymers. In both cases, a secondary peak formed at a distance of roughly $d_c$ below the drying interface due to the accumulation of colloids at the surface. However, there was also a region below this peak that was depleted of some colloids, suggesting that some colloids were still pushed down.

![Figure 5.7: Final colloid density profiles for mixtures with polymers of length $M$ dried at $v = 2 \times 10^{-2} d/\tau$.](image)

We attempted to numerically solve our theoretical model for the colloid–polymer mixture and compare the final density profiles with the ones obtained from our simulations. However, our efforts were stymied by numerical instabilities at long times due to the emergence of large density gradients. We were able to resolve the early stages of drying up to time $1000 \tau$ ($H = 280 d$) and observed qualitatively similar trends to Figure 5.7. Polymer-on-top configurations were obtained for $M = 10$ and $M = 25$, and colloid-on-top configurations had already begun to form for $M = 40$ and $M = 80$. The local-density approximation for the free-energy functional was unable to resolve the multiple colloid peaks near the interface. This model could be improved by adopting a more accurate nonlocal functional [203].
5.4 Conclusions

We developed a theoretical model for stratification in polymer–polymer and colloid–polymer mixtures, and validated it using implicit-solvent Langevin dynamics simulations. Polymer–polymer mixtures exhibited an inverted stratification analogous to colloid mixtures, with a layer of small polymers accumulated on top and larger polymers pushed down. Colloid–polymer mixtures showed polymer-on-top stratification when the polymers were smaller than the colloids \(2R_g \lesssim d_c\) and colloid-on-top stratification otherwise. Our model, based on dynamical density functional, predicts the nonequilibrium stratification behavior using only the equilibrium equation of state and diffusion coefficients. The model gives excellent quantitative agreement with simulations of the polymer–polymer mixtures, and qualitatively predicts both stratified structures in the colloid–polymer mixtures. Stratification of polymer–polymer mixtures and colloid–polymer mixtures has important implications for the preparation of materials such as multilayer polymer coatings or polymer nanocomposites by drying.

The analysis and simulations we performed focused on the initial stages of drying from the dilute solution regime where the polymers have Zimm-like diffusion. During the late stages of drying when the polymer concentration increases, the polymers will cross over to Rouse-like diffusion and, according to our theory and simulations, the stratified layer should cease or slow growing. If stratification is the desired effect, one could slow the polymer dynamics in order to lock in the stratified structure and prevent remixing on the timescales of additional drying, for example by increasing the viscosity of the solvent or lowering the temperature. Our theoretical model can be extended to these higher concentrations by incorporating a concentration-dependent diffusivity and a nonlocal free-energy functional. Experiments or simulations using
methods capable of resolving the late stages of drying will be necessary in order to study this aspect of the process in more detail.

We have so far neglected enthalpic interactions between components of the mixture. The inclusion of such interactions may alter the stratification behavior, as is true for colloid mixtures [159]. These enthalpic interactions could be incorporated into our theoretical model by appropriate modification of the free-energy functional. We have also neglected hydrodynamic interactions mediated by the solvent in our simulations or analysis. We note that Cheng and Grest obtained comparable polymer-on-top stratification for an explicit-solvent simulation of a colloid–polymer mixture [54]. The influence of such hydrodynamic interactions on stratification is discussed in the next chapter.

5.A Colloid–Polymer Mixture Model

For colloid–polymer mixtures where the colloids and monomers may have different diameters, we employ Boublík’s equation of state for a hard-sphere mixture [177, 178]. The excess free-energy density of the mixture is [201]:

\[
\beta f^{hs} = \frac{6}{\pi} \left[ \left( \frac{\xi_3}{\xi_2} - \xi_0 \right) \ln(1 - \xi_3) + \frac{3\xi_1 \xi_2}{1 - \xi_3} + \frac{\xi_3^3}{\xi_3(1 - \xi_3)^2} \right],
\]

(5.10)

where \( \xi_m = \sum_i n_i M_i \pi d_i^m / 6 \). (Note that \( \xi_3 \) gives the total volume fraction.) The excess chemical potential has two contributions, \( \mu_i^{ex} = \mu_i^{hs} + \mu_i^p \). The hard-sphere contribution \( \mu_i^{hs} \) is:

\[
\beta \mu_i^{hs}/M_i = -\ln(1 - \xi_3) + \frac{3\xi_2}{1 - \xi_3} d_i + \left( \frac{3\xi_1}{1 - \xi_3} + \frac{3\xi_2^2}{\xi_3^2} \ln(1 - \xi_3) + \frac{3\xi_3^2}{(1 - \xi_3)^2 \xi_3} \right) d_i^2
\]

\[
+ \left[ \frac{3\xi_1 \xi_2}{(1 - \xi_3)^2} - \left( \frac{\xi_3^3}{\xi_3^2} - \xi_0 \right) \frac{1}{1 - \xi_3} - \frac{2\xi_2^3}{\xi_3^3} \ln(1 - \xi_3) - \frac{\xi_3^3}{\xi_3(1 - \xi_3)^2} + \frac{2\xi_2^3}{\xi_3(1 - \xi_3)^3} \right] d_i^3,
\]

(5.11)

106
and the chain contribution $\mu_i^p$ is:

$$\frac{\beta \mu_i^p}{M_i} = -\frac{M_i - 1}{M_i} \ln \left( \frac{1}{1 - \xi_3} + \frac{3\xi_2 d_i}{2(1 - \xi_3)^2} + \frac{\xi_2^2 d_i^2}{2(1 - \xi_3)^3} \right)$$

$$- \frac{d_i^3 \pi}{6} \sum_j n_j(M_j - 1) \left( \frac{3}{1 - \xi_3} + \frac{d_j/d_i - 2}{2 + d_j \xi_2 - 2 \xi_3} + \frac{d_j/d_i - 1}{1 + d_j \xi_2 - \xi_3} \right).$$

(5.12)

If all diameters are equal, these expressions can be simplified to obtain Eq. 5.8 in Section 5.2.

5.B Numerical Solution of Model

We solved the diffusion equation numerically using the Matlab (R2016b) routine `pdepe` in a reduced set of variables. Time was reduced by the drying time for the entire film, $\tilde{t} = vt/H_0$. The spatial coordinate $z$ was reduced by the time-dependent film height, $\tilde{z} = z/H_0(1 - \tilde{t})$, which converts the shrinking domain $0 \leq z \leq H(t)$ to a fixed domain $0 \leq \tilde{z} \leq 1$. Care must be taken when transforming the time derivative into this set of coordinates. For example, for the polymer–polymer mixtures, the transformed conservation law is:

$$\frac{\partial n_i}{\partial \tilde{t}} + \frac{\tilde{z}}{1 - \tilde{t}} \frac{\partial n_i}{\partial \tilde{z}} = \frac{1}{Pe_i(1 - \tilde{t})^2} \frac{\partial}{\partial \tilde{z}} \left( \frac{\partial n_i}{\partial \tilde{z}} + n_i M_i \frac{\partial h}{\partial \tilde{z}} \right).$$

(5.13)

The transformed boundary conditions enforce no flux at $\tilde{z} = 0$ and $\tilde{z} = 1$. We assumed that the diffusion coefficients were constant and given by their values at infinite dilution. In doing so, we neglected steric and hydrodynamic interactions at finite concentration. The hard-sphere diameters in the free-energy functional were taken to be equal to the monomer and colloid diameters because the effective hard-sphere diameters calculated for the WCA potential (Eq. 5.1) at $T = \varepsilon/k_B$ using the
mapping we employed in Chapter 4 differed by less than 2%. The density profiles were initially uniform at values consistent with the number of particles in the simulations. We solved the coupled differential equations using 100 mesh points, which we found to give acceptable numerical accuracy. The polymer density profiles were converted to monomer density profiles using $n_{i,m} \approx n_i M_i$. 

\[ n_{i,m} \approx n_i M_i. \]
Chapter 6

Influence of hydrodynamic interactions on stratification

For computational and theoretical convenience, many previous studies, including Chapters 4 and 5, have employed a free-draining approximation for hydrodynamic interactions to model stratification. This approach incorporates the Stokes drag on each particle but neglects other interactions. The computer simulations [51–53, 180] implicitly modeled the solvent using Langevin or Brownian dynamics methods that treat the solvent as a quiescent, viscous background, while the theoretical models [51, 175, 180] assumed that the particle mobilities were given by the Einstein relation. However, Sear and Warren point out in a recent analysis [204] that neglecting hydrodynamic interactions overpredicts the phoretic velocity of a single large colloid in an ideal polymer solution, which must be proportional to the extent of stratification in this simple model. They argue that this discrepancy is caused by the omission of solvent backflow [204], leading to an incorrect hydrodynamic mobility [205]. Their argument is qualitatively supported by differences between the exper-

iments and implicit-solvent simulations of Fortini et al. [51]. To our knowledge, though, there has been no direct test of how hydrodynamic interactions influence stratification at finite concentrations or size ratios between components, which are more challenging to analyze theoretically [204, 205].

In this chapter, we systematically demonstrate the influence of hydrodynamic interactions on stratification for a similar drying polymer mixture as in Chapter 5. Computer simulations are ideal tools for such a study because, unlike in experiments, it is possible to artificially remove hydrodynamic interactions between particles in the simulations. Molecular dynamics simulations of drying were performed using both explicit and implicit solvent models. We show that although the implicit-solvent model stratifies for the drying conditions investigated, the explicit-solvent model does not due to the presence of hydrodynamic interactions. Our simulations show that it is critical to incorporate hydrodynamic interactions into models and simulations in order to reliably predict stratification in drying mixtures.

6.1 Models and Methods

We performed nonequilibrium molecular dynamics simulations of drying polymer mixtures using two bead–spring models: one with an explicitly resolved solvent (Section 6.1.1) and a corresponding one with an implicit solvent (Section 6.1.2). While the explicit model included full hydrodynamic interactions, the implicit model only incorporated the Stokes drag on each polymer, thus omitting hydrodynamic interactions between polymers. Details of both models are presented next.

6.1.1 Explicit solvent model

Polymers were represented as linear chains of monomers with diameter $\sigma$ and mass $m$ immersed in an explicit solvent of beads of equal size and mass. All monomers
and solvent particles interacted with the Lennard-Jones potential,

\[ U_{nb}(r) = 4\varepsilon \left[ \left( \frac{\sigma}{r} \right)^{12} - \left( \frac{\sigma}{r} \right)^{6} \right], \quad (6.1) \]

where \( \varepsilon \) is the interaction energy and \( r \) is the distance between particles. The monomer–monomer interactions were made purely repulsive by truncating the potential at its minimum and shifting it to zero [83]. The solvent–solvent and monomer–solvent interactions included the attractive part of \( U_{nb} \) and were truncated at 3.0 \( \sigma \) with the energy and forces smoothed to zero starting from 2.5 \( \sigma \). Bonds between monomers were represented by the finitely extensible nonlinear elastic potential, [85]

\[ U_b(r) = \begin{cases} 
-\frac{1}{2}\kappa r_0^2 \ln \left[ 1 - \left( \frac{r}{r_0} \right)^2 \right], & r < r_0 \\
\infty, & r \geq r_0
\end{cases}, \quad (6.2) \]

with the standard parameters \( r_0 = 1.5\sigma \) and \( \kappa = 30\varepsilon/\sigma^2 \). This choice prevents unphysical bond crossing because the equilibrium bond length is approximately 0.97 \( \sigma \) at a temperature of \( T = 1.0\varepsilon/k_B \), where \( k_B \) is Boltzman’s constant. This model corresponds to good solvent conditions for the polymers.

The drying process was simulated using the method developed by Cheng and coworkers [54, 206, 207]. The polymer solution was supported by a smooth, structureless substrate modeled by a Lennard-Jones 9–3 potential at the bottom of the simulation box,

\[ U_w(z) = \varepsilon_w \left[ \frac{2}{15} \left( \frac{\sigma}{z} \right)^9 - \left( \frac{\sigma}{z} \right)^3 \right], \quad (6.3) \]

where \( z \) is the distance between the particle and the substrate, and \( \varepsilon_w = 2\varepsilon \) is the strength of the interaction. Interactions were truncated for \( z > 3.0\sigma \). The solvent vapor above the liquid film was confined by an additional potential of the same form as \( U_w \) at the top of the simulation box, which was made purely repulsive.
by truncating it for \( z > (2/5)^{1/6} \sigma \). Solvent was evaporated by deleting a small number of randomly-chosen solvent particles from the top 20 \( \sigma \) of the vapor. To maintain temperature control, monomers and solvent in a slab of height 20 \( \sigma \) above the substrate were weakly coupled to a Langevin thermostat with \( T = 1.0 \varepsilon / k_B \) and friction coefficient 0.1 \( m/\tau \) \([160–162]\), where \( \tau = \sqrt{m\sigma^2/\varepsilon} \) is the derived unit of time.

All simulations were performed using the HOOMD-blue simulation package (version 2.2.2) on multiple graphics processing units \([122, 123]\) with a timestep of \( \Delta t = 0.005 \tau \). The solvent coexistence densities were 0.664 \( \sigma^{-3} \) and 0.044 \( \sigma^{-3} \) for the liquid and vapor phases, respectively. The simulation box was periodic in the \( x \) and \( y \) directions with a length of \( L = 50 \sigma \). The height of the box was \( L_z = 540 \sigma \), where \( H = 500 \sigma \) was the initial film height and the remaining 40 \( \sigma \) was filled with solvent vapor. We investigated a mixture of short polymers of \( M_1 = 10 \) monomers and long polymers of \( M_2 = 80 \) monomers at an initial monomer density of 0.006 \( \sigma^{-3} \) each. The simulations consisted of 730000 solvent particles, 900 long chains, and 7100 short chains, resulting in a total of 873000 particles. By deleting one solvent particle every 1.25\( \tau \), we obtained an evaporative flux of \( 3.2 \times 10^{-4}/\sigma^2\tau \).

### 6.1.2 Implicit solvent model

The implicit-solvent model was constructed by matching the structure of the polymer chains in the explicit solvent at infinite dilution. Interactions between the monomers and the solvent swelled the polymers beyond the size of a chain with only the monomer–monomer interactions, so additional effective interactions were required. We measured the distance distribution between monomers separated by two bonds along a chain surrounded by solvent and a chain without solvent. We fit the negative logarithm of the ratio of the explicit and implicit distribution with a
spline potential [208] of the form

\[ U_s(r) = \begin{cases} 
\varepsilon_s, & r < r_s \\
\varepsilon_s(r_c^2 - r)^n r_s^2 + n r_s^2 - (n + 1) r_s^2 \cfrac{r_c^2 - r_s^2}{r_c^2 - r_s^2}^{n+1}, & r_s \leq r < r_c \\
0, & r \geq r_c 
\end{cases} \]  

(6.4)

The fit resulted in parameters \( n = 3.8, \varepsilon_s = 0.68 \varepsilon, r_s = 1.32 \sigma, \) and \( r_c = 2.0 \sigma. \) This potential, modeling the soft repulsion between monomers due to the solvent, was added to the bare monomer–monomer interactions. We validated this effective model by measuring the polymer structure over a range of concentrations and compositions up to a total monomer number density of \( 0.375 \sigma^{-3}, \) finding overall good agreement between the implicit and explicit models (see Appendix 6.A and Appendix 6.B).

The polymer long-time dynamics in the explicit solvent were matched in the implicit solvent using Langevin dynamics simulations [160–162]. This technique incorporates the effects of Brownian motion and Stokes drag from the solvent, but neglects hydrodynamic interactions between monomers. The monomer friction coefficients were adjusted for each polymer to give the same polymer center-of-mass diffusion coefficient at infinite dilution as in the explicit solvent. This approach gives the correct long-time dynamics, but distorts the internal relaxation modes of the polymers as discussed in Chapter 5. We measured diffusion coefficients of \( D_1 = 0.0342 \sigma^2/\tau \) for the \( M_1 = 10 \) polymers and \( D_2 = 0.0071 \sigma^2/\tau \) for the \( M_2 = 80 \) polymers from the polymer center-of-mass mean squared displacement in a cubic box with edge length \( L = 50 \sigma, \) giving \( 2.92 m/\tau \) and \( 1.76 m/\tau \) for the friction coefficients, respectively.

The liquid–vapor interface was modeled by the repulsive part of a harmonic potential (see Eq. 4.5). In order to closely match the explicit-solvent simulations, the spring constant and position of the interface were adjusted to obtain similar initial
density profiles to the explicit-solvent model. We used spring constants of $0.1\varepsilon/\sigma^2$ and $0.18\varepsilon/\sigma^2$ for the monomers of the short and long chains, respectively. The position of the interface was measured throughout the explicit-solvent simulations and used directly in the implicit-solvent simulations. The minimum of the potential was offset for the long polymers by $-1\sigma$.

### 6.2 Results and Discussion

We performed 25 drying simulations for both models and measured the monomer density profiles in the film. Figure 6.1 shows the average profiles at three times: before evaporation, at a film height of roughly $3H_0/4$, and at a film height close to $H_0/2$. We used the average interface speed and the diffusion coefficients at infinite dilution to estimate film Péclet numbers of $\text{Pe}_1 \approx 7$ and $\text{Pe}_2 \approx 33$. (Throughout this discussion, 1 denotes the shorter polymer, while 2 is the longer polymer.) Initially, the polymers were nearly uniformly distributed in the film. When drying began, both the long and short polymers accumulated at the moving interface, as expected from their Péclet numbers. However, there was a noticeable difference in the distribution of chains within the film between the two models. Consistent with previous simulations and the theoretical model of Chapter 5, the implicit-solvent model stratified with the shorter polymers on top of the longer polymers (Figure 6.1b). In stark contrast, the explicit-solvent model showed no small-on-top stratification (Figure 6.1a), and in fact more long polymers than short polymers accumulated immediately below the liquid–vapor interface.

The qualitatively different behavior of the two models suggests a significant difference in the relative migration speeds of the polymer chains, which was previously shown to give rise to stratification for the implicit-solvent model. At isothermal conditions, the relative migration speed $\Delta u$ predicted by model of Chapters 4 and 5
Figure 6.1: Monomer density profiles for the explicit-solvent model (top) and implicit-solvent model (bottom) for three different times during evaporation. Solid lines show the profiles of the long ($M_2 = 80$) polymer, while dashed lines are the profiles of the short ($M_1 = 10$) polymer.

is

$$\Delta u / u_1 = \frac{M_2 (\partial \mu_2 / \partial z)}{M_1 (\partial \mu_1 / \partial z)} - 1$$

(6.5)

where $u_1$ is the migration speed of the short polymers, $\mu_i$ is the chemical potential of component $i$ computed from the equilibrium free-energy functional, and $M_i$ is the effective mobility for component $i$. We previously estimated $M_i$ from the equilibrium diffusion coefficient $D_i$ using the Einstein relation, $M_i = D_i / k_B T$, within the free-draining approximation of hydrodynamics. Our simulations and Eq. 6.5 then suggest several possible explanations for the different drying behavior:

1. temperature gradients from evaporative cooling affect $\mu_i$ or $M_i$ in the explicit model, but are absent from the implicit model,

2. differences in the chemical potentials of the two models lead to different diffusive driving forces,
3. differences in the equilibrium diffusion coefficients change $\mathcal{M}_i$ for the two models, and/or

4. hydrodynamic interactions mediated by the explicit solvent alter $\mathcal{M}_i$ and the migration velocities.

We probed each of these effects in turn. As will be shown, we found that only hydrodynamic interactions (Section 6.2.4) had a significant enough effect to explain the lack of stratification in the explicit-solvent model.

### 6.2.1 Temperature

Solvent evaporation can lead to cooling at fast drying rates [206]. For our explicit-solvent model, the temperature is expected to be fixed at $T = 1.0 \, \varepsilon/k_B$ close to the substrate and, at pseudo-steady state, to decrease linearly in distance from the substrate with a slope proportional to the evaporative flux. The temperature profile $T(z)$ measured in the simulations, shown in Figure 6.2a, is consistent with this expectation. Here, the local temperature $T(z)$ is defined by

$$T(z) = \frac{m}{3N_k k_B} \sum_{j=1}^{N_k} v_j^2$$

(6.6)

where the sum is taken over the $N_k$ particles in a slab $k$ centered around $z$ with thickness $2\Delta z = 1 \, \sigma$, and $v_j$ is the velocity of particle $j$. On the other hand, there were no temperature gradients in the implicit-solvent model because the solvent was treated as an isothermal, viscous background.

Temperature gradients can give rise to mass flux (i.e., the Soret effect) [209], and the mobility is temperature dependent through the viscosity [125]. In order to test how the presence of temperature gradients influenced diffusion in the explicit-solvent model, we coupled all particles to a dissipative particle dynamics (DPD) thermostat
Figure 6.2: Temperature (top) and long polymer ($M_2 = 80$) monomer density (bottom) profiles in the explicit solvent with (dashed line) and without (solid line) a DPD thermostat. The $z$ position is normalized by the initial film height $H = 500 \sigma$. Profiles are shown at 275000 $\tau$ with the thermostat and 250000 $\tau$ without the thermostat.

The DPD thermostat applies pairwise random and dissipative forces to all particles, and so conserves momentum and preserves hydrodynamic interactions. We set the DPD friction coefficient to be $0.5 m/\tau$, which has been shown to have a negligible effect on the viscosity of a fluid of nearly-hard spheres [211]. Figure 6.2a shows that although there was originally roughly 12% evaporative cooling in the explicit-solvent model, the DPD thermostat maintained a constant temperature throughout the film.

We then compared the distribution of polymers in the film with and without evaporative cooling. The temperature gradient in the film led to a corresponding gradient in the the local total density (solvent and polymer), which was higher in colder regions. This in turn decreased the total film height compared to the isothermal case for equal amount of solvent evaporated. To account for this difference, we compared the monomer density profiles at equal film height. The profile for the $M_2 = 80$ polymers is shown in Figure 6.2b at film height 360 $\sigma$, when the temperature
gradient was most pronounced. The monomer density profiles were indistinguishable with and without evaporative cooling. Accordingly, we excluded temperature effects as a possible reason for the different stratification behavior in the implicit- and explicit-solvent models.

6.2.2 Chemical potential

In the absence of temperature gradients, diffusive flux is driven by chemical potential gradients within the regime of linear response [212]. A mismatch in the density or composition dependence of the chemical potentials could result in different stratification behavior for the explicit- and implicit-solvent models. We accordingly measured the relevant chemical potentials for both models using test insertion methods. We first generated trajectories of polymer mixtures in cubic, periodic simulation boxes over a wide range of mixture compositions (see Appendix 6.B), saving 100 independent configurations for each composition.

For the explicit-solvent mixtures, the chemical potential of the solvent $\mu_0$ was calculated by Widom’s insertion method [213]

$$
\mu_0 = \mu_0^{id} - k_B T \ln \langle e^{-\beta \Delta U} \rangle 
$$

where $\beta = 1/k_B T$, $\mu_0^{id}$ is the chemical potential of an ideal gas of particles at the same temperature and density as the solvent in the mixture, and $\Delta U$ is the change in the potential energy on insertion of a test particle. The ensemble average is taken over configurations and insertions. We performed $5 \times 10^5$ insertions per configuration, which we found to be sufficient to obtain a converged value for $\mu_0$.

The chemical potential of the polymers was estimated by the chain increment method [214] for both the explicit- and implicit-solvent models. The incremental chemical potential $\mu^+(M)$ to grow a chain of length $M + 1$ from a chain of length $M$
is
\[ \beta \mu^+(M) = -\ln\langle e^{-\beta \Delta U} \rangle \] (6.8)
where the ensemble average is now taken over configurations and test insertions onto
the ends of chains of length \( M \). The incremental chemical potential \( \mu^+(M) \) converges
to a constant value \( \mu^+(\infty) \) for sufficiently large \( M \) [214]. The greatest deviations from
\( \mu^+(\infty) \) occur for a single monomer, \( \mu^+(0) \), which can instead be measured by Widom
insertion [213]. The chemical potential of the chain was then approximated as [215]
\[ \mu_i = \mu_i^{id} + \mu^+(0) + (M_i - 1)\mu^+(\infty) \] (6.9)
where \( \mu_i^{id} \) is the chemical potential of an ideal gas of chains of length \( M_i \) at the same
chain density and temperature as in the mixture.

In order to improve convergence of the ensemble averages, test particles were in-
serted onto the chain ends at positions consistent with the bond length distribution
[215, 216]. The test particle position was drawn with random orientation relative
to the chain end at distance \( r \) distributed according to \( r^2 \exp(-\beta \phi(r)) \) in the range
\( 0.7 \sigma \leq r \leq 1.3 \sigma \), where \( \phi(r) \) is the interaction potential between bonded monomers
given by Eq. 6.1 and Eq. 6.2. An additional analytical contribution to the chemical
potential was then included to account for the weighted sampling [215]. To mea-
sure \( \mu^+(10) \) and \( \mu^+(80) \), we performed 100 trial insertions per chain end in each
configuration and, finding that \( \mu^+(10) \approx \mu^+(80) \) within statistical uncertainty, took
\( \mu^+(\infty) \approx \mu^+(10) \). The monomer excess chemical potential, \( \mu^+(0) \), was measured
using \( 5 \times 10^5 \) random insertions per configuration, as for the solvent.

In our implicit-solvent theoretical model (Eq. 6.5), the diffusive driving force was
given by the gradient of \( \mu_i \). However, this expression must be modified to account
for the presence of the solvent in the explicit model. If the mixture is approximately
incompressible, the solvent and polymer concentrations are not independent. As-
summing that the the volume occupied by one polymer chain of length $M_i$ is roughly equal to $M_i$ solvent particles, the chemical potential $\mu_i$ is replaced in Eq. 6.5 by the exchange chemical potential $\Delta \mu_i = \mu_i - M_i \mu_0$ [217, 218]. For the implicit model, the solvent is effectively incorporated into the free energy of the polymers, giving $\Delta \mu_i = \mu_i$ and recovering Eq. 6.5 as expected. Batchelor showed that the diffusion process resulting from gradients of $\Delta \mu_i$ is equivalent to sedimentation of the solute (polymers) in a force-free solvent [218, 219].

Although $\Delta \mu_i$ was determined across a wide range of mixture compositions (Figure 6.3), we found that it was essentially only a function of the total monomer density $\rho$. As expected, $\Delta \mu_i$ increased for larger $\rho$. This is in qualitative agreement with Chapter 5, where the excess chemical potential per monomer depended primarily on the total monomer packing fraction and more weakly on the total chain number density. Most importantly, $\Delta \mu_i$ is in excellent agreement between the two models for both the short and long polymers. We then expect both the explicit- and implicit-solvent models to give equivalent driving forces for diffusion and stratification.

![Figure 6.3: Exchange chemical potential per monomer, $\beta \Delta \mu_i/M_i$, for the short ($M_1 = 10$) and long ($M_2 = 80$) polymers as a function of total monomer density $\rho$ for the explicit- and implicit-solvent models. Each point corresponds to a different mixture composition (see Appendix 6.B).](image-url)
6.2.3 Equilibrium diffusion coefficient

Having found good agreement in the exchange chemical potential between the two models, we considered dynamic effects in Eq. 6.5. The polymer mobility $\mathcal{M}_i$ relates the diffusive driving force on the polymers to the migration velocity. In our theoretical model, we assumed that $\mathcal{M}_i$ could be estimated from the equilibrium diffusion coefficient $D_i$. In our simulations, $D_i$ for the implicit-solvent model was matched to the explicit-solvent model in the limit of infinite dilution, but the models may deviate at higher concentrations. Differences in the concentration-dependence of $D_i$, and hence $\mathcal{M}_i$, may accordingly influence the relative migration speeds of the components during drying.

We measured $D_i$ for both the explicit- and implicit-solvent models across a wide range of concentrations and compositions from the mean-squared displacement of the polymer centers of mass. The values of all measured diffusion coefficients are available in Appendix 6.B. At low total monomer densities, the agreement between the two models was good, as expected from the model fitting, with larger discrepancies at higher monomer densities. Over the range of compositions relevant to the drying simulations, the agreement between the two models was quite good, with a deviation of at most 20% for the short polymers and 30% for the long polymers relative to the explicit-solvent model.

According to Eq. 6.5, a larger value of $\mathcal{M}_2/\mathcal{M}_1$, and so approximately $D_2/D_1$, should result in a larger difference in migration velocities for the same chemical potential gradient and lead to stronger stratification. For all concentrations and compositions investigated, $D_2/D_1$ was larger for the explicit-solvent model (Figure 6.4). This suggests that stronger stratification might be expected for the explicit model on the basis of the diffusion coefficients, which is the opposite of the observed behavior. Accordingly, we concluded that differences in the equilibrium diffusion coefficients were not likely the reason for lack of stratification in the explicit-solvent model.
6.2.4 Hydrodynamic interactions

Given that temperature gradients, chemical potentials, and equilibrium diffusion coefficients do not explain the difference in stratification in the implicit- and explicit-solvent models, it remains now to test the influence of hydrodynamic interactions. Hydrodynamic interactions are inherently present in the explicit-solvent model, but are absent from the implicit-solvent Langevin dynamics simulations. The recent analysis by Sear and Warren demonstrated how solvent backflow reduces the phoretic motion of a large colloid in an ideal polymer solution [204]. Brady showed from a rigorous microscopic perspective how this fundamentally results from a neglect of hydrodynamic interactions, which modify the mobility [205].

We performed a simple simulation to test the effect of hydrodynamic interactions on polymer migration. We constructed a simulation box with \( L = 40 \sigma \) and \( L_z = 80 \sigma \). Half of the box \((z > 0)\) was filled with pure solvent, while the other half \((z < 0)\) was filled with a solution of either long or short polymers. A semipermeable membrane at \( z = 0 \sigma \) separated the two compartments, and walls were placed at \( z = \pm 40 \sigma \).
Figure 6.5: Fraction of monomers for short ($M_1 = 10$) and long ($M_2 = 80$) chains diffusing into the with $z > 0$ over time. Initially, all monomers have $z < 0$. Solid line is the explicit solvent model, while dashed lines are the implicit solvent model.

Interactions with the walls were given by Eq. 6.3 truncated at distances greater than $3\sigma$ with $\varepsilon_w = 1\varepsilon$, while the monomer interactions with the membrane were modeled by Eq. 6.3 truncated at $(2/5)^{1/6}\sigma$ with $\varepsilon_w = 1\varepsilon$. The solvent was allowed to exchange across the membrane, coming to equal chemical potential, but the polymers were confined to $z < 0\sigma$. For the explicit-solvent model, there were 78720 solvent particles in the box and the initial monomer density of the polymers in the compartment was $\rho = 0.2/\sigma^3$ (12800 monomers). The polymers in the explicit solvent were used as starting configurations for the implicit-solvent model.

We subsequently removed the membrane and allowed the polymers to diffuse. Based on Figure 6.3, the initial gradient in $\Delta\mu_i$ across the membrane should be comparable for both models, and so any differences in the diffusive flux should be due to hydrodynamic interactions through the mobility. Figure 6.5 shows the fraction of monomers which moved into the opposite side of the box ($z > 0\sigma$) at a given time, averaged over 25 independent simulations. At steady state, half of the monomers should have $z > 0\sigma$. It is clear that the dynamics of the implicit-solvent model are faster than the explicit-solvent model, in agreement with the considerations in Ref. 204 and Ref. 205. Moreover, the long ($M_2 = 80$) polymers and short ($M_1 = 10$)
polymers responded differently to the same density gradient between the two models. In the implicit-solvent model, the long polymers migrated faster than the short polymers, consistent with Eq. 6.5 and stratification in the drying simulations. On the other hand, the long polymers migrated much slower than the short polymers in the explicit-solvent model, explaining the lack of stratification in the drying simulations.

6.3 Conclusions

We demonstrated the influence of hydrodynamic interactions on stratification in a drying polymer mixture using explicit-solvent and implicit-solvent computer simulations. The implicit-solvent model predicted stratification in agreement with previous simulations and theoretical considerations (Chapter 5). However, no such stratification was found for the explicit-solvent model at the drying conditions considered. Despite good mapping of the equilibrium bulk properties (chemical potential, diffusion coefficient) between the explicit- and implicit-solvent models, hydrodynamic interactions out of equilibrium were shown to alter the polymer diffusion in a way that is consistent with the lack of stratification. Our analysis directly tests and confirms that implicit-solvent simulations and theoretical models lacking hydrodynamic interactions are not capable of quantitatively predicting stratification, in agreement with the analysis of Sear and Warren [204] for the special case of a large colloid in an ideal polymer solution. In future, hydrodynamic interactions must be incorporated into any simulations aiming to study stratification, e.g., through explicit-solvent molecular dynamics or with an appropriate mesoscale simulation method.

While there are currently no experiments available for stratification in polymer mixtures, colloid mixtures have been shown to stratify in experiments [51–53]. The extent of colloid stratification in the experiments appears weaker than predicted by models lacking hydrodynamic interactions [51], consistent with our analysis. Larger
chemical potential gradients on the larger component would be required to increase the stratification, which could be induced by, for example, larger size ratios or additional cross-interactions between the components.

6.A Single Chain Structure

We measured the structure of a single $M = 80$ chain in solvent at a bulk density of $0.664\sigma^{-3}$ for the explicit-solvent model and compared it to the implicit-solvent model of the same chain. The distributions of distances between monomers separated by $n$ bonds, $r_n$, are shown in Figure 6.6a–b for both models. The distribution of bonded monomers ($r_1$) and next nearest neighbors ($r_2$) are in very good agreement. This was expected because $r_1$ is mainly determined by the bond potential, which is identical for both models, and $r_2$ was used to fit the pair potential of the implicit model. More significantly, we found that $r_n$ for monomers separated by 3 or more bonds were also in good agreement (Figure 6.6c–e even though these distributions were not fit. We additionally determined the average radius of gyration $\sqrt{\langle R_g^2 \rangle}$ of both models for varying chain length $M$, shown in Figure 6.6f. The excellent agreement for all these metrics demonstrates that the two models reproduce the same internal polymer chain structure at infinite dilution.

6.B Bulk Mixture Properties

Equilibrium properties of the bulk mixtures were computed for a range of polymer concentrations. We equilibrated mixtures of different compositions in the explicit solvent in the $NpT$ ensemble for $4 \times 10^5 \tau$, at $p = 0 \varepsilon/\sigma^3$ and $T = 1.0 \varepsilon/k_B$. The number of polymer chains ranged from $N_1 = 625, \ldots, 2500$ short and $N_2 = 78, \ldots, 312$ long polymers with a fixed number of total particles of 83000, leading to mass fractions $x_i$ from 0.075 up to 0.301 for each polymer. The equilibrium bulk densities ranged from
0.725\sigma^{-3} \text{ to } 0.621\sigma^{-3} (Table 6.1). The density first increased with concentration but subsequently decreased. Adding bonds with an equilibrium bond length shorter than the preferred average solvent distance to the system initially increased the density. However, adding more polymers into the system also decreased the number of attractive interactions and increased the number of purely repulsive interactions, ultimately decreasing the density. We subsequently determined the chain structure, diffusion coefficients, and chemical potentials for the explicit and implicit solvent models from \textit{NVT} simulations at the average equilibrium density.

Table 6.1: Bulk density in units of $\sigma^{-3}$ for mixtures of different numbers of short ($N_1$) and long ($N_2$) polymers at $T = 1.0 \varepsilon/k_B$ and $p = 0 \varepsilon/\sigma^3$.

\begin{tabular}{c|cccc}
$N_1 \rightarrow$ & 625 & 1250 & 1875 & 2500 \\
$N_2 \downarrow$ & 78 & 0.707 & 0.718 & 0.722 & 0.715 \\
156 & 0.721 & 0.725 & 0.718 & 0.700 \\
234 & 0.728 & 0.721 & 0.703 & 0.671 \\
312 & 0.724 & 0.707 & 0.675 & 0.621 \\
\end{tabular}
We measured the internal polymer chain structure at $x_1 = x_2 = 0.075$, which is slightly higher than the initial condition of the drying film, and at $x_1 = x_2 = 0.301$, which is a higher fraction of polymers than occurred during the drying simulations. (Mass fractions for the implicit model are reported as if solvent would be present in the system in order to facilitate comparison with the explicit model.) Since the implicit-solvent model was fit for a single polymer chain in solution at infinite dilution, it was expected to incur some error at higher polymer concentrations where less solvent is present. As shown in Figure 6.7, the structure of a polymer chain is in reasonable agreement between the two models even at these higher concentrations.

Figure 6.7: Distance distributions for monomers along the chain (a) $r_2$, (b) $r_3$, and (c) $r_4$ for mass fractions $x_1 = x_2 = 0.075$ and $x_1 = x_2 = 0.301$ for the explicit model (solid lines) and implicit (dashed lines).

Diffusion coefficients were measured from three sequential simulations of $5 \times 10^5 \tau$ each by averaging the polymer center-of-mass mean-squared displacement. The computed values can be found in Tables 6.2–6.5 and are additionally visualized as a function of mass fraction composition. The two models were in reasonable agreement with each other for mass fractions $x_i \lesssim 0.2$, but started to deviate substantially at higher concentrations of both long and short polymers. The explicit model showed enhanced diffusivity for large concentrations, whereas the implicit model had a decreasing diffusivity with increasing polymer concentration. We attribute this unusual behavior for the explicit-solvent model to the changing balance of attractive and repulsive interactions in the mixture as polymer concentration increases significantly, which
is accompanied by lower density and likely also a lower viscosity. Nonetheless, the implicit-solvent diffusion coefficients for densities relevant to the drying film deviate from the explicit-solvent model by at most about 30% for the long polymers and about 20% for the short polymers.

Table 6.2: Equilibrium diffusion coefficients for the \( M_1 = 10 \) polymer, \( D_1[\sigma^2/\tau] \), in the explicit model.

<table>
<thead>
<tr>
<th>( N_1 \rightarrow N_2 )</th>
<th>625</th>
<th>1250</th>
<th>1875</th>
<th>2500</th>
</tr>
</thead>
<tbody>
<tr>
<td>78</td>
<td>0.0201</td>
<td>0.0177</td>
<td>0.0164</td>
<td>0.0168</td>
</tr>
<tr>
<td>156</td>
<td>0.0155</td>
<td>0.0147</td>
<td>0.0152</td>
<td>0.0168</td>
</tr>
<tr>
<td>234</td>
<td>0.0131</td>
<td>0.0137</td>
<td>0.0155</td>
<td>0.0189</td>
</tr>
<tr>
<td>312</td>
<td>0.0123</td>
<td>0.0142</td>
<td>0.0176</td>
<td>0.0239</td>
</tr>
</tbody>
</table>

Table 6.3: Equilibrium diffusion coefficients for the \( M_2 = 80 \) polymer, \( D_2[\sigma^2/\tau] \), in the explicit model.

<table>
<thead>
<tr>
<th>( N_1 \rightarrow N_2 )</th>
<th>625</th>
<th>1250</th>
<th>1875</th>
<th>2500</th>
</tr>
</thead>
<tbody>
<tr>
<td>78</td>
<td>0.0037</td>
<td>0.0033</td>
<td>0.0032</td>
<td>0.0033</td>
</tr>
<tr>
<td>156</td>
<td>0.0026</td>
<td>0.0025</td>
<td>0.0026</td>
<td>0.0030</td>
</tr>
<tr>
<td>234</td>
<td>0.0020</td>
<td>0.0022</td>
<td>0.0026</td>
<td>0.0032</td>
</tr>
<tr>
<td>312</td>
<td>0.0018</td>
<td>0.0021</td>
<td>0.0027</td>
<td>0.0039</td>
</tr>
</tbody>
</table>

Table 6.4: Equilibrium diffusion coefficients for the \( M_1 = 10 \) polymer, \( D_1[\sigma^2/\tau] \), in the implicit model.

<table>
<thead>
<tr>
<th>( N_1 \rightarrow N_2 )</th>
<th>625</th>
<th>1250</th>
<th>1875</th>
<th>2500</th>
</tr>
</thead>
<tbody>
<tr>
<td>78</td>
<td>0.022</td>
<td>0.0191</td>
<td>0.0168</td>
<td>0.0152</td>
</tr>
<tr>
<td>156</td>
<td>0.019</td>
<td>0.0165</td>
<td>0.0149</td>
<td>0.0135</td>
</tr>
<tr>
<td>234</td>
<td>0.016</td>
<td>0.0146</td>
<td>0.0133</td>
<td>0.0123</td>
</tr>
<tr>
<td>312</td>
<td>0.014</td>
<td>0.0131</td>
<td>0.0121</td>
<td>0.0114</td>
</tr>
</tbody>
</table>

Table 6.5: Equilibrium diffusion coefficients for the \( M_2 = 80 \) polymer, \( D_2[\sigma^2/\tau] \), in the implicit model.

<table>
<thead>
<tr>
<th>( N_1 \rightarrow N_2 )</th>
<th>625</th>
<th>1250</th>
<th>1875</th>
<th>2500</th>
</tr>
</thead>
<tbody>
<tr>
<td>78</td>
<td>0.0036</td>
<td>0.0031</td>
<td>0.0026</td>
<td>0.0024</td>
</tr>
<tr>
<td>156</td>
<td>0.0028</td>
<td>0.0025</td>
<td>0.0022</td>
<td>0.0012</td>
</tr>
<tr>
<td>234</td>
<td>0.0024</td>
<td>0.0021</td>
<td>0.0019</td>
<td>0.0017</td>
</tr>
<tr>
<td>312</td>
<td>0.0020</td>
<td>0.0018</td>
<td>0.0017</td>
<td>0.0015</td>
</tr>
</tbody>
</table>
Chapter 7

Evaporation-induced assembly of colloidal crystals

Chapters 4–6 considered the formation of layered structures in drying mixtures; however, within each layer there was limited long-range order. This chapter focuses on how such crystalline structures form in a dispersion of colloids of uniform size. Colloidal crystals with long-range order are highly desirable materials for applications such as photonics [56] and optics [55]. There is particular interest in developing inexpensive, high-throughput fabrication methods for such materials by self-assembly from solution [22]. However, colloidal particles typically have weak interactions on the order of thermal energy, and competing crystal structures are stabilized against each other by only small differences in free energy [25–27], resulting in polycrystalline solids [23, 24]. Selectivity toward crystal structures has been enhanced by designing particle interactions with DNA coatings [220], patches [221, 222], depletants [18, 223], and shape [224, 225]. However, even such carefully engineered particles must still dynamically assemble into the desired thermodynamically stable crystal.

The nonequilibrium aspects of self-assembly, including the assembly mechanism and processing conditions, therefore present important additional opportunities and challenges to control crystal structures.

One common high-throughput method for fabricating colloidal crystals involves dispersal of particles in a volatile solvent followed by evaporation of the solvent to deposit a crystalline solid onto a horizontal or vertical substrate [22]. This assembly process is sensitive to, among many factors, the effective particle interactions [226–228] and the evaporation rate as controlled by temperature [229, 230], pressure [231–233], relative humidity [234], and solvent properties [235]. For slow evaporation onto a horizontal substrate, crystals form in the bulk once the concentration is sufficiently high [236], whereas for fast evaporation, particles accumulate at the drying interface to form well-ordered monolayers [237] or multilayers [230]. Experiments [230] and simulations [207] have shown that the best crystal quality at the drying interface is obtained when drying is sufficiently fast to increase the local particle density but slow enough that diffusion anneals defects before additional grains or layers form. Much less is known about how three-dimensional crystals form, although it has been hypothesized that bulk crystal growth proceeds by accumulation and incorporation of particles below the surface monolayer [230].

Computer simulations with time-resolved microscopic detail are useful tools to study how evaporation influences crystal formation and growth, which can be challenging to monitor experimentally. Recent simulations without hydrodynamic interactions suggested that crystallinity may exhibit a minimum at intermediate drying rates due to a competition between crystallization at the interface and nucleation in the bulk [238]. Although hydrodynamic interactions are frequently neglected from simulations for computational efficiency, they often play an important role in assembly dynamics, e.g., selecting binary crystal structures for DNA-linked colloids [239]. To our knowledge, there has only been one previous simulation study of evaporation-
induced colloid crystallization with full hydrodynamic interactions [207], but it was computationally restricted to 4 layers of colloids. The effects of evaporation and hydrodynamics on long-ranged three-dimensional crystal morphology thus remain unexplored.

We performed massive-scale nonequilibrium molecular dynamics simulations to study the evaporation-induced assembly of colloidal crystals from solution onto a horizontal substrate. A machine-learning technique was applied to characterize local structures in the growing crystal including particles at interfaces and defects, which are challenging to identify by traditional methods. Our simulations provide mechanistic insight into evaporation-induced crystallization, which is found to occur in three steps: (1) accumulation of particles at the drying interface, (2) formation of a sufficiently ordered monolayer with subsequent bulk crystal growth templated by this surface structure, and (3) deformation of the bulk crystal due to confinement effects. We find that the time evolution of this process is well-described by the characteristic time scales of drying in step 1 and crystal growth in step 2. We also demonstrate the importance of the solvent in controlling the crystallization dynamics using complementary implicit-solvent simulations without hydrodynamic interactions. Ultimately, we find that the final crystal structure does not depend significantly on the dynamics during drying for the range of parameters considered, as the eventual deformation of the confined crystal in step 3 disrupts initial surface ordering.

7.1 Results and Discussion

Nearly-hard spherical colloids were dispersed into a liquid solvent of Lennard-Jones particles in coexistence with its vapor [207, 240, 241]. In this article, quantities are reported using the system of units defined by the diameter $\sigma$, mass $m$, and interaction energy $\varepsilon$ of the solvent particles. Evaporation was modeled by the deletion of solvent
from the vapor phase at a controlled flux [207]. The initial volume fraction based on the colloid diameter \((d = 10\sigma)\) was \(\phi_0 \approx 0.24\), which is comparable to previous simulation studies [207, 238] but larger than typical concentrations in experiments [230, 236]. We show how using a higher initial concentration presents an important opportunity to control the crystal growth mechanism.

The dimensionless film Péclet number, \(\text{Pe} = H_0 v / D_0\), characterizes the evaporation speed as the ratio of the rate of colloid accumulation at the liquid–vapor interface relative to the rate of diffusion, where \(H_0\) is the initial film height, \(v\) is the interface speed, and \(D_0\) is the colloid diffusion coefficient. When \(\text{Pe} \ll 1\), diffusion is fast compared to accumulation, and colloids are expected to be uniformly distributed in the film, eventually leading to nucleation in the bulk [236, 238]. When \(\text{Pe} \gg 1\), colloids are instead expected to accumulate at the liquid–vapor interface, leading to surface-induced nucleation [230, 236, 237]. We first studied the mechanism of crystal growth for evaporative flux \(j = 3.6 \times 10^{-4} \text{m/}\sigma^2 \tau\), which corresponds to \(\text{Pe} \approx 58\), and then varied the flux a factor of two slower (\(\text{Pe} = 29\)) and faster (\(\text{Pe} = 118\)). Multiple independent simulations were conducted for each value of \(\text{Pe}\) due to the stochastic nature of assembly pathways. Local crystalline structures were characterized for each colloid using the template-free neighborhood graph analysis (NGA) method [242]. Complete details are available in Appendix 7.A.

Colloids were initially distributed uniformly in the film, but accumulated at the interface after drying began (Figure 7.1). Heterogeneous nucleation of crystallites at the solvent vapor–liquid interface occurred in all our simulations once the surface layer became sufficiently dense (Figure 7.1a). The crystal front then moved down into the bulk liquid at a nearly constant velocity (Figure 7.1b–c). Once all particles in the film crystallized (Figure 7.1d), the crystal continued to compact as the solvent was dried further and the interparticle spacing decreased, eventually reaching a polymor-

\(^{1}\) Initially, \(H_0 = 205\sigma\), \(D_0 = 1.95 \times 10^{-3} \sigma^2 / \tau\) and \(v \approx j / \rho = 5.5 \times 10^{-4} \sigma / \tau\) with \(\rho = 0.66 \text{m/}\sigma^3\) being the solvent liquid density and \(\tau = \sqrt{m\sigma^2 / \varepsilon}\) being the unit of time.
Figure 7.1: Snapshots from representative simulation trajectories at $\text{Pe} = 58$ for the explicit-solvent (top) and implicit-solvent (bottom) models. Snapshots in column (a) are at $t = 37,500\tau$ when the film height is $H_x$ (see Figure 7.2), and subsequent snapshots (b–f) are shown every $25,000\tau$. Rendered using Visual Molecular Dynamics 1.9.2 [121] and colored according to each particle’s position in the diffusion map presented in Figure 7.4. The colors can be roughly mapped onto the following structures: HCP (red), FCC (dark blue), FCC (111) surface (light blue), FCC (100) surface (teal), surface defect (orange/yellow), weakly ordered surface (yellow/green). White particles correspond to $\bar{q}_6 < 0.2$.

Figure 7.1: Snapshots from representative simulation trajectories at $\text{Pe} = 58$ for the explicit-solvent (top) and implicit-solvent (bottom) models. Snapshots in column (a) are at $t = 37,500\tau$ when the film height is $H_x$ (see Figure 7.2), and subsequent snapshots (b–f) are shown every $25,000\tau$. Rendered using Visual Molecular Dynamics 1.9.2 [121] and colored according to each particle’s position in the diffusion map presented in Figure 7.4. The colors can be roughly mapped onto the following structures: HCP (red), FCC (dark blue), FCC (111) surface (light blue), FCC (100) surface (teal), surface defect (orange/yellow), weakly ordered surface (yellow/green). White particles correspond to $\bar{q}_6 < 0.2$.

The morphology of the bulk crystal was closely coupled with the structure at the liquid–vapor interface. For example, grain boundaries between face-centered cubic (FCC) and hexagonal close-packed (HCP) clusters in the first few layers often propagated to the supporting substrate, up to 15 layers away, in the form of stacking faults. Conversely, annealing of defects during later stages of drying sometimes led to buckling or cracking of the surface (Figure 7.1f). Such rearrangements considerably complicated prediction of the final morphology, and we found no clear correlations between the initial monolayer (Figure 7.1a) and final bulk (Figure 7.1f) morphology.

7.1.1 Evaporation rate

In order to assess how the evaporation rate affects crystallization kinetics, the position of the crystal interface was tracked using $\lambda_2 \psi_2$, the dominant collective variable obtained from the NGA diffusion map (Figure 7.4). This definition of the crystal...
Figure 7.2: (a) Solvent liquid–vapor interface position (dashed lines) and colloid solid–liquid interface position (solid lines) over the course of the simulations at different evaporation rates. Each line is the average of all trajectories with the standard deviation indicated by the shaded bands. $H_x$ is the film height at first bulk crystal appearance, $v$ is the solvent evaporation rate, and $v_x$ is the crystal front velocity. (b) Fraction of solidified particles with $\bar{q}_6 > 0.2$.

The solvent liquid–vapor interface (Figure 7.2a) correlated well with the overall fraction of particles that were considered solid-like (Figure 7.2b) using the $\bar{q}_6$ averaged local bond order parameter [243, 244], but was better at discriminating between bulk and surface structures near the substrate and liquid–vapor interface. The average $\lambda_2\psi_2$ near the interface trended from initially liquid-like toward crystalline values prior to our identification of the first bulk-like crystal layer, consistent with a nucleation event at the liquid–vapor interface. The time to form the first sufficiently thick layer for bulk-like growth was inversely proportional to the evaporation rate $v$. As a consequence, the first crystalline layer formed at roughly the same dried film height $H_x$, marked in Figure 7.2a, for all three evaporation rates.

In contrast, we found that the speed of the crystal front $v_x$ was less sensitive to the drying rate. For evaporation-driven growth, the crystal front speed was expected
to be proportional to $v$, with $v_x \approx v/(1 - \phi_0/\phi_x)$ according to a flux balance and $\phi_x$ being the local crystal volume fraction. Instead, $v_x$ varied by only a factor of two compared to a four-fold increase in $v$ (Figure 7.2a). In all cases, $v_x$ was greater than $v$. This suggests a two-step mechanism in which accumulation first leads to nucleation at the surface after reaching a sufficiently high density, then templated crystal growth from the surface follows. We propose that this crystal growth results from a combination of accumulation to the surface and additional thermodynamically favored crystallization due to the high initial volume fraction. We confirmed this by drying the crystal for a short time to form a surface nucleus and then stopping the evaporation, finding that the crystal continued to grow. This may be the origin of the unexpected annealing behavior reported by Cheng and Grest [207]. Unlike assembly onto carefully designed crystal templates [245], our simulations exhibited multiple surface structures forming together, resulting in polymorphic crystal growth.

Previous simulations suggested that there is an optimal evaporative flux to obtain high-quality crystals [207], which is taken to be a small fraction of a critical flux $j_c \approx 4D_0 \rho/d^4n_0$, with $n_0$ being the colloid number density. This expression for $j_c$ assumes that the most relevant colloid motion to obtain high-quality crystals is in-plane diffusion, which must be fast compared to accumulation to the interface in order to promote growth of large crystal grains. For our model, the slowest drying rate is $j \approx j_c/6$, close to the suggested optimum [207], while the fastest drying rate is $j \approx 2j_c/3$, which should form poorer quality surface structures. Indeed, when we compared surface structures early in the drying process, we found that typically fewer, larger grains were formed at slower evaporation rates, consistent with this model (Appendix 7.B.1). However, the early surface structuring underwent significant rearrangements at later drying times after a sufficient number of crystal layers formed below. Figure 7.3 compares the evolution of the surface structure for two independent simulations at Pe = 118. Initially, one surface is more disordered than
Figure 7.3: Snapshots of interface structure for two independent simulations at $\text{Pe} = 118$ for times (a) $25000\ \tau$, (b) $56250\ \tau$, and (c) $88750\ \tau$. Colors are the same as Figure 7.1. Snapshots of all interfaces are provided in Appendix 7.B.1.

the other (Figure 7.3a). As the drying progresses, both surfaces undergo rearrangements that increase the surface grain size (Figure 7.3b), and ultimately both become well-ordered single grains (Figure 7.3c). Due to such rearrangements, initial surface order did not correlate with overall crystal quality.

No statistically significant differences in the final crystal structures were seen from multiple independent simulations at the different drying rates. Essentially all characteristic morphologies could be obtained at all rates (see Appendix 7.B.1 for snapshots from all 35 simulations). The effects of the evaporation rate are well-described at early times by the Péclet number $\text{Pe}$, controlling the onset of crystallization and associated with diffusion normal to the interface, and the parameter $j/j_c$, affecting the number of grains that form at the surface and associated with in-plane diffusion. However, deformation of the interface at late stages of drying limits the usefulness of initially high-quality surface structures. This suggests a novel route for rapid processing of colloidal crystals. Starting from a sufficiently high colloid concentration, a period of fast drying is used to accumulate a crystalline layer at the interface. Templated growth from this surface is then driven by both the drying process and equilibrium thermodynamics. Finally, when the crystal front reaches the substrate...
7.1.2 Structural evolution

The local descriptors obtained by NGA shed additional insight into the balance of equilibrium and nonequilibrium effects in the crystallization mechanism. From initially liquid-like structures, particles proceeded towards an ordered state through one of two branches corresponding to HCP-like or FCC-like symmetry, beginning with weak hexagonal ordering and adopting increasingly higher coordination number until reaching a bulk crystal state. To visualize this progression, we divided each branch into four reference structures with increasing coordination, illustrated in Figure 7.4. We then recorded the fraction of solid-like particles closest to each reference for each simulation snapshot, shown in Figure 7.5. For the first part of the trajectory, the relevant timescale for comparing the drying rates is the film drying time $\tau_d = H_0/v$, which characterizes the formation of a crystalline surface layer. The evolution of structures prior to this event, particularly A and E, are indistinguishable for the three different evaporation rates when time is scaled by $\tau_d$. 

before all solvent is removed, as in our simulations, additional drying deforms the crystal and anneals out defects inherent to the fast growth process.
After the surface layer forms (white line in Figure 7.5), the transitions from surface structures (A, E) toward bulk structures (D, H) are better compared using $\tau_x = H_x/v_x$, the characteristic time of crystal growth. For example, the transition times between structures C and D are best collapsed between evaporation rates when the time after growth of the surface layer is scaled by $\tau_x$ instead of $\tau_d$. The inability of $\tau_d$ to correctly describe the time evolution of structures is even more apparent in the sample snapshots shown at the right of Figure 7.5 (see Appendix 7.B.3 for additional snapshots), which are taken at equal dried film heights (equal drying times). At this film height, the fastest dried film (Pe = 118) is only roughly 50% crystallized while the slowest dried film (Pe = 29) is mostly crystallized because it has effectively had more time for growth. We note that there are some dissimilarities in the structural evolution near the end of the trajectories (e.g., different fractions of structure H). These small discrepancies are largely due to interactions with the substrate and subsequent deformation of the crystal, which are not coupled to either timescale. This analysis provides strong indication that in the Pe > 1 regime, changes to the evaporation rate affect the onset of crystallization, but the progression of the crystal morphology is only similar when comparable growth times incorporating both evaporation-induced and equilibrium growth are considered.

### 7.1.3 Influence of solvent

Having assessed the crystal growth mechanism and the effects of the evaporation rate, we now consider the role of the solvent itself in controlling the crystallization. Solvent has both a structural effect on the colloids influencing the preferred particle spacing and a dynamical effect mediating hydrodynamic interactions. The typical approach employed in most prior studies [226, 227, 238], as in Chapters 4–6, is to treat the structural effects implicitly while also neglecting hydrodynamic interactions to expedite the simulations and model larger length and time scales. However, this
approximation remains untested for evaporation-induced crystallization, and there has been no direct comparison to assess how solvent-mediated hydrodynamic interactions influence crystal growth.

We constructed an implicit-solvent model for the colloids without hydrodynamic interactions using Langevin dynamics [160, 162] to match the long-time diffusion in the explicit-solvent model. To incorporate the structural effects of the solvent, we optimized a coarse-grained colloid pair potential (Figure 7.6) by minimizing the relative entropy between the explicit and implicit solvent models [247]. Relative entropy minimization ensures that this pair potential, which approximates the potential of mean force, quantitatively reproduces the colloid–colloid pair correlations [248]. We validated our model by computing the pair distribution function (Figure 7.7) and equilibrium diffusion coefficient (Figure 7.8) in the bulk over a range of concentrations, $0.05 \leq \phi_0 \leq 0.26$, and found excellent agreement between the
implicit- and explicit-solvent simulations. Drying simulations were then conducted using this implicit-solvent model.

![Effective pair potential between colloids for implicit-solvent model](image1)

Figure 7.6: Effective pair potential between colloids for implicit-solvent model.

![Radial distribution function](image2)

Figure 7.7: Radial distribution function $g(r)$ for explicit- (top) and implicit-solvent (bottom) models for $N$ colloids in a cubic box with edge length $100 \sigma$ at $T = \varepsilon/k_B$. Configurations for the explicit-solvent model were prepared by the procedure described in Appendix 7.A.

We started the implicit-solvent simulations from the same initial colloid configurations as in the explicit-solvent simulations, but found that the crystallization dynamics were significantly different between the two models (Figure 7.1) despite good
Figure 7.8: Diffusion coefficient $D$ for explicit- and implicit-solvent models for $N$ colloids in a cubic box with edge length $100\sigma$ at $T = \varepsilon/k_B$. Colloid configurations were sampled every $5\tau$ for $75000\tau$ when $N \geq 250$ and for $25000\tau$ when $N < 250$.

agreement for equilibrium properties. Particles accumulated at the liquid–vapor interface faster for the explicit-solvent model (Figure 7.1a), resulting in earlier onset of crystal growth (Figure 7.1b). Additionally, the explicit-solvent model crystallized exclusively down from the liquid–vapor interface, whereas the implicit-solvent model also crystallized up from the substrate (Figure 7.1c). This combined growth led to a more disordered crystal when the two growth fronts met in the middle (Figure 7.1d). We noted that the implicit-solvent model did not undergo surface buckling as early as the explicit-solvent model (Figure 7.1e), which might be expected from our approximation of the liquid–vapor interface as a flat plane. The final morphologies only appeared similar once both models underwent significant rearrangements during compression (Figure 7.1f and Appendix 7.B.4), consistent with the lack of correlation between initial and final structure in the explicit-solvent model.

Hydrodynamic interactions in a suspension influence the particle motion and hence distribution in the drying film. Colloid diffusion is driven by forces due to chemical potential gradients [1]. Brady has shown in the context of phoretic motion of a colloid that hydrodynamic interactions between particles reduce the velocities that result from these forces [205]. Assuming comparable dependence of the chemical potential on concentration in both models, slower particle migration is consistent with
a higher colloid density near the interface, resulting in earlier onset of crystallization. This picture is consistent with the drying polymer mixtures of Chapter 6.

There is also a temperature gradient in the explicit-solvent model due to evaporative cooling that is absent from the Langevin dynamics simulations. At pseudosteady state, the temperature profile within the drying film is predicted to be linear in distance from the substrate with a slope proportional to the evaporative flux, in good agreement with the simulations. We also found that the temperature gradient was larger for the colloid suspensions than for the neat solvent for Pe = 58. The temperature near the liquid–vapor interface was nearly 25% lower than at the substrate for Pe = 118. This large gradient may increase the driving force for crystallization near the liquid–vapor interface and contribute to faster growth from this surface in the explicit-solvent model.

7.2 Conclusions

We analyzed the mechanism of evaporation-induced colloidal crystallization by extensive nonequilibrium molecular dynamics simulations and characterized the crystal structures using a template-free machine-learning technique. In agreement with previous findings, faster evaporation rates led to earlier onset of crystallization with more disordered surface structures. Surprisingly, the final crystal structure did not depend significantly on the evaporation rate for the range of drying conditions investigated because the initial surface structures deformed to anneal defects in the bulk crystal. This finding suggests a key difference between crystals grown by sedimentation, where colloids accumulate onto a rigid substrate, and crystals grown by evaporation, where the interface can deform to relieve defects.

Detailed microscopic structural analysis revealed that the initial regime of the assembly process was well-described by the time scales of film drying and crystal
growth. Unexpectedly, the characteristic time scale for crystal growth was influenced by both the evaporation rate and thermodynamically favored crystallization at the studied initial colloid volume fraction. Our results suggest an unconventional route for rapid processing of colloidal crystals: after formation of a surface layer to initiate growth, a fundamentally nonequilibrium process, crystal growth toward the substrate can then be driven by equilibrium thermodynamic considerations if the colloid concentration is sufficiently large. Evaporation both drives crystal growth and anneals defects by collective rearrangements as the solvent is removed. The initial colloid volume fraction can then be exploited concurrently with the evaporation rate to manipulate the onset and relative rates of these equilibrium and nonequilibrium growth mechanisms.

We were able to resolve bulk-like crystal structures with many more crystal layers than previous simulations [207]. Nonetheless, much larger simulations would be required to model significantly lower initial concentrations; these are infeasible for the explicit-solvent model. While this challenge would typically be overcome through the use of an implicit-solvent model neglecting hydrodynamic interactions, we showed that such a model can yield stark differences in the crystal growth kinetics during evaporation despite good agreement for bulk equilibrium structural and dynamic properties. We conclude that implicit-solvent models lacking hydrodynamic interactions do not faithfully resolve the dynamics of nonequilibrium processes such as drying, in agreement with a recent theoretical analysis of drying colloidal mixtures [204] and Chapter 6. An appropriate mesoscale simulation method incorporating hydrodynamic interactions seems most promising for tackling such problems in the future.
7.A Models and Methods

7.A.1 Explicit solvent model

The explicit solvent model is based on that of Cheng and Grest [207, 241] and is described in the appendix of Ref. 242. The solvent was represented explicitly as a fluid of Lennard-Jones particles with the interaction potential

\[ U_{ss}(r) = 4\varepsilon \left( \frac{\sigma}{r} \right)^{12} - \left( \frac{\sigma}{r} \right)^6 , \]  

(7.1)

where \( r \) is the distance between particle centers. The potential was multiplied by a polynomial \( s(r) \) to smooth it from \( r_0 = 2.5\sigma \) to zero at \( r_c = 3\sigma \),

\[
s(r) = \begin{cases} 
1 & r \leq r_0 \\
\frac{(r_c^2 - r^2)^2(r_c^2 + 2r^2 - 3r_s^2)}{(r_c^2 - r_s^2)} & r_0 < r \leq r_c \\
0 & r > r_c 
\end{cases}, \]  

(7.2)

The density of the coexisting liquid and vapor were \( 0.66 m/\sigma^3 \) and \( 0.045 m/\sigma^3 \) at temperature \( T = \varepsilon/k_B \) with \( k_B \) being Boltzmann’s constant.

Spherical colloids with diameter \( d = 10\sigma \) were modeled as a uniform distribution of the same Lennard-Jones particles as the solvent [240] at density \( 1/\sigma^3 \) (mass \( 523.6 m \)) with the interaction potential

\[
U_{cc}(r) = -\frac{A_{cc}}{6} \left[ \frac{2a^2}{r^2 - 4a^2} + \frac{2a^2}{r^2} + \ln \left( \frac{r^2 - 4a^2}{r^2} \right) \right] + \frac{A_{cc}}{37800} \frac{\sigma^6}{r} \left[ \frac{r^2 - 14ra + 54a^2}{(r - 2a)^7} + \frac{r^2 + 14ra + 54a^2}{(r + 2a)^7} - \frac{2(r^2 - 30a^2)}{r^7} \right], \]  

(7.3)

where \( A_{cc} = 4\pi^2\varepsilon \) and \( a = d/2 \). The colloid interactions were made purely repulsive by truncating \( U_{cc} \) at its minimum, \( r = 10.581 \sigma \). Interactions between the solvent
and colloids were modeled by the integrated Lennard-Jones potential between a point and a sphere [240],

\[ U_{cs}(r) = \frac{2A_{cs}a^3\sigma^3}{9(a^2 - r^2)^3} \left[ 1 - \frac{(5a^6 + 45a^4r^2 + 63a^2r^4 + 15r^6)\sigma^6}{15(a - r)^6(a + r)^6} \right] , \quad (7.4) \]

with Hamaker constant \( A_{cs} = 100\varepsilon \). The potential was smoothed to zero using \( s(r) \) from \( r_0 = 8.5\sigma \) to \( r_c = 9\sigma \). No gravitational forces were applied to the colloids, but the attraction between the colloids and the solvent ensured that they remained well-dispersed in the liquid solvent [241].

The simulation box was periodic along the \( x \) and \( y \) dimensions with edge lengths \( L_x = 138.4\sigma \) and \( L_y = 159.9\sigma \). This size and aspect ratio is commensurate with the FCC (111) crystal face for our model. The suspension was supported from below by a flat substrate at \( z = 0\sigma \) that was attractive for the solvent and repulsive for the colloids. The solvent–substrate interactions were modeled by a Lennard-Jones 9–3 potential [241],

\[ U_{sw}(z) = 2\varepsilon \left[ \frac{2}{15} \left( \frac{\sigma}{z} \right)^9 - \left( \frac{\sigma}{z} \right)^3 \right] , \quad (7.5) \]

where \( z \) is the distance of the particle center normal to the surface with \( U_{sw} \) truncated at \( z = 3\sigma \). The solvent vapor was additionally confined at \( z = 325\sigma \) by a purely repulsive Lennard-Jones 9–3 potential truncated at \( (2/5)^{1/6}\sigma \). The colloid–substrate interactions were represented by the potential [241]

\[ U_{cw}(z) = -\frac{A_{cw}}{6} \left[ \frac{2az}{z^2 - a^2} + \ln \left( \frac{z - a}{z + a} \right) \right] + \frac{A_{cw}}{7560}\sigma^6 \left[ \frac{7a - z}{(z - a)^7} + \frac{7a + z}{(z + a)^7} \right] \quad (7.6) \]

with Hamaker constant \( A_{cw} = 144\varepsilon \) and cutoff at \( 5.5764\sigma \).

We dispersed 2052 colloids into the liquid solvent in coexistence with its vapor (2.3 million solvent particles in total), giving a liquid film height \( H_0 \approx 205\sigma \). Evaporation was modeled by deleting randomly selected solvent particles at a controlled rate.
from the vapor in the top 20\(\sigma\) of the box [207]. Solvent particles within 20\(\sigma\) of the substrate were weakly coupled to a Langevin thermostat [162] at \(T = \varepsilon/k_B\) with friction coefficient 0.1\(m/\tau\). We stopped the simulations before the crystal was completely dried (55% of solvent removed) to avoid spurious forces due to the neglect of gravity. All simulations were performed using HOOMD-blue (version 2.1.5) on multiple graphics processing units [122–124] with a time step of 0.005\(\tau\). We performed 5 independent simulations for Pe = 29 and 15 independent simulations for Pe = 58 and 118.

7.A.2 Implicit solvent model

The implicit-solvent colloid–colloid interactions were optimized using relative entropy minimization. A reference trajectory was prepared by dispersing 250 colloids into a cubic, periodic simulation box with edge length 100\(\sigma\) (\(\phi_0 = 0.13\)) filled with Lennard-Jones solvent at the liquid coexistence density, and removing any solvent particles within a distance of 5.5\(\sigma\) of any colloid. After equilibration, colloid configurations were collected every 5\(\tau\) during a 75000\(\tau\) simulation in the canonical ensemble. The effective colloid pair potential had a cutoff of 18\(\sigma\), which was commensurate with the first minimum of the colloid pair distribution function from the reference trajectory. The potential was modeled using cubic B-splines with a knot density of 2 knots per \(\sigma\), which ensured that the coarse-grained model was able to resolve all structural features in the pair distribution function. Knot values were then optimized to minimize the relative entropy with the reference trajectory using a combination of conjugate gradient and Hessian descent schemes [247]. Since the minimization proceeds by stochastically initiating short trial molecular dynamics trajectories (15000\(\tau\)), we verified that independent optimizations with different knot densities gave nearly identical potentials. In order to approximate the colloid dynamics in the solvent, Langevin dynamics simulations were employed with fric-
tion coefficient $70 m/\tau$ to match the diffusion coefficient measured from the average colloid mean-squared displacement in the reference trajectory. The drying liquid–vapor interface was represented by the purely repulsive part of a harmonic potential (Eq. 4.5) starting with its minimum at $H_0$ and moving down at fixed speed $v$, as in Chapters 4–6. The spring constant was chosen to be $5 \varepsilon/\sigma^2$ so that the colloids could fluctuate near the interface but all remained below it.

7.A.3 Crystal characterization

The crystalline character of particles was assessed throughout the simulations using our recently developed neighborhood graph analysis (NGA) method [242]. NGA uses the diffusion map dimensionality reduction technique [249–251] to identify collective variables that characterize the local neighborhood topology of each particle (Figure 7.4). We first used the $\bar{q}_6$ averaged local bond order parameter [243, 244] to filter solid-like particles ($\bar{q}_6 > 0.2$) for characterization by NGA. Particles were then assigned to structural classes based on the position of their neighborhood graph in the diffusion map compared to nine reference structures (eight of these are shown in Figure 7.4). These reference structures were chosen based on high observation frequency (C, G), the appearance of large contiguous clusters (A, B, E, F), or both (D, H). The diffusion map coordinates of each particle were projected onto lines between each pair of reference structures $(j, k)$ and the nearest such projection (located a distance $d_{proj}$ from the original point) was selected. The particle’s identity was split between the reference structures according to the fraction $(f_j, f_k)$ of the distance from the projection to each reference point. Each particle was further weighted by $w = \exp(-d_{proj}/0.05)$, which is related to the scaling of the Gaussian kernel in the diffusion map, in order to penalize structures that were not in the direct path between reference structures. Channel $k$ in Figure 7.5 then contains the total weight $W_k = \Sigma f_k w$ in each frame of the simulation, normalized by $\Sigma W_k$. Particles were
colored in the snapshots by assigning hues to each reference structure, interpolating between them, and desaturating according to the weight $w$ by mixing with gray.
7.B Additional Snapshots

7.B.1 Explicit solvent surface structure at equal crystal thickness

Figure 7.9: Snapshots from 5 simulations with explicit solvent and $\text{Pe} = 29$ with crystal thickness $40 \sigma$ (distance between crystal interfaces).
Figure 7.10: Snapshots from 15 simulations with explicit solvent and Pe = 58 with crystal thickness 40σ (distance between crystal interfaces).
Figure 7.11: Snapshots from 15 simulations with explicit solvent and Pe = 118 with crystal thickness $40\sigma$ (distance between crystal interfaces).
7.B.2 Explicit solvent fully crystallized

Figure 7.12: Snapshots from 5 simulations with explicit solvent and $\text{Pe} = 29$ at $t = 175000 \tau$, corresponding to a crystal front position of $z = 12.5 \sigma$. 
Figure 7.13: Snapshots from 15 simulations with explicit solvent and $\text{Pe} = 58$ at $t = 117500 \tau$, corresponding to a crystal front position of $z = 8.5 \sigma$. 
Figure 7.14: Snapshots from 15 simulations with explicit solvent and $\text{Pe} = 118$ at $t = 88\,750\,\tau$, corresponding to a crystal front position of $z = 5.5\,\sigma$. 

154
7.B.3 Explicit solvent at equal film height

Figure 7.15: Snapshots from 5 simulations with explicit solvent and Pe = 29 at $t = 175000 \tau$, corresponding to a film height $H = 171\sigma$. Note: identical to Figure 7.12, reproduced for comparison to Figure 7.16 and Figure 7.17.
Figure 7.16: Snapshots from 15 simulations with explicit solvent and $\text{Pe} = 58$ at $t = 87,500 \tau$, corresponding to a film height $H = 172 \sigma$. 
Figure 7.17: Snapshots from 15 simulations with explicit solvent and Pe = 118 at $t = 43750\tau$, corresponding to a film height $H = 171\sigma$. 
7.B.4 Explicit versus implicit solvent at equal film height

Figure 7.18: Snapshots from 15 simulations with explicit solvent and Pe = 58 at $t = 162500 \tau$, corresponding to column f of Fig. 2.
Figure 7.19: Snapshots from 15 simulations with implicit solvent and Pe = 58 at \( t = 162500 \tau \), corresponding to column f of Fig. 2.
Chapter 8

Concluding remarks

In this dissertation, massive-scale molecular simulations were applied to investigate the coupled structure and dynamics of colloid–polymer mixtures out of equilibrium. The primary objective was to connect microscopic properties of colloidal dispersions, such as colloid size or polymer molecular weight, to emerging macroscopic properties or processes like the assembly of large-scale ordered structures. Flow in microfluidic devices and evaporation of a film were considered as two case studies where structure and dynamics are intimately coupled and often led to unexpected behavior.

In Chapter 2, hybrid molecular dynamics simulations were performed to study the viscoelastic focusing of spherical colloids dispersed in a dilute polymer solution in slit-like microfluidic channels. The underlying Newtonian solvent and the ensuing hydrodynamic interactions were incorporated through the multiparticle collision dynamics method (see Appendix B for details of the algorithm), while the constituent polymers were modeled as bead–spring chains, maintaining a description consistent with the microscopic nature of the colloids and polymers. Colloids were found to migrate into the channel center for specific solute properties and flow conditions. The effective forces exerted on the colloids by the solvent and polymers were systematically measured to demonstrate how the migration originated from a competition
between viscoelastic forces from the polymer solution and hydrodynamically-induced inertial lift forces. However, significant fluctuations in the colloid positions were obtained compared to the distributions predicted by the measured effective forces. This finding suggested that microscopic structural and dynamic coupling with the polymers plays an important role in viscoelastic focusing at the colloidal scale. In these cases, the polymer solution cannot be treated simply as a viscoelastic background medium when designing flow-focusing devices.

Chapter 3 then considered the distribution of colloids in the axial (flow) direction of a microfluidic device. A complete theoretical framework was developed to predict the dispersion of a colloidal suspension confined in a parallel plate channel, extending the Taylor–Aris treatment [48–50] to particles with diameters comparable to the channel width. The theoretical model incorporated the effects of confinement on the colloid distribution, corrections to the velocity profile due to the effects of colloid concentration on the suspension viscosity, and position-dependent diffusivities. The theoretical model was tested using explicit-solvent molecular dynamics simulations that fully incorporated hydrodynamic correlations and thermal fluctuations. Good quantitative agreement was obtained between the theory and simulations. The non-uniform colloid distributions that arise in confinement due to excluded volume between the colloids and channel walls significantly impacted the axial dispersion. It was shown that this microscopic structural effect must be included to reliably predict the macroscopic axial dispersion.

In Chapter 4, the focus was shifted to structure formation in evaporating films. First, the vertical segregation (stratification) of colloid mixtures was investigated using implicit-solvent molecular dynamics simulations. For large particle size ratios and film Péclet numbers greater than unity, the smaller colloids migrated to the top of the film, while big colloids were pushed to the bottom, creating an “inverted” stratification consistent with recent experiments [51–53]. To rationalize this behavior,
particle size ratios and drying rates spanning qualitatively different Péclet number regimes were systematically studied, and the dynamics of the inverted stratification were quantified in detail. The stratified layer of small colloids was found to grow faster and to larger thicknesses for larger size ratios. Interestingly, inverted stratification was observed even at moderate drying rates where the film Péclet numbers were comparable to unity, but the thickness of the stratified layer decreased. A model based on dynamical density functional theory was proposed to explain the observed phenomena, which attributed the effect to an enhanced downward migration speed of the large colloids compared to the small colloids. This enhancement was induced by the larger chemical potential gradients (forces) on the larger colloids that were not compensated by larger effective friction coefficients. This force can be understood microscopically as resulting from the larger excluded volume of a bigger colloid in a dense layer of small colloids, showing how large-scale stratified layers form due to the microscopic dynamics and interactions.

Chapter 5 extended the principles of Chapter 4 to drying polymer and colloid–polymer mixtures. The polymer mixtures stratified into layers analogous to the colloid mixtures, with the shorter polymers enriched near the drying interface and the longer polymers pushed down toward the substrate. The colloid–polymer mixtures stratified into a polymer-on-top structure when the polymer radius of gyration was comparable to or smaller than the colloid diameter, and a colloid-on-top structure otherwise. Dynamical density functional theory gave excellent quantitative agreement with the simulations for the polymer mixtures and qualitatively predicted the observed polymer-on-top or colloid-on-top structures for the colloid–polymer mixtures. Chapters 4 and 5 demonstrated that stratification, a nonequilibrium separation, depends sensitively on the microscopic structural and dynamic properties of the solute.
Hydrodynamic interactions were neglected from the models in Chapter 4 and Chapter 5 due to computational restrictions. Chapter 6 employed large-scale nonequilibrium molecular dynamics simulations to investigate the influence of hydrodynamic interactions on stratification, using a polymer mixture as a test case. In agreement with Chapter 5, short polymers stratified on top of long polymers when hydrodynamic interactions between polymers were neglected. However, no stratification occurred at the same drying conditions when hydrodynamic interactions were incorporated through an explicit solvent model. This analysis demonstrated that models lacking hydrodynamic interactions do not quantitatively predict stratification in drying mixtures, highlighting the importance of dynamics in stratification.

Finally, the influence of solvent evaporation on the assembly of colloidal crystals was studied in Chapter 7 using massive-scale nonequilibrium molecular dynamics simulations with unprecedented microscopic structural resolution. In agreement with previous experiments and simulations, faster evaporation rates led to earlier onset of crystallization and more disordered surface crystal structures. Surprisingly, it was found that collective rearrangements of the bulk crystal during later stages of drying reduced the influence of this initial surface structure, and the final morphology was essentially independent of the evaporation rate. The structural analysis revealed that the crystallization process was well-described by two time scales, the film drying time and the crystal growth time, with the latter having an unexpected dependence on the evaporation rate due to equilibrium thermodynamic effects at high colloid concentrations. These two time scales may be leveraged to control the relative influence of equilibrium and nonequilibrium growth mechanisms, suggesting a route to rapidly process colloidal crystals while also annealing defects. Our analysis additionally revealed that, as for the stratifying films of Chapter 6, solvent-mediated interactions also play a critical role in the crystallization kinetics during evapora-
tion, from which it was concluded that commonly used implicit-solvent models do not faithfully resolve nonequilibrium processes such as drying.

This dissertation highlights the important coupling of structure and dynamics in colloidal dispersions, both at the microscopic level and across much larger length scales. This fundamental feature of colloidal dispersions naturally extends to many related problems. Chapters 2 and 3 explored the behavior of rigid colloids in microfluidic channels, but the migration of soft colloids, including polymers, cells, and multiphase fluids, is also of interest for applications such as the collection of rare cell markers for HIV or cancer from the blood stream [252, 253]. Chapters 4–7 demonstrated how multilayer coatings or colloidal crystals can be formed by controlling the colloid size or concentration and the processing conditions. Hydrodynamic interactions were shown to play an important role that must be considered in order to engineer processes to create films or crystals by this method. Additional enthalpic interactions between components and competing thermodynamic phase transitions were only partially considered here and may present new opportunities and challenges for engineering evaporation-induced structures.

Many of the models and simulations in this dissertation required careful use of extensive computational resources. Appendix A presented an algorithm inspired by graphics processing to accelerate molecular dynamics simulations of colloidal mixtures, which was heavily utilized in Chapters 3–7. Appendix B described a massively-parallel implementation of the multiparticle collision dynamics algorithm. Such mesoscale simulation methods are promising candidates to address many of the previously described questions that may be challenging or infeasible to explore by other methods. In future, development of open-source scientific software must continue to enable the community to efficiently tackle these challenges with state-of-the-art tools.
Appendix A

Efficient neighbor list calculation for molecular simulation of colloidal systems using graphics processing units

There has been a steady growth of computational resources available to the scientific community [254]. The fastest supercomputers today offer petascale performance through hundreds of thousands of CPU cores with dedicated coprocessors or graphics processing units (GPUs) as accelerators. Whereas in the past most atomistic molecular simulations were restricted to no more than a few hundred particles over nanosecond time scales, modern computing architectures and simulation techniques have enabled simulations of millions of particles [255] and up to millisecond [256] time scales. In particular, molecular dynamics (MD) methods have emerged as powerful tools for large-scale molecular simulations for two important reasons: (1) the

MD algorithm is highly parallel and so is easily adapted for supercomputing, and (2) many highly optimized and flexible simulation packages are readily available to researchers. In the past two decades, significant time and resources have been devoted to the development of such MD packages, including GROMACS [257], LAMMPS [258], and NAMD [259], among many other commercial and open-source options. These packages provide robust MD implementations for massively parallel computers. A more recent addition is HOOMD-blue [122, 123], which was developed and optimized for GPUs, and has a single GPU performance an order of magnitude faster than a single CPU.

Despite these advances in hardware and software, MD simulations of soft matter remain challenging because there is typically a large disparity in length and time scales between components. For example, colloidal particles (nanometers to micrometers in diameter) in solution are separated in size by several orders of magnitude from an atomistic description of the molecular solvent. MD simulations retaining full atomistic detail of the solvent can become intractable because many solvent atoms must be included to model only a few colloidal particles. Moreover, the time scales associated with the degrees of freedom of the solvent are generally much shorter than the relatively slow motion of the larger colloids. This means that these simulations require very short MD time steps to faithfully capture the dynamics of the solvent, and many such steps are required to observe any appreciable dynamics of the colloids.

The MD algorithm in its simplest form consists of two basic steps: (1) calculation of the forces on all particles and (2) integration of Newton’s equations of motion. The force calculation is by far the most computationally expensive part of the MD algorithm. In particular, the calculation of nonbonded pair interactions between particles typically dominates the force calculation. In the simplest MD implementation, the forces between all possible pairs of the $N$ total particles in the simulation are evaluated, leading to $O(N^2)$ scaling.
To reduce the number of force pairs evaluated, the interaction potential between particles of types \(i\) and \(j\) is typically truncated at a radial cutoff distance \(r_{ij}\) where the force has decayed sufficiently so that truncation does not significantly influence the properties of interest. A neighbor (Verlet) list storing a list of particles that are within the cutoff is then created for each particle [260]. The pair forces only need to be computed for the particles in the neighbor list, which is a small subset of \(N\) for each particle. The neighbor list can be rebuilt less frequently than every MD step if a small buffer width is added to \(r_{ij}\), trading wasted pair force distance checks with the frequency of rebuilding the neighbor list, which accelerates the calculation compared to evaluating all possible force pairs at every step. However, the force calculation is ultimately still \(O(N^2)\) if the neighbor list is built by simply checking the distances between all particle pairs.

Acceleration structures reduce the computational cost of building the neighbor list by restricting the neighbor search for each particle to a subset of the particles in the system. The most commonly employed acceleration structure in general-purpose MD codes is the cell list. A typical cell list spatially bins particles into uniformly sized cells in \(O(N)\) [260]. Distance checks must only be performed for particles that are in neighboring cells, effectively reducing the cost of computing the neighbor list to \(O(Nm)\), where \(m\) is the average number of particles in a cell (usually \(m \ll N\)). The cell width is typically determined by the largest cutoff radius between all pairs so that 27 cells must be checked for each particle in three-dimensional simulations.

The cell list is extremely efficient in simulations that have nearly equal pair force cutoffs and a uniform particle distribution between the cells. However, performance degrades significantly in colloidal systems due to the large disparity in interaction lengths. Many unnecessary distance checks are performed for particles with short interaction ranges when the cell width is based on the largest cutoff, schematically illustrated in Figure A.1. The cell width is based on the largest cutoff \(r_{BB}\). The
solvent particles with cutoff \( r_{AA} \) must check the same number of cells (particles) as the colloids with the larger cutoff \( r_{BB} \). However, unlike the colloids, the solvent particles reject many of these particles from their neighbor lists.

Figure A.1: Cell list needed to determine the neighbors of solvent (A) and colloid (B) particles when the bin size is based on the largest cutoff length \( r_{BB} \). The pair interaction ranges are illustrated for each particle type. The shaded areas indicate the cells that each particle must search.

A general solution to this problem has been successfully deployed in LAMMPS [261]. The standard cell list is extended so that the cell width is based on the shortest cutoff and each particle type searches a different “stencil” of adjacent cells based on the largest cutoff radius. Distances to each cell in the stencil are precomputed so that a distance check can be skipped for many of the searched particles. This stenciled cell list method was reported to give speedups of nearly 100x for a colloidal solution with a 20:1 ratio in diameter compared to a standard cell list in LAMMPS. However, simulations of colloidal systems with such large size disparity are still extremely computationally intensive, requiring hundreds of CPU cores to obtain reasonable performance [261].

In this chapter, we explore two parallel algorithms for efficiently building neighbor lists in colloidal systems on the GPU: one based on stenciled cell lists and one based on a hierarchical tree acceleration structure. To our knowledge, the stenciled cell list algorithm [261] has not been previously implemented and tested on the GPU. In graphics processing, hierarchical tree data structures are used for performance-
critical neighbor searches [262]. One such tree structure, the bounding volume hierarchy (BVH), has previously been used to generate neighbor lists between large macromolecules on the CPU [263, 264]. However, these prior studies did not extend their approach to general-purpose molecular simulations, did not address size disparity between different particle types, and did not discuss implementation of the algorithm on GPUs.

In Section A.1, we present and compare parallel algorithms for the stenciled cell list and BVH. Technical details of the implementation of these algorithms within the HOOMD-blue simulation package are described in Section A.2. Systematic performance benchmarks for the algorithms are reported in Section A.3.

A.1 Algorithms

A.1.1 Stenciled cell list

The stenciled cell list [261] is a straightforward extension of the standard cell list, and is illustrated in Figure A.2. All particles are binned into a cell list of nominal cell width $\Delta_{\text{bin}}$. The maximum cutoff radius is determined for each particle type, and a stencil is computed from the list of offsets to neighboring cells that have a nearest separation distance within that cutoff. For example, the solvent particle has a stencil radius corresponding to $r_{AB}$, and the list of offsets in 2D is $(0,0), (+1,+2), (+2,-1), \ldots$. The colloid has a stencil radius $r_{BB}$. All the cells included in the stencils are shaded and outlined with a solid line.

Because the stencil size is set by the maximum cutoff radius per type, many particles that will not be included in the neighbor list must still be iterated over for shorter $r_{ij}$. In Figure A.2, both the solvent particle and colloid have the same effective stencil size for the given cutoffs. However, extra distance checks can be eliminated by precomputing the minimum distance to each cell in the stencil. For a
Figure A.2: Schematic illustration of a stenciled cell list with $\Delta_{\text{bin}} = r_{\text{AA}}$. Solid outlines of shaded areas indicate cells included in the stencil. The area marked by the dashed line around the solvent particle marks the cells in that stencil that are inside $r_{\text{AA}}$.

particle of a given type, if the minimum distance to the nearest cell is greater than the pairwise cutoff, that particle can be skipped without distance checking or reading its position. The solvent particle only needs to distance check particles of type A inside the cells indicated by the dashed line corresponding to $r_{\text{AA}}$, and all particles of type A can be skipped in the other cells in the stencil.

The neighbor list is then built as described in Algorithm A.1. A given particle looks up the appropriate stencil of cells based on its particle type (line 4). Each member of the stencil is iterated over (line 5), and each offset from the stencil is converted into a neighbor cell based on the current cell of the particle, including wrapping through the periodic boundaries (line 6). Particle $i$ then iterates through all potential neighboring particles in the cell. Initially, only the type of neighbor particle $j$ and its actual cutoff (for example, $r_{\text{AA}}$ or $r_{\text{AB}}$) are read into memory (line 9). If the minimum distance to the nearest cell is greater than the cutoff, particle $j$ can be skipped without a distance check (line 10). Otherwise, a distance check is performed, and the particle is saved if it is a neighbor (lines 11–13).

Although reducing the cell width can significantly reduce the number of distance checks performed, there is an associated penalty due to the additional data that is accessed for each cell. As the cell width shrinks, more cells must be accessed, and each
Algorithm A.1 Stenciled cell neighbor list

1: for each particle $0 \leq i < N$ in parallel
2: $x_i \leftarrow$ particle position
3: $c_i \leftarrow$ particle cell
4: $S \leftarrow$ stencil for type of $i$
5: for each $s$ in $S$ do
6: $c \leftarrow \text{WRAP}(c_i + s)$
7: $d \leftarrow$ minimum distance to $c$
8: for each particle $j$ in $c$ do
9: $r_{ij} \leftarrow$ cutoff between types of $i$ and $j$
10: if $d > r_{ij}$ then continue
11: $x_j \leftarrow$ position of $j$
12: if $|\text{WRAP}(x_j - x_i)| \leq r_{ij}$ then
13: AddNeighbor($j$)
14: end for each
15: end for each

The number of particles in these cells must still be read regardless of occupancy. In the original CPU implementation in LAMMPS [261], the optimal $\Delta_{\text{bin}}$ was found to be half the minimum cutoff distance, or $\Delta_{\text{bin}} = r_{AA}/2$. It is unclear if this will be the optimal value for GPUs, and we will discuss the effect of $\Delta_{\text{bin}}$ in Section A.3.

A.1.2 Bounding volume hierarchy

Although the stenciled cell list significantly improves upon the standard cell list by skipping many distance checks, it still requires iteration over many particles that are ultimately unnecessary because all particles are read out of a single cell list. One possible way to alleviate this would be to construct one cell list per type, and to construct different stencils for each particle type on each cell list so that only those particles that need to be checked are actually included. However, there are two complications to doing this. First, the construction of one cell list per type increases memory demands, since memory must be allocated for each cell list that spatially covers the entire simulation box. Perhaps more significantly, separating cell lists by
type will lead to lower occupancy of cells, which increases the overhead associated with searching the cell lists. This is especially problematic if a certain particle type is dilute, as is often the case for colloidal systems, or if there are many types of particles.

To address these issues in a general way, we can draw inspiration from graphics rendering where an analogous neighbor search problem occurs. Realistic rendering of scenes on the computer requires the accurate projection of light. One way to do this is to cast and trace light rays from a source onto a scene, detect the objects that the rays reflect from, and appropriately render the reflections. Acceleration structures are used to efficiently determine the objects that the rays reflect from. For scenes with non-uniform object density, tree data structures such as $k$-d trees, octrees, and bounding volume hierarchies are generally favored over cell-based structures. In this chapter, we have chosen to focus on bounding volume hierarchies (BVHs). BVHs have been demonstrated as useful accelerators for ray tracing collision detection in real-time rendering engines for computer gaming [262], in part because they can be reconstructed very quickly for scenes undergoing dynamic changes. Although BVHs do not always give the fastest traversal performance compared to other types of trees [265, 266], their total performance, including both construction and traversal time, is very competitive.

BVHs partition a system based on objects rather than space. An object or multiple nearby objects are enclosed to form a leaf node. Leaf nodes are then enclosed by larger bounding ("parent") internal nodes. In an axis-aligned bounding box (AABB) BVH, the volume of the nodes is chosen so that all "child" objects are enclosed in an orthorhombic box aligned to the Cartesian axes, illustrated in the left panel of Figure A.3. Together, the nodes form a tree hierarchy that can be traversed using a binary search algorithm that tests for volume overlap between a query AABB and the AABBs of the tree.
Figure A.3: Schematic illustration of a bounding volume hierarchy. *left:* Two particles are enclosed into each leaf node, outlined in green AABBs and labeled from 0 to 3. The internal nodes are labeled A to C. A query AABB is shown as a dashed box. *right:* Hierarchical representation of the tree nodes.

The BVH neighbor list algorithm is outlined in Algorithm A.2. One BVH is first built per particle type, and each particle is considered a point object. The height of the BVH (and so traversal time per particle) is $O(\log N)$, so multiple points that are nearby in space are merged into a leaf node for more efficient traversal. Each particle queries each tree (line 3) with a cutoff length specified per type (line 4). Periodic boundary conditions are implemented by translating the query AABB by the appropriate combinations of the box dimensions, leading to a maximum of 27 queries per tree in three dimensions (lines 5–6). However, many of these queries are trivially rejected when the AABB is translated to a position where it does not overlap the simulation box.

A binary search is performed on each tree using the query AABB (lines 8–20). The search is illustrated in the right panel of Figure A.3 for the query AABB from the left panel. The query AABB overlaps the root node A (line 9), and so proceeds to test against internal node B (line 11). It intersects with B, and so tests against leaf node 0, which it overlaps. A direct distance check is performed against objects contained within leaf node 0 for inclusion as neighbors (lines 13–16). Note here that no periodic wrapping of the distance between particles is required because all periodic images of $i$ are considered. Traversal then proceeds to the remaining untested nodes (line 17). The query AABB does not overlap 1, so it does not need to distance check
Algorithm A.2 BVH neighbor list

1: for each particle $0 \leq i < N$ in parallel
2: \hspace{1em} $x_i$ ← particle position
3: \hspace{1em} for each BVH tree $t$ do
4: \hspace{2em} $r_{ij}$ ← cutoff between types of $i$ and $t$
5: \hspace{2em} for each image $v$ do
6: \hspace{3em} $a$ ← AABB($x_i + v$, $r_{ij}$)
7: \hspace{3em} $n$ ← root node of $t$
8: \hspace{2em} while untested nodes remain do
9: \hspace{3em} if $a$ overlaps $n$ then
10: \hspace{4em} if $n$ is not a leaf node then
11: \hspace{5em} $n$ ← left child of $n$
12: \hspace{4em} else
13: \hspace{5em} for each particle $j$ in $n$ do
14: \hspace{6em} $x_j$ ← position of $j$
15: \hspace{6em} if $|x_j - x_i| \leq r_{ij}$ then
16: \hspace{7em} AddNeighbor($j$)
17: \hspace{5em} end for each
18: \hspace{4em} else
19: \hspace{5em} $n$ ← next node to test
20: \hspace{3em} else
21: \hspace{4em} Mark branch inactive
22: \hspace{3em} $n$ ← next node to test

those objects, and proceeds to test against $C$ (lines 19–20). It also does not overlap $C$, so none of the nodes in that branch need to be checked either. Traversal of the tree for this image is then complete, and the neighbor search returns to line 5.

Although traversing the BVH is a relatively straightforward parallel algorithm, efficiently generating the BVH in parallel is much more difficult. Several parallel algorithms have been proposed to build BVHs, differing mainly in the quality of the tree that is constructed and the speed of the algorithm. Lauterbach et al. introduced one parallel method to construct “linear” BVHs (LBVHs) by ordering objects along a Z-order curve, which groups spatially close objects near each other in the tree [267]. The sorted objects can then be processed in parallel to determine the hierarchy of the tree. This algorithm has been further refined [268, 269] to build “hierarchical” LBVHs that provide better construction speeds and lower memory footprints.
These algorithms process the tree from the top down (starting from the tree root); typically one level of the tree is processed in parallel at a time. However, this significantly restricts parallelism at top levels of the tree because these levels contain very few nodes [267]. Karras developed an alternative LBVH construction algorithm that proceeds from the bottom up and fully exploits the parallelism of the GPU by processing all internal nodes concurrently [270]. The quality of these LBVHs have been refined for higher ray casting performance [271]. The additional cost of refining the LBVH required casting $O(10^7)$ rays to see a significant speedup in the overall performance. However, typical molecular dynamics simulations contain $O(10^5)$ particles, and so a full neighbor list build requires only $O(10^6)$ traversals. Accordingly, we have chosen to implement the unrefined LBVH algorithm [270].

A.2 Implementation

Here we describe the implementation of the algorithms described in Section A.1 in the HOOMD-blue simulation package [122, 123], designed to run on NVIDIA GPUs using the CUDA programming language. In particular, we target modern NVIDIA GPU architectures (“Kepler” and newer) for optimization of our design. The code for both algorithms is available starting in HOOMD-blue version 1.3.0.

A.2.1 Stenciled cell list

The stenciled cell list algorithm consists of two steps: building the cell list and stencils, followed by neighbor list generation from the cell list. A cell list is constructed on the GPU every time the neighbor list is built using the existing methods in HOOMD-blue. The cell list stencil is computed on the CPU. Because it depends only on the maximum cutoff radius of each particle and the size of the cells, the cell stencil only needs to be recomputed when one of these changes. Each member of the
stencil is saved in a four element structure that holds the $x$, $y$, and $z$ offsets from a reference cell and the minimum distance to that cell. Care is taken during the stencil construction so that no cells are “double counted” due to periodic boundaries when the stencil covers the entire simulation box.

In HOOMD-blue, the standard cell list is processed to build the neighbor list using multiple threads per particle to increase parallelism [123]. The stenciled cell list similarly benefits from this optimization, and so Algorithm A.1 is slightly modified in our implementation. We assign $n$ threads per particle. Each thread follows Algorithm A.1 in a strided manner so that it only processes every $n$-th particle read from the cell list, effectively parallelizing lines 8–13 of Algorithm A.1. The optimal value of $n$ is automatically tuned at run time because the performance gained by using multiple threads per particle varies depending on the number of particles in the system, the number of particles per cell, and the GPU architecture. We restrict $n$ to be a power of 2 smaller than the CUDA warp size so that the full neighbor list can be aggregated efficiently using intra-warp stream compaction. The original Algorithm A.1 is recovered when $n = 1$.

Significant divergence can occur in the inner loop of Algorithm A.1 because randomly ordered particle types can have very different stencil sizes to search or can search different areas of space. In HOOMD-blue, all particle data is periodically sorted along a space-filling curve to improve spatial locality of the data. We then additionally sort the particle indexes by type before the neighbor list is calculated. By applying a stable sorting algorithm, we still preserve most of the benefits of the space-filling curve sort. During the neighbor list build, threads operate on the particle data in this sorted order.
A.2.2 Building LBVHs

We build LBVHs using the highly parallel algorithm of Karras [270], which is outlined in Algorithm A.3 for a single particle type. All particle indexes are first sorted along a Z-order curve using 30-bit Morton codes (lines 1–4) [272]. The Morton codes are generated by placing each particle into one of \(2^{10}\) bins along each coordinate axis in the simulation box (line 2) and interleaving the three 10-bit integers corresponding to these bins (line 3). In order to construct one LBVH per particle type, we also prepend the bit string representing the particle types to the Morton codes to effectively sort first by particle type and then along the Z-order curve. Figure A.4a shows a schematic of this process, where 4-bit Morton codes are assigned in two dimensions for two particle types. Reading in lexicographic order, the first bit corresponds to the particle type, while the next 4 bits are the Morton code representation of the bin. The type-Morton codes are efficiently sorted using a parallel radix sort (line 4). Because radix sort is \(O(Nk)\) where \(k\) is the number of bits to compare in the key, we restrict the sort to the lowest \(30 + b\) bits where \(b\) is the number of bits necessary to represent the largest type index. Figure A.4b shows the particles in sorted order.

The number of leaf nodes in each tree is computed assuming a constant number \(w\) particles per leaf filled in order along the Z-order curve (line 5). Using one thread per
Algorithm A.3 LBVH build

1: for each particle 0 ≤ i < N in parallel
2: \( (x, y, z) \leftarrow \) 3D bin of \( i \)
3: \( m_0(i) \leftarrow \text{MortonCode}(x, y, z) \)
4: SortMortonCodes() in parallel
5: \( N_{\text{leaf}} \leftarrow \lceil N/w \rceil \)
6: for each leaf node 0 ≤ i < \( N_{\text{leaf}} \) in parallel
7: \( \{p\} \leftarrow \) set of \( w \) particles in \( i \)
8: \( a \leftarrow \text{MERGE}(\{p\}) \)
9: \( m(i) \leftarrow m_0(p_0) \)
10: GenerateHierarchy(\( m \)) in parallel
11: for each leaf node 0 ≤ i < \( N_{\text{leaf}} \) in parallel
12: \( n \leftarrow i \)
13: do
14: \( p \leftarrow \) parent of \( n \)
15: \( s \leftarrow \) sibling of \( n \)
16: atomic
17: \( v \leftarrow \) times \( p \) has been visited
18: \( v \leftarrow v + 1 \)
19: end atomic
20: if \( v = 0 \) then return
21: \( p \leftarrow \text{MERGE}(n, s) \)
22: \( n \leftarrow p \)
23: while

leaf, we merge successive sets of \( w \) particles into an AABB enclosing all the particles, and take the Morton code of the first particle as an approximation of the Morton code for that leaf node (lines 7–9). Schematically, this is indicated by the indexes and boxes shown in Figure A.4b. Although this is not necessarily the optimal way to group particles into leaves or to represent the spatial position of the merged AABB, it is inexpensive and guarantees that the Morton codes remain in lexicographic order without an additional sorting call. We found that employing \( w = 4 \) particles per leaf gave the best traversal times in our benchmarks (see Section A.3 for benchmark details).

The tree hierarchy, defining the parent-child edges between the nodes, is then processed using one thread per internal node (line 10). In essence, each thread
processes the Morton codes to determine a “split position” in the leaf nodes for the current internal node. This split identifies both the indexes of the children of the internal node as well as whether those children are leaf nodes or internal nodes. The reader is referred to Ref. 270 for a detailed presentation of this algorithm.

Finally, the AABBs of the internal nodes are determined by processing up from the leaf nodes using one thread per leaf (lines 11–21). Each internal node is processed by the second thread to arrive at the node, with the arrival order determined by an atomic counter (lines 16–19), which guarantees that the parent internal node is processed only after its children have been processed. The AABB of each internal node is determined by merging the AABB of the active thread with the AABB of the active thread’s sibling, so that the AABB of the parent node encloses both child AABBs (line 20). The active thread then iterates to the next level in the BVH (line 21).

A.2.3 Traversing LBVHs

Once the LBVH is built, traversing it is a trivially parallel process, since each particle may independently query the tree and record its neighbor list. The most challenging part of Algorithm A.2 is to determine the next nodes to traverse or skip. One way to perform this traversal is to explicitly manage a stack of node pointers to test per thread. However, this can lead to large memory demands per thread, which can be undesirable if it reduces the device occupancy. We instead implemented a stackless traversal method based on “ropes” [273–275], in which each AABB stores the indices of its left child and a “rope” that is the index of the node that traversal should proceed to if the current node is a leaf node or it is not intersected (lines 17 and 20 in Algorithm A.2). In practice, we generate the skip ropes during the hierarchy generation and AABB determination.
The ropes are shown schematically in Figure A.5, where green arrows indicate ropes to other nodes within the tree while red arrows indicate that traversal should be terminated. If the query AABB overlaps an internal node, the traversal proceeds to the left child. If it does not overlap, then traversal proceeds to the next node along the rope. The rope is always followed for a leaf node regardless of overlap. For example, if the query AABB overlaps internal node 0, traversal proceeds to internal node 1. If it does not overlap internal node 1, then leaf nodes 0 and 1 can be skipped, and traversal proceeds along the rope to leaf node 2. After leaf node 2 is processed, traversal of this tree is complete.

Figure A.5: Schematic LBVH tree constructed on GPU, including bubbled AABBs and skip ropes. Internal nodes are shaded in blue, leaf nodes are in black. A green arrow indicates a skip rope, while a red arrow indicates termination of traversal.

We found that memory access patterns significantly affected the traversal speed. The AABB data was aligned in two four element structures containing the lower and upper bound coordinates of the AABB, the traversal rope, and either the left child for an internal node or the number of particles in a leaf node. With this layout, our implemented rope algorithm is advantageous compared to other proposed stackless traversal algorithms [276–278] because its memory accesses are minimal and fully coalesced. Similarly to the stenciled cell list, the neighbor list is constructed from the sorted particle order so that threads within a warp traverse similar parts of the tree. We have not implemented the LBVH traversal using multiple threads per particle because Algorithm A.2 does not lend itself as naturally to strided access as Algorithm A.1. Such an optimization is left as possible future work.
A.2.4 Memory management

The neighbor list memory layout affects the speed of the subsequent pair force calculation. In previous versions of HOOMD-blue, the neighbor list was stored in an effectively two-dimensional row-major matrix with the particle indexes as columns and the neighbors as rows [122]. This memory layout demands that the number of rows be equal to the maximum number of neighbors for any particle. Although this works well when all particles have roughly the same number of neighbors, it presents problems in size-asymmetric systems when a minority of particles has many more neighbors than the majority. The number of neighbors for a particle scales as \( r_{ij}^3 \). In a system with a cutoff radius disparity of 10:1, the large particles have roughly 1000 times more neighbors than the small particles. This in turn leads to intense memory demands that can easily exceed the typical capacities of GPUs, even if there is only a single large particle.

To remedy this problem, we manage the neighbor list memory in an alternative fashion that significantly reduces memory requirements. We track the maximum number of neighbors per particle type, and allocate the memory for a one-dimensional array holding the maximum total number of neighbors for the system based on the number of particles per type. We round the maximum number of neighbors per type to the next-highest multiple of 8 for improved memory alignment and less frequent resizing. Because particles are randomly ordered in memory by type, we maintain a second array that gives each particle an index into its memory in the one-dimensional neighbor list array. We compute this list of indexes in parallel using a device-wide exclusive prefix sum on an array initially filled with the number of neighbors reserved for each particle.
A.3 Performance

All benchmarks were performed on XSEDE [279] Comet, hosted by the San Diego Supercomputer Center, on an NVIDIA Tesla K80 GPU. HOOMD-blue was compiled using CUDA 7.0 in single precision without MPI support because this is a typical configuration for many users. The open-source CUB library [280] (version 1.4.1) is embedded in HOOMD-blue for the radix sort.

A.3.1 Lennard-Jones binary mixture

We first performed a synthetic benchmark of a Lennard-Jones binary mixture of particles that differ solely in their cutoff radius. Because the Lennard-Jones potential decays quickly with radial distance, the local density of the bulk liquid is essentially unchanged by expanding the cutoff beyond a sufficiently long length. This means that varying the cutoff systematically varies the number of neighbors to compute while the particle configurations are mostly unperturbed. This benchmark can then systematically probe the performance of the neighbor list algorithms as a function of both size disparity and composition.

We first equilibrated a Lennard-Jones liquid of 192,000 particles (all marked type “A”) at $T = 1.5$ in a cubic box of edge length $L = 72$ with all quantities defined in the standard reduced Lennard-Jones units. We used a cutoff of $r_{AA} = 3.0$ between all particles during the equilibration. Integration was performed with particles coupled to a Langevin heat bath using a simulation timestep $\Delta t = 0.005$ and damping factor $\gamma = 1.0$. After equilibration, we marked a certain number fraction $x_B$ as “B” particles with a large cutoff radius $r_{BB}$ between them, using arithmetic mixing rules for the cross interactions.

We then performed short benchmark runs in the NVE ensemble using the standard cell list, stenciled cell list, and LBVHs for building the neighbor list. The
neighbor list buffer width was set to 0.0 to force a rebuild at every simulation step. For the stenciled cell list, we performed most benchmarks with $\Delta_{bin} = r_{AA}$. We performed 1,000 warm-up steps to determine optimal kernel launch parameters for each algorithm, and then profiled the time required to construct the neighbor list during 10,000 simulation steps. We varied $r_{BB}$ from 3.0 to 18.0, choosing values so that an integer number of cells covered the simulation box. We then varied $x_B$ from 10% to 50%. The benchmarked absolute neighbor list build times are reported in Table A.1.

We expected and confirmed that the stenciled cell list performance was essentially indistinguishable from the standard cell list for the single component Lennard-Jones liquid ($x_B = 0.0$) with $\Delta_{bin} = r_{AA}$. For other values of $x_B$ and $r_{BB}$, we scanned over a range of nominal cell widths $\Delta_{bin}$, and found that $\Delta_{bin} = r_{AA}$ gave the optimal performance in almost all systems. This bin width gave neighbor list build times that were between 50% and 100% faster than with $\Delta_{bin} = r_{AA}/2$. For the particles of type A, setting $\Delta_{bin} = r_{AA}/2$ reduced the number of distance checks by about 40% from $\Delta_{bin} = r_{AA}$; however, nearly five times more cells were accessed. The overhead of these additional cell accesses evidently outweighs the benefits of choosing a smaller $\Delta_{bin}$ in our GPU implementation. Accordingly, we set $\Delta_{bin} = r_{AA}$ for all subsequent results.

The authors of Ref. 261 describe an additional optimization for the stenciled cell list to further reduce the number of distance checks. In their CPU algorithm, the maximum distance between cells in the stencil is also computed in addition to the minimum distance. If the maximum distance is shorter than the cutoff, the particle can be included without distance check. We implemented this on the GPU with a single thread and multiple threads per particle, but found that it led to comparable or increased neighbor list build times in our benchmarks, and so chose not to include it in our final implementation.
Table A.1: Benchmarked time per neighbor list build (s/build) for the Lennard-Jones binary mixture using the standard cell list (cell), stenciled cell list (stencil), and LBVH with varying largest cutoff $r_{BB}$ and composition $x_{BB}$.

<table>
<thead>
<tr>
<th>$r_{BB}$</th>
<th>$x_B = 0.1$</th>
<th>$x_B = 0.2$</th>
<th>$x_B = 0.3$</th>
<th>$x_B = 0.4$</th>
<th>$x_B = 0.5$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>cell</td>
<td>stencil</td>
<td>LBVH</td>
<td>cell</td>
<td>stencil</td>
</tr>
<tr>
<td>3.0</td>
<td>0.00044</td>
<td>0.01087</td>
<td>0.02096</td>
<td>0.00027</td>
<td>0.0128</td>
</tr>
<tr>
<td>6.0</td>
<td>0.04337</td>
<td>0.04385</td>
<td>0.02874</td>
<td>0.04184</td>
<td>0.04514</td>
</tr>
<tr>
<td>9.0</td>
<td>0.11824</td>
<td>0.05319</td>
<td>0.04370</td>
<td>0.12234</td>
<td>0.06040</td>
</tr>
<tr>
<td>12.0</td>
<td>0.28644</td>
<td>0.11316</td>
<td>0.06739</td>
<td>0.29787</td>
<td>0.13114</td>
</tr>
<tr>
<td>14.4</td>
<td>0.46182</td>
<td>0.13778</td>
<td>0.09125</td>
<td>0.47775</td>
<td>0.16530</td>
</tr>
<tr>
<td>18.0</td>
<td>0.91959</td>
<td>0.23702</td>
<td>0.14507</td>
<td>0.95992</td>
<td>0.30291</td>
</tr>
</tbody>
</table>
For this synthetic benchmark, the most important measurement is the relative speed of the three neighbor list algorithms. Figure A.6 compares the speedup of the stenciled cell list and the LBVH neighbor list build times versus the standard cell list for different concentrations and cutoffs. Here, a speedup greater than one indicates that the stenciled cell list or LBVH is faster than the standard cell list. For $r_{BB} < 6.0$, the stenciled cell list performs comparably to the standard cell list, while the LBVH is slower. This behavior is expected because the A and B interaction lengths are nearly the same, and it is faster to build and search the cell list than to build and traverse the LBVH. As $r_{BB}$ is increased, both the stenciled cell list and the LBVHs considerably outperform the simple cell list. We note that this speedup is more pronounced for systems with smaller $x_B$ for both algorithms.

![Figure A.6: Speedup in the stenciled cell list (top) and LBVH (bottom) neighbor list build time compared to a standard cell list for the Lennard-Jones binary mixture with varying largest cutoff $r_{BB}$ and composition $x_B$.](image)

Figure A.7 compares the speedup in the neighbor list build time between the LBVHs and the stenciled cell list. Here, a speedup greater than one indicates that LBVH is the faster algorithm. For $r_{BB} < 6.0$, the stenciled cell list typically performs better than the LBVH, as does the simple cell list. The LBVH outperforms the stenciled cell list for larger asymmetries $r_{BB} > 6.0$ and low concentrations $x_B < 0.3$. 

185
This behavior is in agreement with our expectation that trees outperform cell lists for non-uniform or sparse systems. For many colloidal systems, the number density of large particles is typically quite low, so in these systems we might expect LBVHs to outperform the stenciled cell list.

Figure A.7: Speedup in the LBVH neighbor list build time compared to a stenciled cell list for the Lennard-Jones binary mixture of Figure A.6.

A.3.2 Colloid solution

We benchmarked the stenciled cell list and LBVHs for a representative colloidal system, the colloid solution described in Ref. 261. We fixed the solvent density at \( \rho = 0.6 \) and varied the colloid diameter \( a \) and the colloid volume fraction \( \phi = \pi a^3 \rho_C / 6 \) where \( \rho_C \) is the colloid number density. The solvent–solvent interactions were modeled with a standard Lennard-Jones potential with cutoff \( r_{SS} = 3.0 \). The colloid–colloid and colloid–solvent interactions were modeled through integrated Lennard-Jones potentials [240] using the same parameterization as Ref. 261 with the colloid–colloid interactions truncated at \( r_{CC} = 5a/2 \), and the colloid–solvent interactions at \( r_{CS} = a/2 + 4.0 \). The maximum colloid diameter explored was \( a = 20 \), which led to a maximum interaction range asymmetry of 50:3.

We initially equilibrated only the Lennard-Jones solvent coupled to a Langevin heat bath in a cubic simulation box of edge length \( L \) at \( T = 1.0 \) (simulation timestep \( \Delta t = 0.005 \), damping factor \( \gamma = 1.0 \)). We then randomly dispersed \( N_C \) colloids into
the simulation box, and deleted solvent particles within 5% of the colloid–solvent contact distance \((a + 1.0)/2\). The final system parameters are listed in Table A.2, where \(N_S\) is the number of solvent particles after deletion. The resulting dispersion was then equilibrated using the same integration scheme and a neighbor list buffer radius of 1.0. For profiling, we performed 25,000 NVE simulation steps to determine the optimal kernel launch parameters and an additional 25,000 simulation steps for data collection using the stenciled cell list and the LBVHs. During the profiling time, the neighbor list was rebuilt between roughly 1,200 and 1,400 times.

<table>
<thead>
<tr>
<th>(a)</th>
<th>(\phi)</th>
<th>(N_C)</th>
<th>(N_S)</th>
<th>(L)</th>
<th>neighbor list speedup</th>
<th>total speedup</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>0.1</td>
<td>500</td>
<td>158,517</td>
<td>68.9</td>
<td>2.1x</td>
<td>1.5x</td>
</tr>
<tr>
<td></td>
<td>0.2</td>
<td>1,000</td>
<td>123,267</td>
<td></td>
<td>2.1x</td>
<td>1.5x</td>
</tr>
<tr>
<td>10</td>
<td>0.1</td>
<td>250</td>
<td>665,540</td>
<td>109.4</td>
<td>4.1x</td>
<td>2.1x</td>
</tr>
<tr>
<td></td>
<td>0.2</td>
<td>500</td>
<td>548,740</td>
<td></td>
<td>4.1x</td>
<td>2.0x</td>
</tr>
<tr>
<td>20</td>
<td>0.1</td>
<td>100</td>
<td>2,176,957</td>
<td>161.2</td>
<td>8.2x</td>
<td>3.6x</td>
</tr>
<tr>
<td></td>
<td>0.2</td>
<td>200</td>
<td>1,841,931</td>
<td></td>
<td>8.0x</td>
<td>3.3x</td>
</tr>
</tbody>
</table>

Figure A.8 shows the profiling results for three colloid diameters \(a = 5, 10,\) and 20 at volume fractions \(\phi = 0.1\) and 0.2. Here, the solid shaded areas indicate the time required to build the neighbor list while the cross-hatched areas indicate the total simulation time to complete 25,000 steps. In all cases, the total run time is shorter for \(\phi = 0.2\) than \(\phi = 0.1\). This is due to the larger amount of colloid excluded volume, which decreases the number of solvent particles in the simulation.

Table A.2 reports the speedups in the neighbor list build time and total run time. We observe that for all systems the LBVHs outperform the stenciled cell list, with a maximum speedup in total run time of 3.6x. It is clear that as the colloid diameter increases, the speedup of the LBVHs versus the stenciled cell list increases, which is consistent with our expectation that LBVHs are more favorable for decreasing number density from the synthetic Lennard-Jones benchmark. For the
A.4 Conclusions

We developed a parallel algorithm for computing neighbor lists based on linear bounding volume hierarchies on GPUs. We compared this algorithm to a GPU implementation of an established CPU algorithm based on stenciled cell lists. We found that both the stenciled cell list and LBVH algorithms outperform a standard cell list for a synthetic benchmark with asymmetric interaction ranges, and that the highest speedups relative to a standard cell list are obtained when the fraction of particles with large cutoffs is small. These benchmarks also revealed that LBVHs outperform the stenciled cell list for large asymmetries and low number fractions of large parti-
cles. We confirmed this in a realistic colloidal system where the LBVHs consistently outperformed the stenciled cell list.

We have focused our discussion on the GPU implementation of the LBVH algorithm, but we emphasize that a BVH neighbor-list algorithm is also applicable to CPU codes. We observed sizable speedups in the neighbor list build time using CPU BVHs compared to a CPU standard cell list in HOOMD-blue as size asymmetry increased (not reported in this chapter). Both the CPU and GPU BVH neighbor list implementations are available as open-source software starting in HOOMD-blue version 1.3.0, and can be downloaded from the HOOMD-blue webpage.

The LBVH neighbor-list algorithm described here has numerous applications in soft matter simulations, especially when the system of interest has components with disparate length scales. These length scales may be geometric constraints, as is the case for colloidal particles dispersed in an atomistic solvent, or due to disparate interaction length scales. As in graphics processing, LBVH neighbor lists may also find useful applications in systems with uniform length scales but spatially non-uniform particle distributions, as is often the case for studying systems with interfaces or particle clustering.

Although this work has focused on the application of LBVHs to general-purpose molecular simulations, other tree data structures, including octrees and k-d trees, are also likely excellent candidates for neighbor list construction. Higher quality LBVHs may also provide better performance for certain problems [271]. Moreover, hybrid approaches that combine cell lists and hierarchical trees are also known to significantly accelerate graphics processing [262]. These algorithms incur different costs for tree construction and traversal [265], and should be carefully compared to each other in simulation applications. Such alternative hierarchical algorithms are promising avenues for future work.
Appendix B

Efficient mesoscale hydrodynamics: multiparticle collision dynamics with massively parallel GPU acceleration

Complex fluids and soft matter, including colloidal suspensions, polymer solutions, and biological materials, are characterized by structure on disparate length scales, dynamics that occur over a range of time scales, and interactions on the scale of thermal energy [281, 282]. This combination of properties can give rise to many complex behaviors, including non-Newtonian rheology [283–285] and the ability to self-assemble into organized structures [20, 21]. Computer simulations have emerged as useful tools for studying complex fluids and soft matter because they offer detailed, simultaneous resolution of structure and dynamics, and can be used to predict and systematically engineer a material’s properties.

Many complex fluids and soft materials consist of a mesoscopic solute, such as polymer chains or spherical colloids, dispersed in a molecular solvent. The solute can be orders of magnitude larger than the solvent and have corresponding slower dynamics [92]. For example, colloidal particles typically have diameters ranging from a few nanometers to micrometers, but are dispersed in solvents such as water that have molecular diameters less than a nanometer. Classical molecular dynamics (MD) simulations [161, 260], such as those described in Appendix A, are often unsuitable for such problems because explicitly resolving the solvent with the solute quickly becomes intractable, both in terms of the number of particles required for the model and the number of simulation time steps that are necessary to study the time scales of interest. On the other hand, continuum constitutive models employed in computational fluid dynamics [286, 287] neglect details of fluctuating microscopic structures, which may be important for accurately describing complex liquids such as polymer solutions when the rheological properties are influenced by the local structure. A simulation approach bridging these regimes to retain a detailed model of the solute while resolving only the essential effects of the solvent is necessary. Various mesoscale methods have been developed for this purpose, including lattice-based models such as the family of Lattice-Boltzmann methods [288–290]; implicit-solvent models including Brownian dynamics [161], Stokesian dynamics [291], and fast lubrication dynamics [292]; and particle-based methods such as dissipative particle dynamics [210, 293, 294], direct simulation Monte Carlo [295, 296], and multiparticle collision dynamics (MPCD) [80, 81]. The reader is referred to Ref. 82 and the references therein for an excellent discussion on the various advantages and disadvantages of these methods, particularly as applied to colloidal suspensions. This chapter focuses on the MPCD method, first developed by Malevanets and Kapral [80], and its massively parallel implementation for graphics processing units (GPUs).
In MPCD, the solvent is represented as a set of off-lattice point particles that undergo ballistic streaming steps followed by cell-based, stochastic multiparticle collisions [80, 81, 297]. The nature and frequency of the collisions determines the transport coefficients of the solvent [91, 298, 299]. The MPCD algorithm is unconditionally numerically stable, has an H-theorem, and naturally includes the effects of thermal fluctuations [80]. Polymers [99, 300], colloids [82, 301–303], and fluid-solid boundaries [87, 304, 305] can be coupled to the solvent, making MPCD an excellent tool for studying soft matter [77, 78, 81, 306–310]. The MPCD equations of motion are less computationally demanding to integrate than those in MD because MPCD particles do not usually have interactions with each other (e.g., dispersion forces), which typically require some of the most time-consuming calculations in the MD algorithm [161, 260]. On the other hand, MPCD can require a high particle density in order to achieve a sufficiently liquid-like solvent [91], which still poses significant demands on the memory required to represent the particle configuration and the number of calculations required to propagate the solvent particles.

MPCD is particularly well-suited for parallelization because of its particle-based nature and spatially localized collisions. Yet, to our knowledge, there are few publicly available massively parallel implementations of the MPCD algorithm. RMPCDMD [311] is a multithreaded program for performing MPCD simulations, but lacks support for decomposing the simulation onto multiple CPUs, limiting the simulation sizes that can be reasonably accessed. The SRD package [96] of the massively parallel MD code LAMMPS [258] implements a limited set of MPCD features and supports spatial decomposition onto multiple CPU processes using the Message-Passing Interface (MPI). Recently, Westphal et al. [312] described an implementation of the MPCD algorithm for GPUs. GPUs have become indispensable for performing MD simulations at large scale because of their massively parallel architectures, and essentially all major MD software packages support GPU acceleration to some extent.
Westphal et al. demonstrated a 1-2 order of magnitude acceleration of their MPCD code on a GPU compared to single-threaded CPU code [312]. However, their implementation, which to our knowledge is not publicly available, was designed to use a single GPU as an accelerator, and so the GPU memory capacity effectively limits the size of the simulations that can be performed. It is critical to be able to utilize multiple GPUs through MPI in order to efficiently perform large-scale MPCD simulations.

In this chapter, we present a high-performance, open-source implementation of the MPCD algorithm within the HOOMD-blue simulation package [122, 123] that scales to run on hundreds of NVIDIA GPUs. We first give an overview of the MPCD algorithm (Section B.1). We then describe the implementation of this algorithm with a focus on optimization for the GPU and for supporting multiple GPUs with MPI (Section B.2). We assess the accuracy and performance of a mixed-precision computing model for the MPCD solvent. We report weak and strong scaling benchmarks of a reference MPCD solvent and a benchmark of a polymer solution (Section B.3). We demonstrate that massively parallel GPU acceleration can enable simulations of mesoscale hydrodynamics at length and time scales that would be otherwise challenging or impossible to access.

B.1 Algorithm

In MPCD [80, 81, 297], the solvent is modeled by point particles of mass $m$ with continuous positions $r_i$ and velocities $v_i$. MPCD particles are propagated through alternating streaming and collision steps. During the streaming step, the particles follow Newton’s equations of motion. In the absence of any external forces, the
positions are simply updated during an interval of time $\Delta t$ according to:

$$r_i(t + \Delta t) = r_i(t) + v_i(t)\Delta t.$$  \hfill (B.1)

The time step $\Delta t$ effectively sets the mean free path for the particles, $\lambda = \Delta t\sqrt{k_B T/m}$, where $k_B$ is Boltzmann’s constant and $T$ is the temperature. Particles are then binned into cubic cells of edge length $a$, which sets the length scale over which hydrodynamics are resolved [313]. For values of $\lambda$ much smaller than $a$, the MPCD algorithm violates Galilean invariance, which can be restored by applying a random shift to the particle positions of $\pm a/2$ before binning [89, 314]. All particles in a cell undergo a stochastic collision that updates their velocities while conserving linear momentum. (Collision rules can be extended to also enforce angular-momentum conservation [81, 315].) These stochastic collisions lead to a build up of hydrodynamic interactions, and the choice of collision rule and solvent properties determine the transport coefficients of the fluid.

In the stochastic rotation dynamics (SRD) collision rule [80], the center-of-mass velocity of each cell, $u$, is first computed and the relative particle velocities are rotated by a fixed angle $\alpha$ about a randomly chosen axis for each cell,

$$v_i(t + \Delta t) = u(t) + R(\alpha) \cdot (v_i(t) - u(t)),$$  \hfill (B.2)

where $R(\alpha)$ is the rotation matrix for the cell. Malevanets and Kapral showed that the SRD rule yields the hydrodynamic equations for compressible flow, has an H-theorem, and gives the correct velocity distribution for the particles [80]. Although SRD in this form conserves linear momentum, Eq. B.2 requires modification in order to also enforce angular-momentum conservation [316, 317].

Because SRD is an energy-conserving collision rule, a cell-level thermostat is required to enforce isothermal conditions in nonequilibrium simulations. Huang
et al. [318] recently explored several thermostat schemes and found the Maxwell-Boltzmann thermostat [319] to be most effective. Here, the current kinetic energy of each cell, \( E_k \), in the reference frame of the cell center-of-mass velocity \( \mathbf{u} \) is first determined,

\[
E_k = \frac{1}{2} \sum_{i=1}^{N_c} m (\mathbf{v}_i - \mathbf{u})^2,
\]

where the sum is taken over the \( N_c \) particles in the cell. Then, a random energy \( \hat{E}_k \) is drawn from the Maxwell-Boltzmann distribution consistent with \( N_c \) at the desired temperature, and the relative particle velocities are scaled by \( \xi = \sqrt{\hat{E}_k/E_k} \).

This method generates average kinetic energies and fluctuations consistent with the canonical ensemble [318, 319].

Alternatively, the Andersen thermostat (AT) collision rule [298] implicitly generates isothermal conditions. In an AT collision, a random velocity \( \delta \mathbf{v}_i \) consistent with the Gaussian distribution of velocities at the desired temperature is chosen for each particle in a cell, and the velocities are updated by

\[
\mathbf{v}_i(t + \Delta t) = \mathbf{u}(t) + \delta \mathbf{v}_i - \frac{1}{N_c} \sum_{j=1}^{N_c} \delta \mathbf{v}_j,
\]

where the sum is again taken over the particles in the cell. The last term enforces linear-momentum conservation in the cell. Angular-momentum conservation can also be enforced for the AT collision rule by applying an additional constraint [316].

Solute particles, such as the monomers of a polymer chain, can be coupled to the solvent through the collision step [300]. Embedded particles propagate using standard molecular dynamics methods [161, 260] between MPCD collisions. Typically the MD timestep is much shorter than the MPCD collision time in order to faithfully integrate the MD equations of motion. The embedded particles are then binned into cells with the solvent particles during the collisions. Care must be taken when com-
puting the center-of-mass velocity or kinetic energy of a cell to appropriately weight the masses of the embedded particles, which are typically different from the solvent.

There are many additional variations and features of the MPCD algorithm [81], including the modeling of fluid-solid boundaries [82, 87, 302–305], nonideal [320] and multiphase [321] fluids, nonequilibrium flows [87, 298], viscoelasticity [322], and deformable objects [323]. A thorough treatment of these aspects of the algorithm is beyond the scope of this chapter. However, the open-source software presented here has been implemented in a way that can be readily extended to support these features.

B.2 Implementation

In this section, we present our implementation of the previously described MPCD algorithms for GPUs. Throughout, we will focus on optimizations for NVIDIA GPUs (Section B.2.1) and use terminology from the CUDA programming model; however, many of the strategies described here can be generalized to other multithreaded programming frameworks. Figure B.1 shows a flow diagram summarizing the steps of the MPCD algorithm required to complete one time step of streaming and collision. Each block represents an independent step in the algorithm, and blocks with dashed outlines require communication with nearest-neighbor ranks in MPI. By separating the steps of the algorithm in this way, we ensure a modular design that can be readily extended.

An MPCD time step proceeds as follows. First, particles are streamed to their new positions according to Eq. B.1. Particles are then migrated onto the appropriate rank (Section B.2.3) and binned into cells (Section B.2.4). Cell properties such as the center-of-mass velocity are then computed (Section B.2.5) before finally the particles undergo a multiparticle collision according to, e.g., Eq. B.2 or Eq. B.4 (Sections B.2.6
Figure B.1: Flow diagram of a single step of the MPCD algorithm. Each block represents an independent step of the algorithm. Blocks with dashed outlines require nearest-neighbor MPI communication for multi-GPU simulations. Particle sorting is an optional step that is performed periodically.

and B.2.7). For performance reasons, the MPCD particle data (Section B.2.2) is optionally resorted periodically after the cell list is constructed (Section B.2.8). Each of these points will be discussed in further detail below.

B.2.1 Framework

We chose to implement the MPCD algorithm in the HOOMD-blue simulation package [122, 123], although the algorithms described in this article can be adapted to other packages. HOOMD-blue is an open-source, general-purpose molecular dynamics code that was designed for NVIDIA GPUs. It is primarily used to study soft matter and perform coarse-grained simulations, and has a rich set of features for these purposes. Unlike other MD software that uses the GPU as an accelerator to the CPU at performance-critical steps, HOOMD-blue performs essentially all computation directly on the GPU [122]. This approach has the advantage that data structures can be optimized for the GPU, and only limited amounts of data must be migrated between the CPU and GPU. The efficient MD engine can be used to simulate particles embedded within the MPCD solvent completely on the GPU. (There
HOOMD-blue additionally supports simulations on multiple CPUs and GPUs with MPI domain decomposition and has been shown to scale to thousands of NVIDIA GPUs [123]. Moreover, it has a modular, object-oriented design and flexible Python user interface, which make it straightforward to implement MPCD, as outlined in Figure B.1, alongside the existing MD engine and integrate it in users’ simulation scripts.

B.2.2 Particle data

We considered two designs for the MPCD particle data structures: using HOOMD-blue’s existing MD particle data or creating a standalone container. The first approach is more convenient for the programmer and is the design adopted in the LAMMPS SRD package. However, considerably more information is usually tracked for MD particles than is required for MPCD particles. This can lead to massive memory waste that restricts the problem sizes that can be studied. MPCD particles are also fundamentally different from MD particles, and so may have different optimal data structures. For example, MPCD particles nearly always have the same mass, and so it is more efficient to save this value once for the fluid rather than per particle. Accordingly, we implemented a standalone container for the MPCD particle data in HOOMD-blue.

The particle data is stored as a structure of arrays. The entries in these arrays for a single MPCD particle are schematically illustrated in Figure B.2. The position and velocity vectors of each MPCD particle are stored as floating-point values. In order to promote coalesced memory accesses, the particle positions and velocities are saved as arrays of four-element floating-point vectors. The precision of these vectors can be selected during compilation. The fourth elements of the vectors are used to store additional data or as caches for data that may be commonly requested concurrently with the position or velocity. Currently, an integer identifying the type of the particle
Figure B.2: Particle data and memory requirements (in bytes) per MPCD particle. Each particle property is stored in a separate array. Position and velocity arrays are four-element vectors of 32-bit (or 64-bit) floating-point values, and tag and flag arrays are 32-bit integers. A duplicate is allocated for each array to facilitate sorting, indicated by $\times 2$.

is stored in the position, and the current cell index of the particle is cached in the velocity. For purposes of saving and manipulating configurations, each particle is assigned a unique identifying integer tag. The particle tag is expected to be rarely accessed during the simulation, and is used primarily for manipulating particle data through the Python user interface. In MPI simulations, an additional integer is allocated per particle to be used as a flag for removing particles. A duplicate array is allocated for each property in order to facilitate fast swapping of data as particles are sorted. Hence, the particle data requires a maximum of 80 bytes per particle for single-precision vectors and 144 bytes per particle for double-precision vectors.

### B.2.3 Domain decomposition

HOOMD-blue employs a standard spatial domain decomposition using MPI for simulations on multiple GPUs [123, 258]. The simulation box is subdivided by planes along each of the box vectors, and one MPI rank is assigned to each local simulation box. When multiple GPUs are available in a node, a two-tier decomposition attempts to place spatially close ranks within the same node. We adopt the same domain decomposition for MPCD as for the MD particles, although the algorithms presented here could be straightforwardly modified to other strategies [257, 324, 325].
The MPCD particles are initially decomposed onto each rank during initialization. They are subsequently migrated between ranks when communication is requested by part of the algorithm. The communication algorithm for the MPCD particles, which is similar to that employed for the MD particles [123], is described below. All steps are performed on the GPU except where explicitly noted otherwise.

Particles which have left the local simulation box are marked with an integer consisting of bitwise flags denoting the directions each particle should be sent. All marked particles are subsequently removed from the particle data and staged into a send buffer, while the unmarked particles are compacted into the local data arrays. After packing, the flags of particles in the send buffer are transformed into destination MPI ranks, and the buffer is sorted by destination. Nonblocking MPI calls then exchange particles with a maximum of 26 unique nearest neighbor ranks. If available, a CUDA-aware MPI implementation [326] could operate on buffers in device memory directly, which would permit the MPI library to optimize these transfers asynchronously and take advantage of GPUDirect technologies [327, 328]. However, in preliminary benchmarks, we found little performance gains from using CUDA-aware MPI for this communication, and so instead the entire send buffer is copied from the device to the host before the MPI calls are made. After the communication is completed, received particles are wrapped back into their destination simulation boxes and appended to the particle data arrays.

The CPU-only code employs a simpler communication pattern, as used in LAMMPS [258], where particles are exchanged with a maximum of 6 nearest neighbors along the cardinal directions in three rounds of MPI calls. Particles received in each round are checked to see if they require additional communication and, if so, are pushed into the send buffer for the next round of communication. This approach increases the message sizes in order to take advantage of the full network bandwidth and decrease communication latency. However, on the GPU, additional kernel calls
and host-device data copies would be required in order to repack the send buffers during each round of communication, leading to a serial bottleneck which makes one round of communication with smaller messages more efficient.

B.2.4 Cell list

Binning particles into local collision cells in order to compute average cell properties like the center-of-mass velocity is central to the MPCD algorithm. In principle, such properties can be computed by direct summation of particle data. However, on the GPU, multiple atomic operations would be required to perform the summation without a read-modify-write race condition, and such operations are typically slow. Instead, it can be more efficient to first construct a list of particles belonging to each cell (Figure B.3a) [312], and then perform the summation on particles within the cells. Moreover, a cell list is more flexible for applying different collision rules (see Section B.2.7) and can be reused for efficient particle sorting (see Section Section B.2.8).

Westphal et al. proposed a sophisticated method for constructing a cell list using hash tables and partial lists with shared memory staging [312]. In their benchmarks, this method performed well on NVIDIA GPUs with the Fermi architecture, but performance deteriorated for GPUs with the Kepler architecture that succeeded Fermi. Since subsequent GPUs more closely resemble the Kepler architecture, we require a different optimized solution. For example, the cell list could be constructed by first assigning a cell index to each particle, and then sorting the particle indexes using the cell index as a key. This approach has been applied successfully in the HALMD package [329], and it has the benefit that the generated cell list can be stored using an array whose length is exactly equal to the number of particles. However, sorting can be slow when the number of key-value pairs is large, as is the case for MPCD.
Additionally, an auxiliary array is required to find the location of each cell’s particles in compact cell list storage, adding to the overhead of reading from the cell list.

In this work, we employ a method for generating the cell list based on limited use of atomic operations that is also currently used in HOOMD-blue for the MD particles. The cell list is stored as a two-dimensional matrix in row-major order [330]. Each row corresponds to a cell, and each column gives the index of a particle in that cell (Figure B.3b). The row-major ordering is beneficial for computing cell properties (see Section B.2.5). Because of the regular data layout, entries in the cell list can be straightforwardly accessed without an auxiliary array. However, enough memory must be allocated so that each cell can hold the maximum number of particles in any cell. This layout leads to some memory waste, particularly if one cell has many more particles than another (Figure B.3c). The number of MPCD particles in a cell should follow a Poisson distribution [81, 319], and we found that the typical cell list memory was only roughly 4 times larger than a perfectly compact array in our benchmarks (see Section B.3). This overhead is comparable to the total memory requirements to build a cell list using an efficient parallel radix sort algorithm [280, 331].

Figure B.3: (a) Schematic illustration of particles binned into MPCD cell list. (b) Two-dimensional row-major layout of cell list with one row per cell and columns containing particles. (c) One-dimensional cell list memory with numbers indicating region for each cell and white space showing unused storage per cell.

The cell list is built as follows. Each particle is first binned into a cell (Figure B.3a). In order to support random grid shifting, particle positions are optionally
translated by a vector, subject to periodic boundary conditions, during binning. A particle is then inserted into the row of its cell using one atomic operation to determine the column. The cell index is also cached into the fourth element of the particle velocity vector for use in subsequent steps of the MPCD algorithm. The particle is not inserted if the determined column index would exceed the allocated size of the cell list. However, in this case, the maximum number of particles that should be inserted into any cell is still implicitly tracked, and the cell list is subsequently reallocated and recomputed. Reallocation occurs infrequently and primarily during the first few times the cell list is built.

In MPI simulations, cells may overlap the boundaries of domains. This is further complicated by grid shifting, since particles may be shifted outside the local simulation box. To accommodate these boundaries, we introduce a layer of ghost cells around the domain (Figure B.4). We compute the minimum number of cells required along each dimension so that a particle shifted from the local simulation box by ±a/2 will still be binned into a cell. Because of this assumption, particles must be migrated to lie within their local simulation boxes before the cell list is built, effectively requiring that particle migration occurs before every MPCD collision. In order to reduce the frequency of particle migration, it has been suggested to use additional ghost cells as a buffer so that particles can diffuse farther off rank [332]. We found that using additional ghost cells was not effective for the GPU because increased cell communication posed a bigger bottleneck.

B.2.5 Cell properties

After the cell list has been built, the cell properties can be computed by iterating over particles in each cell. The main quantities of interest are the center-of-mass velocity and the kinetic energy. The temperature in the cell, which is defined by the kinetic energy relative to the center-of-mass velocity (see Eq. B.3), can be determined from
Figure B.4: Schematic of MPCD cells with domain decomposition. The global and local simulation box boundaries are indicated by solid black lines. Cells requiring communication, including ghost cells outside global simulation box, are shaded grey. Arrows schematically illustrate the point-to-point MPI communication pattern with neighboring ranks used in both particle migration (Section B.2.3) and cell property calculation (Section B.2.5) for the GPU.

these properties. The simplest GPU implementation of this calculation would use one thread per cell to loop over the member particles and accumulate the momentum and kinetic energy [312]. However, this approach has limited parallelism, especially when scaling to multiple GPUs, because the number of cells may be too small to fully occupy a GPU.

Instead, we compute the cell properties using an array of $w$ threads to process the particles in a cell. Using multiple threads increases parallelism and also promotes coalesced reads from the row-major ordered cell list. Each thread reads from the cell list in a strided fashion to compute the total momentum, mass, and kinetic energy of the particles it has accessed. The number of threads $w$ is restricted to be a power of two less than the size of a CUDA warp so that these quantities can then be reduced using shuffle instructions available on Kepler and newer NVIDIA GPUs that allow threads within a warp to efficiently read from each other’s registers. The first thread within each array then computes the center-of-mass velocity from the momentum, determines the temperature, and writes the final result. The optimum value of $w$
depends on the MPCD particle density, the volume of the simulation box, and the specific GPU hardware and is determined using runtime autotuning [123].

It is essential to accurately compute the cell center-of-mass velocity because the MPCD collisions are performed relative to this quantity, and inaccuracies can lead to momentum drift and poor energy conservation. Accordingly, all reductions are performed with double-precision floating-point values even if the particle velocities are stored in single precision. Alternatively, we could perform the operations using double-single floating-point arithmetic [329] or fixed-point arithmetic [333]. These strategies replace 64-bit floating-point arithmetic with multiple equivalent 32-bit floating-point operations or 64-bit integer fixed-point arithmetic, respectively, which can be faster for certain GPU architectures due to differences in instruction throughput. However, we found that the primary bottleneck in the cell property calculation was actually the memory rather than arithmetic operations, and so chose to simply use double-precision floating-point operations for reducing the cell properties (see Section B.3.1).

In MPI simulations, communication between ranks is required in order to compute properties in cells overlapping the boundaries. This communication pattern is regular provided that the domain decomposition does not change during the simulation. We adopt a similar communication pattern as for the particle migration on the GPU, and employ nonblocking MPI calls to a maximum of 26 nearest neighbors for both CPU and GPU code pathways. All buffers are packed and unpacked on the GPU if available, and only the minimum required amount of data is communicated. Still, the amount of data that must be transferred from the device to host is large, leading to a significant latency. To mask this latency, we divide the computation of cell quantities into two stages. First, we compute the properties for “outer” cells requiring communication (shaded cells in Figure B.4). We then begin the MPI communication, and calculate properties of cells lying fully within the local simulation box (white
cells in Figure B.4). Last, we finalize communication from the outer cells and finish reducing their properties. We found this strategy to be much more efficient than performing three rounds of communication with fewer, larger messages because it (1) does not lead to serial bottlenecks of host-device data migration and (2) allows computation to overlap with network communication.

The total momentum, energy, and temperature of the system may be required for measurement purposes. These quantities can be obtained by summation of the cell properties, which are reduced using a parallel device-wide reduction scheme. In MPI simulations, we ensure that cells overlapping the boundary are only included in this sum by one of the ranks. The net properties are then reduced across all MPI ranks with a collective communication call.

### B.2.6 Stochastic rotation dynamics

Once the MPCD cell list and cell properties have been computed, it is straightforward to apply the SRD collision rule to the particle velocities. The most important technical challenge to address is how to randomly draw and store the rotation matrix for each cell. The rotation vector must be picked randomly and uniformly from the surface of the unit sphere. We employ the cylindrical projection method for picking points, which requires drawing only two uniform random numbers. This method is particularly efficient on the GPU compared to rejection sampling methods because there is no branch divergence. The rotation axis for each cell is stored in double precision to ensure numerical accuracy of the rotation.

We generate the uniform random numbers using a cryptographic-hash approach to create unique independent microstreams of random numbers [162]. This approach has proven to be highly useful for massively parallel computing because it allows independent threads to generate random numbers without needing to access or advance a shared state [334]. In this work, we employ the Saru random number generator
used throughout HOOMD-blue, although other generators could be straightforwardly substituted. Saru takes three seeds to its hash to initialize a compact state [162]. To generate random vectors for the cell, we feed three seeds: (1) the unique global index of the cell (within the entire simulation box), (2) the current simulation timestep, and (3) a user-supplied seed. This choice of seeding ensures a unique stream (1) for each cell, (2) at each time step, and (3) between simulations. Moreover, because the global cell index is used, the chosen rotation vector for any given cell can be reproduced at any given time step on any rank, regardless of the previous history of drawing numbers. This is particularly useful for MPI simulations because it eliminates the need to communicate random rotation vectors between ranks, significantly decreasing latency.

Particle velocity rotation simply proceeds as outlined in Algorithm B.1. Each particle velocity is loaded into memory, including its cell index, which was stored as the fourth element of its velocity (line 1). The average velocity (line 2) and rotation axis (line 3) for the cell are read from memory, and the rotation is applied to the relative particle velocity (lines 6–7). The center-of-mass velocity is added back to the rotated relative velocity (line 11), and the updated velocity is stored (line 12). We found that, regardless of the precision of the particle velocities, it was necessary to perform the rotations in double precision in order to obtain good momentum conservation. We accordingly upcast the particle velocities to double precision before performing any steps in the rotation (line 4). The precision of the rotated velocity vector is then downcast for storage (line 12). We will discuss the ramifications of this type conversion for accuracy of the simulations in Section B.3.1.

As discussed in Section B.1, the SRD collision rule optionally applies a cell-level thermostat. This step requires first computing the rescaling factor ξ per cell according to Eq. B.3 and then applying this factor to the relative particle velocities. If thermostatting is enabled, we compute ξ for each cell at the same time that we
Algorithm B.1 SRD collision rule.

1: **for each** particle \( i \) in parallel
2: \( \{v_i, c\} \leftarrow \) particle velocity and cell
3: \( u \leftarrow \) center-of-mass velocity for \( c \)
4: \( R \leftarrow \) rotation vector for \( c \)
5: Cast \( v_i \) to double precision.
6: \( \Delta v_i \leftarrow v_i - u \)
7: Rotate \( \Delta v_i \) by angle \( \alpha \) around \( R \).
8: if thermostat enabled **then**
9: \( \xi \leftarrow \) scale factor for \( c \)
10: \( \Delta v_i \leftarrow \xi \Delta v_i \)
11: \( v_i \leftarrow u + \Delta v_i \)
12: Store \( v_i \) in native precision.

draw the rotation vectors. A random kinetic energy is drawn for each cell from a \( \Gamma \) distribution using Marsaglia’s efficient rejection sampling method [335]. This method requires drawing a Gaussian random variable, which we generate by the Box-Muller transformation [336] of two uniform variables to avoid additional branch divergence [337]. Cells having fewer than two particles are assigned a scale factor of \( \xi = 1 \) because the temperature is not defined in these cases. Velocity rescaling is then applied to the relative velocities after rotation but before they are shifted by the center-of-mass velocity and stored (lines 8–10 in Algorithm B.1).

Solute particles are coupled to the MPCD solvent during the collision step. This necessitates including the solute particles in all stages of the algorithm, including cell list construction, cell property calculation, and then the velocity rotation. In HOOMD-blue, the coupled particles are treated as a subset (group) of MD particles. The solute particles evolve according to the standard MD equations of motion between collisions using the velocity Verlet algorithm [161, 260]. Since the solute is coupled only during the collision step, the MPCD particles are only required to be streamed every MPCD collision time \( \Delta t \) and not every MD time step.
B.2.7 Andersen thermostat

The AT collision rule requires the generation of a set of random velocities per MPCD particle and the subsequent reduction of these velocities within each cell in order to apply Eq. B.4. We first draw random velocities for each particle from a Gaussian distribution using the Box-Muller transformation. The uniform random values are generated using Saru with the timestep, particle tag, and a user-supplied seed. These velocities are then summed using the methods described in Section B.2.5. The particle velocities are then updated by adding the randomly drawn velocity for each particle to the cell center-of-mass velocity and subtracting the summed random contribution. The velocity update step is simpler than the SRD algorithm (Algorithm B.1) because only simple summation is required and the AT collision rule implicitly thermostats the solvent. However, we found that the AT collision rule performed slower than SRD overall for two reasons: (1) more random numbers must be drawn (per-particle rather than per-cell), and (2) an additional reduction is performed for each cell. This second step especially incurs a performance penalty in MPI simulations because additional communication is required compared to SRD. To help mask some of this latency, we overlap the process of drawing the random velocities with communication during the initial calculation of the center-of-mass velocities for each cell.

B.2.8 Particle sorting

Many steps of the MPCD algorithm involve processing particles within one cell. Performance of these steps can be improved by first reordering the particle data into cell order [312], which improves data locality. HOOMD-blue uses a three-dimensional Hilbert curve to sort the MD particles [122], which is beneficial for evaluating pair forces between particles. The MPCD particles can be sorted more simply into the
order they reside in the cell list since this strategy gives optimal ordering for cell property calculation during the first MPCD collision after sorting. Random grid shifting, particle diffusion, and insertion order of particles into the cell list will decrease this ordering on subsequent collisions, but particles still retain some locality. Using the cell list to sort the particles is particularly convenient because it does not require significant additional calculations, and sorting can be injected into the usual MPCD algorithm (see Figure B.1). The ability to sort the MPCD and MD particles independently is another advantage of the self-contained MPCD particle data structure described in Section B.2.2.

The particles are sorted as follows. First, the cell list is constructed as described in Section B.2.4. If sorting should occur at the current MPCD step, the two-dimensional cell list array (Figure B.3b) is compacted into a one-dimensional list of MPCD solvent particle indexes. The compaction step is necessary in order to remove any empty entries from cells with fewer particles than the maximum allocation per cell (Figure B.3c). Embedded solute particle indexes are also removed to preserve the Hilbert-curve ordering already applied independently to the MD particles. The embedded particles usually comprise only a small fraction of particles in the cells, and so their ordering should have negligible impact on performance. The MPCD particle data is then sorted according to the compacted list, and the indexes in the cell list are updated to the new ordering.

Although particle sorting improves performance of other steps of the MPCD algorithm, there is a significant cost associated with data movement during the sort, especially on the GPU. There is accordingly an optimum frequency of sorting that balances the improved performance from data locality with this added cost. The optimum will depend on the specific properties of the MPCD fluid, the number of particles per GPU, and also the GPU architecture. We therefore advocate tuning the sorting frequency with a series of short simulations to achieve maximum performance.
B.3 Performance

Performance was benchmarked using the SRD collision rule with typical simulation parameters. MPCD particles having unit mass \( m \) were randomly placed into a cubic simulation box of edge length \( L \) at number density \( \rho = 10/a^3 \). Here, the MPCD cell size \( a \) defines the unit of length. The particle velocities were drawn randomly from the Gaussian distribution consistent with temperature \( T = \varepsilon/k_B \), where \( \varepsilon \) is the unit of energy. The SRD rotation angle was \( \alpha = 130^\circ \), and the time between collisions was \( \Delta t = 0.1 \tau \), where \( \tau = \sqrt{ma^2/\varepsilon} \) is the unit of time. Random grid shifting was applied to ensure Galilean invariance. With this choice of parameters, the solvent viscosity was \( 8.7 \varepsilon \tau/a^3 \) and the estimated Schmidt number was 14 [28]. These values are consistent with a liquid-like solvent [91, 338]. The MPCD particle data was sorted every 25 collisions.

We performed four benchmarks of our MPCD implementation. We first tested the accuracy and performance of the mixed-precision model (Section B.3.1) for this SRD fluid. We then assessed the weak and strong scaling efficiency of our implementation (Section B.3.2). Finally, we performed a research-relevant benchmark of polymer chains in solution (Section B.3.3). These benchmarks are described in detail next and summarized in Table B.1.

B.3.1 Precision

HOOMD-blue’s MD codes can be compiled to use single-precision or double-precision floating-point values for its data structures and calculations. Single precision can give a sizable performance increase compared to double precision due to reduced data size and higher 32-bit floating-point arithmetic instruction throughput. Our MPCD implementation is often bottlenecked by accessing particle data in global memory, and so MPCD should similarly benefit from using single-precision floating-point values.
Table B.1: Summary of benchmarks. $L$ is the edge length of the simulation box, and $N$ is the number of MPCD particles. For the weak scaling benchmark, $L$ refers to the size of a cubic box per node. The MPCD parameters for all benchmarks are density $\rho = 10/a^3$, temperature $T = \varepsilon/k_B$, SRD rotation angle $\alpha = 130^\circ$, and collision time $\Delta t = 0.1\, \tau$.

<table>
<thead>
<tr>
<th>benchmark</th>
<th>$L$</th>
<th>$N$</th>
<th>simulated time</th>
<th>hardware</th>
</tr>
</thead>
<tbody>
<tr>
<td>precision (Section B.3.1)</td>
<td>50 $a$</td>
<td>1.25 million</td>
<td>$10^5, \tau$</td>
<td>Tesla P100 /</td>
</tr>
<tr>
<td></td>
<td>100 $a$</td>
<td>10 million</td>
<td>$10^5, \tau$</td>
<td>GeForce GTX 1080</td>
</tr>
<tr>
<td>weak scaling (Section B.3.2)</td>
<td>50 $a$ / node</td>
<td>1.25 million / node</td>
<td>$500, \tau$</td>
<td>Blue Waters (Tesla K20x)</td>
</tr>
<tr>
<td>strong scaling (Section B.3.2)</td>
<td>400 $a$</td>
<td>640 million</td>
<td>$500, \tau$</td>
<td>Blue Waters (Tesla K20x)</td>
</tr>
<tr>
<td>polymer (Section B.3.3)</td>
<td>50 $a$</td>
<td>1.25 million</td>
<td>$2000, \tau$</td>
<td>Tesla P100</td>
</tr>
</tbody>
</table>
when possible. In what we will refer to as the mixed-precision model for our implementation, particle positions and velocities are stored in single precision instead of double precision (Section B.2.2), and single-precision floating-point operations are used when possible in the algorithm. However, some key steps always require double-precision values and arithmetic for numerical accuracy (Sections B.2.5 and B.2.6), prohibiting the exclusive use of single-precision floating-point operations throughout. In these steps, the single-precision particle data is upcast to double precision, and intermediate calculations are performed using double-precision values before down-casting the result to single precision. Here, we study the numerical accuracy and performance of the SRD algorithm for this mixed-precision model compared to a fully double-precision model.

We measured the $x$-component of the center-of-mass velocity $v_{c,x}$ of the benchmark SRD fluid for box sizes $L = 50\ a$ and $L = 100\ a$ over $10^6$ MPCD time steps. (Comparable results were obtained for the $y$ and $z$ components.) Random initialization of the particle velocities imparted a nonzero initial value, which we aimed to remove by distributing a constant correction to each particle, as is the typical strategy [161, 260]. For the double-precision model, $v_{c,x}$ was essentially zero for all times considered after the initial correction was applied ($|v_{c,x}| \lesssim 10^{-16}\ a/\tau$). For the mixed-precision model, however, this process still left a small nonzero value due to loss of precision when the velocities were stored in single precision after the correction. The center-of-mass velocity then fluctuated around zero during the simulation (Figure B.5). Importantly, we did not observe any significant systematic drift over the simulation times considered, corresponding to good global momentum conservation. The absolute value of $v_{c,x}$ remained small and bounded, and decreased for larger numbers of particles (larger $L$). It appeared that $v_{c,x}$ for the mixed-precision model was essentially a random variable due to the digits that were lost when casting the precision of the particle velocities for rotation since a single-precision floating-
point value typically contains only 7 or 8 significant digits [339]. Comparable values of \( v_{c,x} \) are obtained in MD simulations with good momentum conservation [329]. We accordingly judged that the accuracy of the mixed-precision model was acceptable and that mixed precision could be used provided that it improved performance.

![Figure B.5: Center-of-mass velocity of benchmark SRD fluid along \( x \) direction, \( v_{c,x} \), in mixed-precision model over \( 10^6 \) MPCD time steps for box sizes \( L = 50 \) \( a \) and 100 \( a \).](image)

We tested the performance of the double- and mixed-precision models on two recent NVIDIA GPUs, Tesla P100 and GeForce GTX 1080, using CUDA 8.0. The average performance in time steps per second (larger is faster) is shown in Figure B.6 for the \( L = 100 \) \( a \) box. For the Tesla P100, the mixed-precision model was 1.7x faster than the double-precision model. This difference is close to the maximum 2x speedup that could be obtained from the difference in peak theoretical floating-point performance in single and double precision. For the GeForce GTX 1080, the speedup from the mixed-precision model was slightly smaller at 1.5x. Our benchmarks indicate performance is mostly limited by data accesses, which is perhaps unsurprising for the MPCD algorithm where many key steps involve only little computation. The MPCD algorithm benefits accordingly from the mixed-precision model, which reduces the size of the particle data while delivering reasonable accuracy and performance.
Figure B.6: Time steps per second for SRD collision rule in double precision and mixed precision on NVIDIA Tesla P100 and GeForce GTX 1080 GPUs for the $L = 100 a$ simulation box.

B.3.2 Scaling

Efficiently utilizing multiple GPUs is a requirement for studying problems too large to reside in the memory of a single GPU and also decreases the time required to complete a simulation. We performed scaling benchmarks of our MPCD implementation on Blue Waters, hosted by the National Center for Supercomputing Applications at the University of Illinois at Urbana-Champaign. HOOMD-blue was compiled using the GNU Compiler Collection (version 4.9.3) and CUDA 7.5 with the double-precision model to provide a performance baseline. The GPU code was tested using Blue Waters’s Cray XK7 nodes, each of which contains one NVIDIA Tesla K20x GPU and one AMD Interlagos 6276 CPU with 8 floating-point cores. Previous benchmarks of HOOMD-blue’s MD code found no benefit of multiplexing multiple MPI processes onto the same GPU, and we expect the same to hold true for MPCD. Accordingly, one MPI rank was assigned per XK7 node. The CPU code was tested using Blue Waters’s Cray XE6 nodes with 16 MPI ranks per node. Each XE6 node has two AMD Interlagos 6276 CPUs (16 total floating-point cores). We chose to compare the CPU and GPU code using these configurations to obtain a practical test of the maximum performance that could be achieved per node hour.
In order to test scaling of the MPCD code to problems with large length scales, we performed a weak scaling test with a constant number of particles per node. A cubic box with edge length $50\,a$ was assigned per node so that there were 1250000 particles per node on average. We first performed 2000 SRD time steps in order to allow all runtime autotuners to determine their optimal parameters. These parameters were then fixed, and the performance was measured over 5000 SRD time steps with HOOMD-blue’s internal profiler enabled. We found that overhead from profiling incurred a small performance penalty of roughly 5% for the GPU in this test. Each benchmark was repeated three times to determine the average performance, and the 95% confidence interval was estimated from these measurements.

Both the CPU and GPU codes gave excellent weak scaling performance up to 1024 nodes, as shown in Figure B.7. At this largest node count, there were over one billion MPCD solvent particles in the simulation box. The CPU weak scaling efficiency relative to two nodes was greater than 90% for all node counts tested, while the GPU scaling efficiency was greater than 65%. Most of the loss of efficiency in the GPU code came from an increase in the time required to compute cell properties, which requires the most communication in our implementation. There was also a large initial drop in efficiency for the GPU code as the domain decomposition was increased from one dimension (at 2 nodes) to three dimensions (at 8 nodes). This drop was expected because the amount of communication per node increased. The CPU code did not show this initial drop because there were 16 MPI ranks per node, and so the simulation box was three-dimensionally decomposed for all node counts. The inset to Figure B.7 shows the absolute performance of the two codes in time steps per second. The GPU code was roughly 3 times faster than the CPU code on 2 nodes and 2 times faster on 1024 nodes. For reference, the theoretical peak double-precision performance of the K20x is roughly 4 times the XE6 node.
Figure B.7: Weak scaling efficiency for benchmark SRD fluid on NCSA Blue Waters. One cubic simulation box with $L = 50 \, a$ was replicated per node so that there were 1250000 particles per node on average. GPU benchmarks (circles) were performed using 1 MPI rank (1 NVIDIA Tesla K20x GPU) per XK7 node, while CPU benchmarks (squares) used 16 MPI ranks (16 floating-point cores) per XE6 node. The inset shows the performance in time steps per second (TPS).

We then tested the strong scaling performance of the MPCD code for a large simulation box with $L = 400 \, a$, which corresponded to 640 million solvent particles. Efficient strong scaling is required to fully utilize computational resources and decrease the time to solution. As for the weak scaling tests, we measured the time required to complete 5000 SRD time steps with internal profiling enabled after runtime autotuning was completed. The performance overhead of this profiling was again roughly 5% on the GPU for this test, and was most significant at the largest node counts. We measured the scaling from 32 nodes up to 1024 nodes. Although the CPU code could be run on smaller node counts, the GPU code required a minimum of 32 nodes due to the limited capacity of the GPU memory (6 GB). Each benchmark was repeated three times.

Figure B.8 shows the strong scaling performance and efficiency (inset) of both the CPU and GPU codes. The CPU code exhibits excellent strong scaling, with over 80% efficiency on 1024 nodes (16384 MPI ranks). The GPU code exhibits similarly good scaling, with some efficiency lost at high node counts. In order to investigate
this loss of efficiency further, we separated the time required per step into three main components: particle migration, cell property calculation, and all other steps (including streaming and collision) that require no communication. Figure B.9 shows these components for various node counts. It is clear that the communication steps, especially cell property calculation, limit the scaling efficiency at high node counts.

Figure B.8: Strong scaling performance for benchmark SRD fluid in a cubic simulation box with $L = 400 \, a$ on NCSA Blue Waters. The total number of particles was $N = 640000000$. GPU benchmarks (circles) were performed using 1 MPI rank (1 NVIDIA Tesla K20x GPU) per XK7 node, while CPU benchmarks (squares) used 16 MPI ranks (16 floating-point cores) per XE6 node. Solid lines indicate ideal scaling, and the inset shows the strong scaling efficiency.

Figure B.9: Profile of GPU strong scaling for SRD fluid on NCSA Blue Waters (see Figure B.8). Required time per step is shown for the total simulation (circles), core SRD methods (squares), particle migration (triangles), and computing cell properties (diamonds). Solid black line indicates ideal scaling of the cell property communication.
At small node counts, the cell property calculation is in principle dominated by the inner cells, which can effectively mask most of the latency from communication. At larger node counts, however, the latency associated with both host-device data migration as well as the MPI communication itself become significant. Theoretically, it is expected that the time for communication of cell properties should scale with $1/P^{2/3}$ for $P$ MPI ranks due to the surface-to-volume ratio of the domain decomposition [332]. Initially, we did indeed observe this scaling as indicated in Figure B.9, but it is clear that for $P > 256$ much weaker scaling was obtained. More detailed profiling revealed that this bottleneck was due to host-device data migration. This latency might be reduced using an MPI library that operates on device buffers directly since the library can optimize copies asynchronously to pipeline messages or take advantage of GPUDirect RDMA technologies [326–328]. Unfortunately, we found essentially no performance benefits from these optimizations on Blue Waters due to limited support for GPUDirect RDMA.

Overall, the GPU code is faster than the CPU code by a factor between 2 and 3. As a point of reference, we note that our CPU code also outperforms a comparable benchmark with LAMMPS SRD (31 March 2017 release) [96, 258] by nearly a factor of 2 when running on 128 XE6 nodes on Blue Waters. (There are some minor SRD parameter differences in the LAMMPS benchmark due to limitations of that implementation.) Our weak and strong scaling benchmarks clearly demonstrate the feasibility of using GPUs to efficiently perform and accelerate MPCD simulations at massive scale.

After finalizing our benchmarks on Blue Waters, we determined an improved method for packing particle data for migration compared to the version that we initially tested. In the improved version, we partition the indexes of the particles to be migrated from the particles that are to be retained. Holes in the particle data arrays left by migrated particles are backfilled from the list of retained particles to maintain
compact arrays, reducing the number of read and write operations compared to our
previous implementation based on stream compaction. For the SRD benchmark
fluid with $L = 200\, a$ on 8 NVIDIA Tesla P100 GPUs, this optimization reduced the
time to pack the MPI buffers by a factor of 12.5x, the overall particle migration
time by a factor of 2.8x, and the total simulation time by a factor of 1.3x. The
performance timings reported in Figures B.7–B.9 should also be similarly improved
by this optimization.

B.3.3 Polymer solution benchmark

In addition to the pure SRD fluid benchmarks, we also performed a more complex
simulation of a solution of polymers embedded in the benchmark SRD solvent on
an NVIDIA Tesla P100 GPU using the double-precision model and CUDA 8.0. The
polymers were represented by a bead-spring model as described in [28]. Each polymer
chain consisted of 50 beads, and the number of chains was varied from 16 to 1024,
giving between 800 and 51200 total monomers in the simulation box with $L = 50\, a$.
The equations of motion for the polymers were integrated between collisions with the
solvent using the velocity Verlet algorithm with a time step of $0.002\, \tau$. (An MPCD
collision occurred every 50 MD timesteps.) The MD neighbor list used to compute
nonbonded pair forces had a buffer radius of $0.4\, a$. The MPCD solvent particles were
streamed every 50 MD timesteps because their updated positions were only required
for collisions. We performed a short run of $200\, \tau$ to determine the optimal runtime
kernel launch parameters and then profiled performance for $2000\, \tau$.

In order to measure the contribution of the MPCD solvent to the simulation time,
we performed simple Langevin dynamics simulations [162] of the polymers as a base-
line. To obtain comparable long-time polymer dynamics, we adjusted the monomer
friction coefficient as in Chapter 5 to give a long-time polymer diffusion coefficient
consistent with the value predicted by the Zimm model for dilute polymer solutions
We emphasize that this baseline does not include any hydrodynamic interactions, which is a clear advantage of the MPCD simulations, and simply serves as a point of comparison of the contributions of MD and MPCD to the overall simulation time.

The total required simulation time increased nearly linearly with the monomer density, shown in Figure B.10. The density spanned from the dilute to semidilute polymer concentration regimes [28]. Initially, the Langevin dynamics simulation time did not increase much with density due to limited occupancy of the GPU for small numbers of MD particles. The required time for the MPCD simulations closely tracked the Langevin dynamics simulations, indicating that the performance of the MPCD algorithm itself did not depend significantly on the polymer concentration. This trend is reasonable because the embedded particles comprise only a small fraction of the particles in an MPCD cell. The MPCD simulations were at most 1.4x slower than the Langevin dynamics simulations, which is only a modest cost to incorporate hydrodynamic interactions. Even though there were many more solvent particles than monomers, the MPCD streaming and collision steps happened infrequently compared to the MD step.

Figure B.10: Required time to simulate polymer solutions at various concentrations for 2000 $\tau$ using MPCD (circles) and Langevin dynamics (squares) on an NVIDIA Tesla P100 GPU.
This benchmark demonstrates the strengths of implementing MPCD within an existing MD package designed to run exclusively on the GPU. We were able to easily perform simulations with up to 51200 monomers embedded in 1250000 solvent particles on a single GPU and obtained excellent performance. Moreover, the MD algorithms could be used as a “black box” without special considerations for hiding latency of migrating MD particle data to the GPU.

B.4 Conclusions

We developed an implementation of the MPCD algorithm for simulating mesoscale hydrodynamics that is optimized to exploit the massively parallel computational capabilities of GPUs. Spatial domain decomposition onto multiple GPUs is supported with MPI, enabling simulations of mesoscale hydrodynamics at length and time scales that would be otherwise challenging or impossible to access. We showed that our MPCD implementation efficiently scales to hundreds of GPUs. Here, the performance was primarily bottlenecked by host-device data transfers, which we speculate may be mitigated in computing environments having multiple GPUs within a node and using a CUDA-aware MPI library in conjunction with GPUDirect RDMA. We found that adopting a mixed-precision computing model for the MPCD particle data improved performance on a single GPU with acceptable numerical accuracy. We also showed for a benchmark polymer solution how MPCD can be used to incorporate hydrodynamics into research-relevant simulations with only modest additional computational cost. Our MPCD implementation is available open-source as part of the HOOMD-blue project beginning in version 2.3.0 and should prove useful for conducting simulations of soft matter and complex fluids when hydrodynamic interactions are important.
In this work, we have focused on implementing the most fundamental components of the MPCD algorithm. However, there are many extensions of MPCD that are of interest for areas of active research, including coupling the solvent to solid boundaries such as walls and imposing external fields to drive solvent flow. We have designed our software to be modular in order to readily support these extensions. For example, fluid-solid coupling and flow could be added by modification of the streaming step. We hope to continue to expand our software with support from the community to incorporate these and other features in the future.
Bibliography


