Exploring the Limits of Hard Shape Models for Self-Assembly

by

Vyas Ramasubramani

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Chemical Engineering)
in the University of Michigan
2021

Doctoral Committee:

Professor Sharon C. Glotzer, Chair
Professor Ronald G. Larson
Professor Xiaming Mao
Professor Robert M. Ziff

Vyas Ramasubramani

vramasub@umich.edu

ORCID: 0000-0001-5181-9532

To my parents, who made everything possible.
And to Patti. I miss you.

Acknowledgments

The complete list of people who I need to thank for helping me make it this far would span many pages. I don't have that kind of space here, but I'll do my best.

First and foremost, I have to thank my advisor Professor Sharon Glotzer, who sets impossibly high standards as both a brilliant scientist and an innovative leader and somehow lives up to them every day. I'd also like to thank my committee members. Professor Bob Ziff has been a delight, not only to learn from, but also to work with as a GSI and just to chat with about everything from science to life in Ann Arbor. Thanks also go to Professor Ron Larson and Xiaoming Mao for taking the time to discuss my work and keep me on the right path.

Special thanks go to a few senior members of the group who have helped me along the way. To Karen Coulter goes the credit for keeping the roof from falling in on a daily basis. Dr. Joshua Anderson deserves a huge acknowledgment not only for maintaining so much of the code that makes our research possible, but also for setting an incredible example of how to be a scientific software developer that has allowed me to flourish in that role. Dr. Jens Glaser, I've often described learning from you as drinking from a fire hose: I wouldn't have learned even half as much if not for taking on that incredibly rewarding challenge, so thank you.

Next, the friends I've had the joy of working with over the past 5+ years. Shannon, as glad as I am to be finishing up grad school, a part of me misses the days of us working on recruitment, peer mentoring, and literally everything else together; thanks doing it all with me. Will, thank you for injecting plenty of levity into my time here and making sure I never keep things too serious; it's been real. Simon and Bradley, you've both been fantastic friends and incredibly fun to work with, and I know we'll continue to work together in the future. Thi, thanks for being both a great friend and a scientific mentor; writing papers with

you has taught me many new ways to think about problems and made me a much better scientist. To other friends I happened to work with, in no particular order: Chrisy Du, James Antonaglia, Julia Dshemuchadse, Bryan VanSaders, James Proctor, Pengji Zhou, Erin Teich, Andrew Karas, Brandon Butler, and Kelly Wang - thank you for all you've taught me, and for making our lab a fun, vibrant place to be.

To Dom, Joey, and Andrew: thanks for making Ann Arbor home. To Efe and Anil, thanks for sticking around to eat all our food and for all the good times we had in the process. Alison, Steven, Mario, Dylan, Alex, Andrew, Stephen, Sam, Sarah, Anil, Doruk, Yeliz - thanks for your support and all the fun over the years.

To PJ Loury, who's been with me pretty much through it all - it's been a long road here, I wouldn't have made it without you, and I'm looking forward to not having to fly across the country to see you. To all my friends outside Ann Arbor who are far too many to name, thank you for your friendship over all these years.

Last but certainly not least, to my parents. Your love and support have meant everything to me over the years. I never could have made it here without you.

The research in this dissertation was made possible by the generous support of numerous contributions. Thank you to the Beyster family for funding my 2019-2020 J. Robert Beyster Computational Innovation Graduate Fellowship. Various aspects of this work were supported by the following: the National Science Foundation, Division of Materials Research Awards # DMR 1808342 and DMR 1409620; the Department of the Navy, Office of Naval Research under ONR award number N00014-18-1-2497; the Office of Advanced Cyberinfrastructure Award # OAC 1835612; the Simons Foundation (256297).

This work used the Extreme Science and Engineering Discovery Environment (XSEDE), which is supported by National Science Foundation grant number ACI-1548562; XSEDE award DMR 140129. This research also used resources of the Oak Ridge Leadership Computing Facility (OLCF), which is a DOE Office of Science User Facility supported under Contract DE-AC05-00OR22725. This research was supported in part through computational resources and services supported by Advanced Research Computing at the University of Michigan, Ann Arbor.

Table of Contents

List of Figures

List of Appendices

List of Abbreviations

AIBN 2,2'-azobis(2-methylpropionitrile).

AIChE American Institute of Chemical Engineers.

ALJ anisotropic Lennard-Jones.

API Application Programming Interface.

APS American Physical Society.

ATT Archimedean truncated tetrahedron.

AWCA anisotropic Weeks-Chandler-Anderson.

BCC body-centered cubic.

BOOD bond-orientational order diagram.

BVH bounding volume hierarchy.

CD cubic diamond.

CI continuous integration.

CNA Common Neighbor Analysis.

CNN convolutional neural network.

CPU central processing unit.

CUDA Compute Unified Device Architecture.

DDAB didodecyldimethylammonium bromide.

DEM discrete element method.

DLS dynamic light scattering.

DVCS distributed version control system.

EDMD event-driven molecular dynamics.

EM electron miscroscopy.

FCC face-centered cubic.

FFT fast Fourier transform.

FOMMS Foundations of Molecular Modeling and Simulation Conference.

FVT Free Volume Theory.

GB Gay-Berne.

GFP green fluorescent protein.

GJK Gilbert-Johnson-Keerthi.

GPU graphics processing unit.

GROMACS GROningen Machine for Chemical Simulations.

GUI Graphical User Interface.

HCP hexagonal-close packed.

HD hexagonal diamond.

HIPAA Health Insurance Portability and Accountability Act.

HPC high-performance computing.

HPMC hard particle Monte Carlo.

JH Johnson subalgorithm.

JIT just-in-time.

JSON JavaScript Object Notation.

LAMMPS Large-scale Atomic/ Molecular Massively Parallel Simulator.

LCC liquid core cluster.

LJ Lennard-Jones.

MC Monte Carlo.

MD molecular dynamics.

ML machine learning.

MPI Message Passing Interface.

MSD mean squared displacement.

NN neural network.

OBB oriented bounding box.

OLCF Oak Ridge Leadership Computing Facility.

OSI Open Source Initiative.

PDB Protein Data Bank.

PMF potential of mean force.

PMFT potential of mean force and torque.

PS polystyrene.

PSC Pittsburgh Supercomputing Center.

PTM polyhedral template matching.

PyPI Python Package Index.

RDF radial distribution function.

SC simple cubic.

SciPy Scientific Computing with Python Conference.

SDSC San Diego Supercomputer Center.

SE Stokes-Einstein.

SED Stokes-Einstein-Debye.

SEM scanning electron microscope.

SES solvent-excluded surface.

SuPrA supercharged protein assembly.

SV Signed Volumes subalgorithm.

SVN Subversion.

TACC Texas Advanced Computing Center.

TBB Intel Threading Building Blocks.

TDD tetragonal diamond derivative.

THF tetrahydrofuran.

TPM 3-(trimethoxysilyl)propyl methacrylate.

VCS version control system.

VMD Visual Molecular Dynamics.

WCA Weeks-Chandler-Anderson.

XP extreme programming.

XSEDE Extreme Science and Engineering Discovery Environment.

Abstract

Effectively coarse-graining particle shape is crucial to extending the time and length scales accessible to particle simulations. While models of biomolecular and other systems with many complex interactions tend to model shape by incrementally collecting smaller degrees of freedom to retain maximum model fidelity, models of colloidal systems where dominant interactions may more easily be identified frequently use highly simplified minimal models based on a single simple interaction potential. Among the most successful examples for modeling the equilibrium behavior of simple spherical particles are the hard sphere and Weeks-Chandler-Anderson potentials. The justification for these models is the van der Waals picture, which asserts that short-ranged repulsive forces are the dominant driver of ordering phenomena in many systems. This work explores the limits of this picture in the context of anisotropic particle self-assembly and discusses various ways in which these simple approaches may be generalized or augmented to accurately model additional important interactions to effectively model complex molecules like proteins.

I first consider two different experimental systems, a system of polymer coated silica polyhedral nanoparticles and a system of supercharged proteins. I show that in the first case a simple hard shape model sufficiently accurate under various different circumstances, such as in the presence of depletants and when the particles are rounded. Meanwhile, in the second case I find that hard shape is not enough; I develop a model combining hard shape and point charges that explains the relative stability of different configurations of these proteins, but this model lacks both the performance and the accuracy to achieve self-assembly.

I pursue two different approaches to improving coarse-grained models of shape. My first approach uses machine learning techniques to attempt to quantitatively assess the role

of different interactions in protein crystallization. I use convolutional neural networks to correlate different physicochemical properties of protein surfaces with properties of the resulting crystals, and I discuss various future studies that may be conducted with this model and data pipeline. My second approach employs a mean-field approach to embed shape into interaction potentials, providing a rigorous means to derive anisotropic analogues to isotropic pair potentials. I implement this approach on graphics processing units and show that it is faster than the state of the art without sacrificing accuracy.

I next apply this mean-field approach to study the dynamics of anisotropic particles. I show that introducing anisotropy leads to fundamental dynamic changes by introducing additional time scales associated not only with rotational motion, but also with nontrivial modes of translation-rotation coupling. This study illuminates the sensitivity of dynamic as well as thermodynamic properties to particle shape and highlights the further importance of efficient coarse-graining methods to study these dynamic behaviors.

Finally, I discuss the various scientific software packages I have developed as part of my research. I first present the *lazy refactoring* heuristic I advanced for efficiently developing reusable software in an academic environment. With this context, I review software tools that I developed or contributed to over the course of my dissertation research. I focus in particular on `freud`, a high-quality, high-performance particle simulations analysis toolkit that I have played a key role in developing.

CHAPTER 1

Introduction

1.1 Coarse-grained particle simulation

Since their advent in the mid-1900s, particle simulation methods like molecular dynamics and Monte Carlo simulation have become a standard component of the research toolkit, offering a powerful complement to experimental in fields ranging from materials science to biochemistry to chemical physics [1–6] Simulations offer a high-throughput technique for exploring large parameter spaces and investigating fundamental mechanisms at nearly arbitrary resolutions. Despite these apparent advantages, however, the range of applicability of such methods remains limited due to both the high dimensionality of parameter spaces of interest and the inherent complexity of accurately modeling various physical phenomena at sufficiently fine resolutions [7, 8]. Overcoming such limitations would confer immense benefits: the reliable mapping of self-assembly building blocks to the resulting structures and their properties would enable the creation of specific materials on demand, while the ability to thoroughly investigate phenomena like protein folding and crystallization could enable any number of biomedical and biotechnological advances.

Coarse-graining methods provide one way out of this complexity trap, promising dramatic increases in performance without degrading model fidelity [8–10]. These methods can generally be divided into two classes: bottom-up approaches, which systematically combine detailed (typically atomistic) interactions into simpler effective interactions, and top-down approaches, which parameterize models based on empirical observations at the scales of interest [7]. While rigorous bottom-up approaches are theoretically attractive, top-down approaches are far easier to apply at larger length scales and have been remarkably successful in studying colloidal systems, where relevant interactions may often be relatively easily identified by inspection [11–14]. Moreover, top-down approaches offer a much more direct route to generating minimal models – models containing exactly the details mandated for studying a given phenomenon and no more – by explicitly selecting interactions of

interest, rather than attempting to extract them from a more detailed model.

1.2 Modeling particle shape

Particle shape is one such feature amenable to top-down coarse-graining that is particularly important in the context of assembly-related phenomena. Recent works have shown that particle shape can play a dominant role in directing the self-assembly of colloids [12–15], and the importance of steric effects and shape complementarity in biomolecular (e.g. protein) systems indicates the importance of shape even when other interactions also play significant roles [16]. The importance of shape is a consequence of the van der Waals picture, the empirical observation that material structure in many systems is primarily governed by short-ranged repulsions while attractive forces simply provide a long-ranged, cohesive background force [17]. In such systems where steric effects are dominant, the relevant physics can often be modeled by considering nothing more than the effective shape of a particle. This picture provides the justification for the hard particle model, a minimal model for anisotropic particles interacting solely via excluded volume interactions that has been wildly successful in modeling a wide range of colloidal systems [13, 14, 18–21].

The success of the hard particle model is in part due to the fact that many real colloidal particles are precisely engineered to minimize all other interactions. In systems where other interactions play a significant role – for instance, systems of proteins – the van der Waals picture breaks down and the hard particle approximation becomes a poor one [17]. Indeed, a great deal of literature discusses the many ways in which a simple shape-based colloidal representation of proteins is inadequate to fully describe a complex protein structure [7, 22]. Still, the computational and conceptual simplicity of such models makes them extremely attractive, and researchers have continually expanded the range of systems that can be adequately modeled in this fashion.

Over the years, most efforts to model proteins in this fashion generally focused in one of two directions: classic patchy models [23, 24] based on a hard shape with some embedded interaction sites [25–27], or (possibly rigid) unions of bonded interaction sites judiciously chosen to capture the shape with minimal computational effort [5]. Although the later approach has met with some success, it does not provide the transparent insight into the underlying driving forces as would be desirable from intuition-driven coarse-grained models [7]; meanwhile, more classic patchy models have generally proven too simplified to be accurate except in highly specialized situations [28]. Nonetheless, modern developments in colloidal sciences suggest new ways to more effectively construct these models, and modern machine learning techniques offer a data-driven approach that can more effectively capture

the features of interest by augmenting a researcher's intuition with algorithms capable of identifying relevant trends in vast quantities of data [29–31].

The goal of this thesis is to investigate how the role of shape in various assembly phenomena can be captured effectively using generalizations and extensions of existing approaches. I will explore the extent to which the hard particle model can be used for anisotropic particles, where it breaks down, and potential means to generalize or augment it to achieve efficient, high-fidelity models of specific behaviors. I will also consider how these models can be used to study the dynamics of anisotropic particles, an area that has been explored in far less detail than particle self-assembly. Finally, I will discuss some of the software tools and methodologies I have developed and contributed to over the course of my research, and how they can be applied by other researchers to more effectively conduct their own scientific research.

1.3 Outline of projects

In the first study in this dissertation, chapter 2 discusses the assembly of colloidal truncated tetrahedra into diamond structures in collaboration with experimentalists. In a system precisely tuned so that the tetrahedra behave like nearly hard particles, we will demonstrate that a straightforward hard particle model largely captures the expected assembly behavior of these particles. We then employ rigorous thermodynamic theory to supplement assembly simulations with free energy calculations to elucidate the role of the depletion force in this system. We conclude by expounding upon the role of particle convexity and rounding in this system, demonstrating that these hard particles do not assemble into higher-ordered structures when their shapes are perturbed in undesirable ways.

Chapter 3 moves on to show that the hard particle model alone is often a poor approximation for proteins, which generally have more complex interactions that cannot be coarse-grained away into such a simple representation. First, I discuss a collaborative effort with experimentalists in which we study the assembly behavior of a system of supercharged green fluorescent protein. We show that, due to the overwhelming strength of electrostatic interactions in these supercharged proteins, a straightforward modification of a hard particle approach allows prediction of thermodynamic stability (although not assembly) of these proteins on reasonable time scales. I next propose an alternate machine learning approach to determining more generally the role of shape in the binding and assembly of proteins. I identify various candidate metrics for informing this prediction and develop a workflow to identify the role of various features (including shape) in the assembly propensities of various shapes. I close by proposing various tests of this workflow and future studies that

can exploit it to more thoroughly elucidate the role of shape in protein crystallization and assembly.

In chapter 4, I discuss a novel framework in which particle anisotropy can be rigorously captured in the form of an anisotropic pair potential interaction. I present the theoretical background for this framework, and as a sample case, I demonstrate how anisotropy can be embedded in the popular Lennard-Jones isotropic pair potential using this theory. I then discuss an optimized implementation of this method in the simulation engine HOOMD-blue [32], showing that the resulting anisotropic pair potential outperforms state of the art methods while reproducing some expected assembly results. This framework generalizes existing methods and provides a stepping stone to developing more sophisticated composite representations of complex molecules for the purpose of assembly simulations.

Chapter 5 uses this new framework to conduct a rigorous study of the dynamics of anisotropic particles, a field that has been largely overshadowed by studies of self-assembly [28]. Using a two-dimensional system of polygons, I find that classical theories of Brownian motion fail to recognize additional dynamic features associated with the orientational relaxation of anisotropic shapes that manifest in both mean squared displacements and various time correlation functions. Drawing from standard methods in nonequilibrium statistical mechanics, I develop theory to explain both the emergence of these additional modes and their precise dependence on the shapes present in these simulations. These results hint at the critical role of particle anisotropy in dynamics, and they demonstrate the application of the new method developed in chapter 4.

The final two chapters of this thesis detail some of my work on scientific methods and software development that has been instrumental to conducting the research contained within this dissertation. In chapter 6, I present a heuristic I developed to guide the time and effort expended in the development of scientific software. I discuss the practical application of this heuristic and demonstrate how, through its application, I have contributed to a large range of widely used software tools developed in our research field. Chapter 7 focuses on `freud`, a tool for analyzing particle simulations of which I have been a principle architect. This chapter discusses the various analysis methods in `freud` and the core infrastructure around which these methods are built.

Finally, chapter 8 summarizes each of these results in the overall context of this thesis. I close with suggestions for next steps for these studies and future directions for the field in light of the results presented here. Chapters 4, 6 and 7 are reproduced from published manuscripts [33–35], while chapters 2 and 5 are adapted from manuscripts about to be submitted. Chapter 3 contains elements of a published article [27] as well as some as yet unpublished results.

CHAPTER 2

Crystallization of Colloidal Truncated Tetrahedra Driven by Entropic Forces

This chapter is reproduced from Gong*, Z., Ramasubramani*, V., Morin, A., Lawton, P., Hueckel, T., Glotzer, S. C. & Sacanna, S. Crystallization of colloidal truncated tetrahedra driven by entropic forces. *In Preparation.* (*these authors contributed equally to this work). My contribution to this work was the simulations predicting the assembly of truncated tetrahedra under various conditions and the thermodynamic calculations used to explain why different polymorphs form under different conditions.

2.1 Introduction

Onsager's groundbreaking work on the isotropic to nematic transition of hard rods demonstrated the ordering power of entropy, and in particular, shape [18, 37, 38]. Although his work specifically examined the effect of shape on the partitioning of entropy, subsequent studies of entropic ordering transitions in the following decades focused on simpler entropic phenomena like depletion interactions [37, 39], in large part due to the difficulty of synthesizing anisotropic particles in the lab. In recent years, advances in synthetic techniques have begun overcoming these limitations, precipitating an explosion in the availability of particles with previously unattainable shapes [19, 40–42].

Despite these successes, experimental examples of structure assembly driven by shape alone remain few and far between. The principal limitation in modern syntheses is that at the length scales where nontrivial shapes, e.g. polyhedra, can currently be synthesized, interactions other than pure hard core repulsion usually cannot be screened sufficiently to completely remove their impact on self-assembly. While the shapes of sub-$100\,\mathrm{nm}$ particles can be finely controlled through a number of synthetic methods, enthalpic interactions at this scale are nearly impossible to screen to a sufficient degree, so they inevitably wash out

entropic interactions on the order of k_BT [13, 43–45]. Conversely, interactions between larger particles on the $0.1\,\mu\mathrm{m}$ to $1\,\mu\mathrm{m}$ scale can be screened more effectively, but precise control over nontrivial shapes is difficult to attain. Therefore, the most common syntheses are still of simple shapes like spheres and cubes [12, 46, 47].

Recent simulation results provide strong motivation to surmount these limitations. While Onsager's work was followed by many theoretical and simulation studies predicting various entropy-driven phase transitions for isotropic particles [1, 39, 48–52], recent simulations provide striking examples of the diverse array of structures that anisotropic particles can self-assemble into, even in the absence of other interactions [15, 53]. Moreover, these works show that the entropic driving forces for such assemblies can be finely tuned, both by exploiting traditional entropic interactions like the depletion force and by fine-grained control over particle shape [12, 54, 55]. One family of shapes now known to exhibit exceptionally rich shape-driven entropic assembly is the family of truncated tetrahedra, a continuous one-parameter family of shapes ranging from the tetrahedron to the octahedron [15]. Not only have hard truncated tetrahedra been shown to assemble a wide range of crystal structures, their strong directional interactions indicate the ease by which assemblies may be finely tuned by small modifications of particle shape or by introducing additional entropic interactions [56]. However, experimental studies of this family of shapes have invariably exhibited deviations from the ideal hard particle behavior due to the presence of attractive forces, typically resulting in dense packings or strong competition between structures [13, 44, 45].

In this chapter, we report the first synthesis of micron-scale hard truncated tetrahedra. Building on our previously introduced colloidal fusion technique for creating tetrahedral particles [57], we show how precisely tuning the degree of cross-linking and the deformation of liquid core clusters affords us precise control over the degree of truncation and surface curvature of the tetrahedra. The tunability and scalability of the synthetic protocol allow us to harvest a large number of uniform micron-sized colloidal tetrahedra We focus on the entropy-driven assembly of Archimedean truncated tetrahedra, showing that these particles can be assembled into a cubic diamond crystal in this manner, and that the introduction of depletion interactions and modulation of particle convexity can readily bias structure formation. In the process, we use these results to suggest possible ways in which systems may be engineered to prefer specific assembly behaviors. Altogether, our results demonstrate the efficacy of shape-based strategies for engineering target structures [58], and generalizing our synthetic methods to other polyhedra promises to pave the way to effectively exploit shape-driven entropic assembly to achieve desired complex crystal structures, such as a photonic colloidal diamond crystal [59–61].

2.2 Results and discussion

2.2.1 Synthesis of colloidal tetrahedra

Colloidal tetrahedra are synthesized in bulk by using a modified version of the synthetic pathway introduced in ref. 57: *colloidal fusion*. Figure 2.1a illustrates the principle of colloidal fusion, in which tetrahedral clusters of positively charged polystyrene (PS) satellites aggregate around a negatively charged 3-(trimethoxysilyl)propyl methacrylate (TPM) core via polymer bridging flocculation [57] to form liquid core clusters (LCCs). The satellite/core size ratio is carefully tuned to ~ 2 to favor the tetrahedral coordination of satellites around each core [62]. Upon swelling of the satellite PS particles by an organic solvent, core particles deform to fill the interstices, taking the shape of tetrahedra, at which point they are solidified and the satellites are dissolved and removed (fig. 2.1a). This scalable synthesis reliably yields large amounts of monodisperse tetrahedral particles as illustrated by the scanning electron microscope (SEM) image in fig. 2.1b. Critically, we modify the droplet synthesis process in [57] to yield a considerably cross-linked siloxane network (see section 2.4). This cross-linking of liquid cores alters the deformation dynamics of LCCs, resulting in colloidal tetrahedra with truncated vertices, see the zoomed-in SEM image of fig. 2.1b. Therefore, our synthetic protocol offers two independent avenues for tuning the particle shape: the vertex truncation through the degree of cross-linking of liquid core, and the face curvature through the deformation degree of satellite particles. Each row of fig. 2.1c shows a series of particles synthesized with TPM cores of increasing cross-linking, see section 2.4. The slightly different deformation paths presumably originate from an increase in the TPM viscosity, as surface tension measurements show barely any change with cross-linking (see fig. A.6).

Meanwhile, each column in fig. 2.1c shows tetrahedra at a different point in time during the deformation step shown in fig. 2.1c: moving right in the figure corresponds to longer deformations. When no tetrahydrofuran (THF) is added, the PS satellites remain solid and compress the liquid core, which extrudes through the interstices between the satellites to form tetrahedra with relatively flat truncated tips and concave faces set by the balance of surface tensions. The addition of THF turns the satellites into a viscous liquid which triggers the deformation of the cluster towards the new equilibrium state: a core-shell particle (see fig. A.4). Along this deformation path, particles evolve by reducing the truncation of their tips while changing the curvature of their faces from concave to flat to convex. These shapes exists on a continuum of truncated tetrahedra, for which we employ a parameterization $0 \leq t \leq 1$, where $t = 0$ is a tetrahedron and $t = 1$ is an octahedron (see fig. 2.1c, and see section 2.4 for more detail).

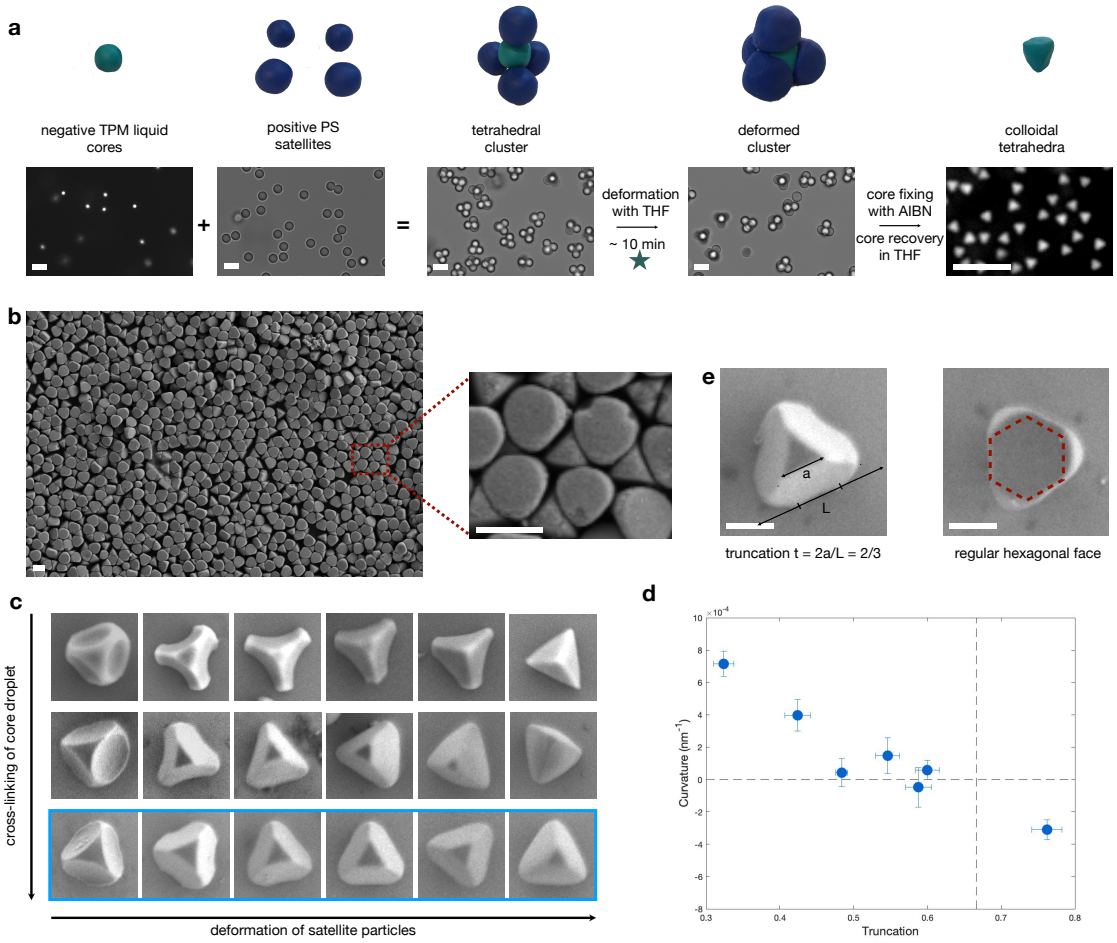

Figure 2.1 | **Synthesis of truncated colloidal tetrahedra. (a)** Illustration of the principle of the synthesis. Spherical core and satellite particles are mixed to form tetrahedral clusters composed of four satellites surrounding a core. Deforming these clusters, solidifying the core and removing the satellites yield tetrahedral particles. Optical microscope images on the lower row showing the process of colloidal fusion yielding colloidal tetrahedra. Liquid polymeric cores are mixed with solid polystyrene satellites to form tetrahedral clusters which are further deformed by adding tetrahydrofuran. Cores are fixed by radical polymerization initiated by 2,2'-azobis(2-methylpropionitrile) prior to dissolution of the satellites in pure THF. Scale bars: $5\,\mu\mathrm{m}$. **(b)** Scanning electron image of the recovered colloidal tetrahedra. Scale bars: $1\,\mu\mathrm{m}$. **(c)** Colloidal tetrahedra with a variety of truncations and face curvatures are obtained by separately tuning the cross-linking of liquid core, and the deformation of satellite particles. From top to bottom, each row corresponds to a synthesis with core droplets of increasing cross-linking. From left to right, the deformation of the satellites in THF evolves for longer times. The left side corresponds to $0\,\mathrm{min}$, and the right to $60\,\mathrm{min}$. **(d)** The face curvature of colloidal tetrahedra typically decreases with increasing truncation. The vertical dashed line indicates the Archimedean truncated tetrahedron (ATT), while the horizontal line indicates zero curvature. Blue dots correspond to the third row in (d). **(e)** High magnification scanning electron images reveal the Archimedean nature of the truncated colloidal tetrahedra which display regular hexagonal facets corresponding to a truncation $t = 2/3$. Scale bars: $0.5\,\mu\mathrm{m}$.

This evolution is quantified in fig. 2.1d (as detailed in Methods and fig. A.7) and indicates that colloidal tetrahedra in fig. 2.1c are out-of-equilibrium intermediates, captured along

the path from a tetrahedral cluster with solid satellites to a fully liquid core-shell particle. To further ascertain this scenario, we checked that the swelling of PS particles due to THF absorption is fast ($\sim 1\,\mathrm{min}$, see fig. A.5). Therefore, the changes in the satellites' properties occur and end at an early stage of the slow, $\sim 10\,\mathrm{min}$ long, deformation. These two independent avenues for tuning the particle shape dramatically expand the morphology library of experimentally achievable tetrahedral colloids. They provide a unique opportunity to investigate experimentally how shape directs self-assembly in the presence of thermal fluctuations and to test simulation predictions.

2.2.2 Assembly of 3D cubic diamond crystal

For the purposes of assembly, we tune the deformation dynamics of LCCs to obtain Archimedean truncated tetrahedra [65], truncated tetrahedra with flat, nearly regular hexagonal facets. The ATT corresponds to the truncation parameter $t = 2/3$, as shown in fig. 2.1e, and to our knowledge this synthesis is the first realization of ATTs at the nano-/colloidal scale. ATTs are of particular interest because simulations have shown that hard ATTs assemble into the CD structure, but pack densely into the α-arsenic structure [15]. Therefore, we would expect that properly screened colloidal ATTs would assemble into CD, while the strong van der Waals attractions in poorly screened systems would result either in α-arsenic or in a kinetically arrested mixture.

To induce self-assembly, colloidal ATTs are suspended in a toluene-based solvent and allowed to sediment within an upright glass tube, see fig. 2.2a and Methods. The colloids and the solvent have matched optical indices, which allows for 3D visualization of the sample by scanning confocal microscopy. In this non-polar solvent, particles are charged and are kept far from each other by electrostatic repulsion [66]. Accordingly, fig. 2.2b shows that particles at the bottom of the sediment form a rotator phase with hexagonal translational order but no orientational order. Particles are spaced by about $2\,\mu\mathrm{m}$ and free to undergo Brownian rotational diffusion.

Particles can be further crowded by adding salt to the suspension, which screens the particle charges and reduces the repulsion range. We use didodecyldimethylammonium bromide (DDAB) and initially suspend the particles in $200\,\mathrm{mM}$ DDAB without sealing the glass tube. Figure 2.2c shows the time evolution of the system as toluene slowly evaporates, increasing the salt concentration in the suspension. Confocal microscopy reveals that particle orientations become strongly anti-correlated (fig. 2.2c stage (iii)), signaling the start of crystallization, but the aggregation of the particles eventually prevents the emergence of long-ranged order (stage (iv)). This behavior suggests that within some optimal range of salt

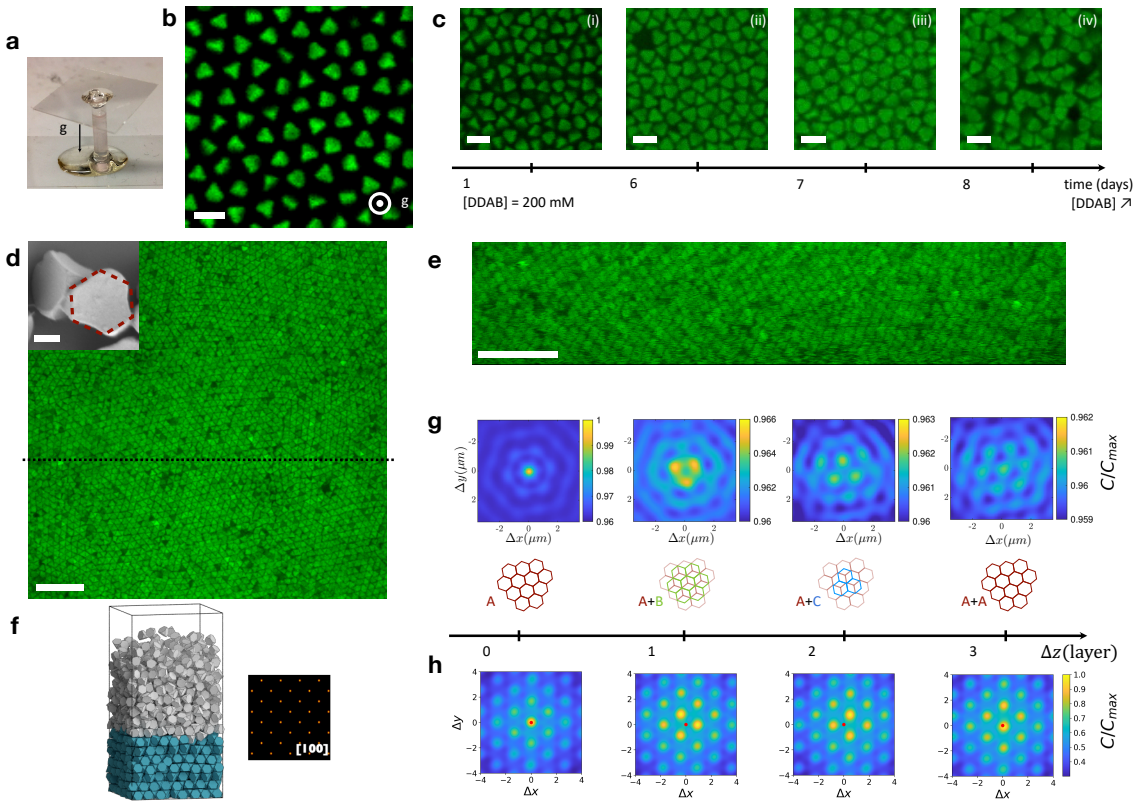

Figure 2.2 | **Self-assembly of truncated tetrahedra into cubic diamond. (a)** Particles dispersed in an index-matched solvent are left to sediment in $2.4\,\mathrm{mm}$-diameter NMR tubes. **(b)** In $0.5\,\mathrm{wt}\%$ OLOA toluene, colloidal tetrahedra form a rotator phase with positional order but no orientational order revealed by confocal microscopy. Scale bars: $2\,\mu\mathrm{m}$. **(c)** Particles in an unsealed tube get closer to one another over time as the solvent evaporates. Correlations in orientation build up (stages (ii) and (iii)) before particles aggregate (stage (iv)). Initial salt concentration: $200\,\mathrm{mM}$ didodecyldimethylammonium bromide (DDAB). Confocal microscope images. Scale bars: $2\,\mu\mathrm{m}$. **(d)** $75 \times 75\,\mu\mathrm{m}^2$ horizontal confocal slice of a crystal formed by the sedimentation of colloidal ATTs in a sealed tube dispersed in $0.5\,\mathrm{wt}\%$ OLOA toluene with $250\,\mathrm{mM}$ DDAB. Scale bar: $10\,\mu\mathrm{m}$. Note that due to the difficulty of synthesizing perfectly flat particles, these are slightly concave (as shown in the inset SEM image, Scale bar: 400 nm), but the same results hold for a small range of concavities indicating that up to a point these effectively behave like flat particles. We discuss the effects of convexity further in section 2.2.4. **(e)** $14\,\mu\mathrm{m}$ high vertical confocal cross-section through the dashed line of the crystal in (d) revealing 3D positional order. Scale bar: $10\,\mu\mathrm{m}$. **(f)** Simulations of hard ATTs sedimenting in a gravitational field show a cubic diamond (CD) crystal forming near the bottom (blue), with a fluid phase (gray) higher up where the pressure is lower due to fewer layers above. Particles with local CD are identified using OVITO [63, 64]. **(g)** The intensity autocorrelation function reveals an ABCA... stacking, corresponding to subsequent layers translated relative to one another as sketched below each panel and indicating the formation of CD. **(h)** For comparison with experiment, the positional autocorrelation function is computed from the simulation snapshot in (f). To generate an analogous measurement, rather than computing the autocorrelation of a discrete set of positions, positions are "smeared out" by placing a Gaussian at each particle center and summing the result prior to the autocorrelation calculation.

concentrations, the electrostatic and van der Waals interactions are sufficiently well-balanced to allow particles to order solely due to hard core excluded volume interactions – and thus entropy – as seen in fig. 2.2c stage (iii).

Indeed, we observe 3D crystallization by dispersing colloidal ATTs with slightly concave faces in a sealed tube containing a $250\,\mathrm{mM}$ DDAB solution. After equilibration of the sediment, we recorded vertical stacks of confocal microscope images to investigate the structure of the 3D assembly. Figure 2.2d-e shows horizontal and vertical slices through the sediment, indicating hexagonal symmetry in the horizontal slice and a less clear but still evident ordering through the vertical cross-section. Within each layer, ATTs form rings where the centers of mass are positioned like the carbon atoms in a cyclohexane molecule in its chair conformation. This stacking of hexagonal layers is characteristic of the (111) plane of cubic diamond CD, a structure that consists of an ABCA... stacking of these hexagonal planes (see also fig. 2.3a).

Since ordered stacking is evident in the alignment between hexagonal layers, we characterize the three-dimensional structure of the sediment by computing the autocorrelation of the fluorescence intensity $C(\Delta x, \Delta y, \Delta z)$, see section 2.4. Figure 2.2g shows $C(\Delta x, \Delta y)$ for $\Delta z = 0,\ 1,\ 2\,\mathrm{and}\,3$ layers. The six clear peaks related to the in-plane hexagonal symmetry at $\Delta z = 0$ in the autocorrelation indicate the presence of a dominant crystal grain throughout the entire sediment, and the 3-layer periodicity in the vertical direction evidences a CD structure displaying an ABCA... stacking of hexagonal planes. This global measurement confirms that the sediment in fig. 2.2d-e is a CD single crystal.

This agreement with previous hard particle Monte Carlo (HPMC) simulations indicates that this assembly is entropically driven, and the absence of α-arsenic suggests that we have successfully screened out interactions, unlike in previous sedimentation-driven assemblies where the observed structures were primarily dense packings [13]. The fact that subsequent layers adopted a variety of relative positions with equivalent electrostatic contributions (see fig. A.3), along with the recognition that nothing requires face-to-face alignment of hexagonal facets, shows that electrostatics are not responsible for this assembly. To understand the role of gravity in this system, we perform HPMC simulations in a grvaitational field (fig. 2.2f). We find that CD is indeed the preferred assembled structure, even in the presence of gravity. Figure 2.2h shows the positional autocorrelation function of the layers in the assembled structure in simulation aligned with the experimental results in fig. 2.2g, verifying that it is indeed the same structure. To decisively characterize the structure, we computed diffraction patterns along both the (111) plane (6-fold symmetry) and the (100) plane – the latter of which corresponds to the 4-fold symmetry axis customarily shown for CD – to verify the identity of the structure (fig. A.10); additionally, particles are colored in fig. 2.2f using OVITO's diamond identifier [63, 64].

Notably, we observe a dominant CD crystal rather than an even mix of CD and the HD polymorph (also known as Lonsdaleite). The distinction between these two structures

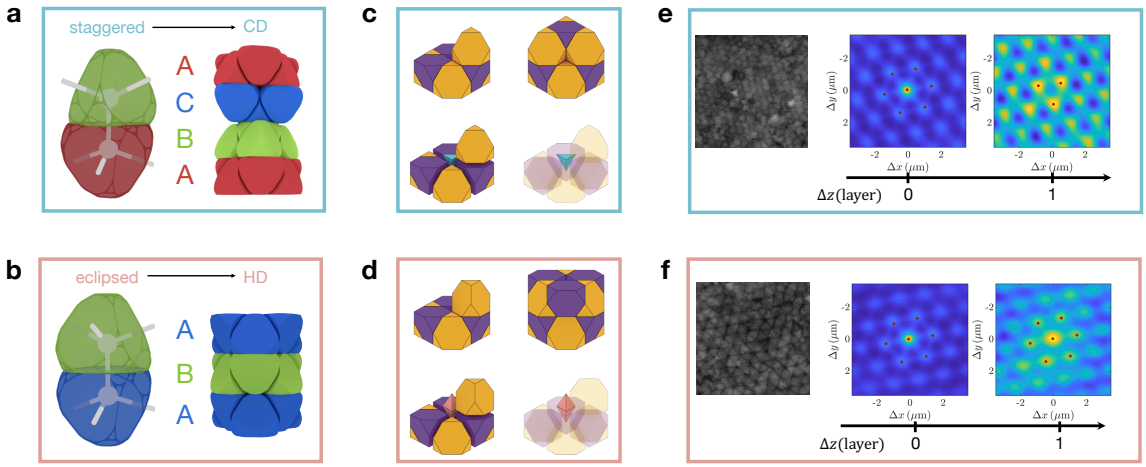

Figure 2.3 | **Comparison of CD and hexagonal diamond (HD) structures (a)** Illustration of a pair of tetrahedra in a staggered conformation (left). Staggered interlayer conformations lead to the formation of a CD with an ABCA... sequence of stacked layers (right). **(b)** Illustration of tetrahedra in an eclipsed conformation (left). Eclipsed interlayer conformations lead to the formation of a HD with an ABA... sequence of stacked layers (right). **(c)** When computing the intensity autocorrelation in a patch of staggered tetrahedra from the sample in fig. 2.2d (left), we see the expected single dominant peak for $\Delta z = 0$ (middle) and the characteristic triangular pattern for $\Delta z = 1$. **(d)** Conversely, the intensity autocorrelation in a patch of eclipsed tetrahedra from the same sample shows hexagonal rings with holes in the eclipsed case. Scale bar: $2\,\mu$m. **(e)** ATTs in a staggered conformation have tetrahedral interstices (shown in blue). **(f)** ATTs in an eclipsed conformation have triangular bipyramidal interstices (shown in pink). These interstices can be viewed as the merging of two aligned tetrahedral interstices like those present in staggered conformations.

is well-documented in the literature [67, 68]: particles in CD all exhibit local *staggered* contacts (fig. 2.3a, left), leading to a "chair" conformation in hexagonal rings and an ABCA... stacking along the (111) plane (fig. 2.3a, right), while 1 in 4 contacts in HD is *eclipsed* (fig. 2.3b, left), leading to a "boat" conformation and an ABA... stacking (fig. 2.3b, right). This distinction is analogous to the well-known crystal structures of spheres, face-centered cubic (FCC) and hexagonal-close packed (HCP): indeed, our CD and HD are the decorations of an FCC and an HCP lattice, respectively, with a pair of ATTs pointing in opposite directions (see fig. A.1). The fact that our particles are ATTs provides additional distinguishing characteristics between the two structures due to the different alignments of interstices and contacts between truncated tips that would not be present in systems of spheres (fig. 2.3c-d).

Due to the close similarity in their free energies (see fig. A.9), CD and HD are very difficult to synthesize independently of one another [68], again similar to FCC and HCP for spheres. To gain deeper insight into the microstructure of our sediment, we compute the fluorescent intensity within small, 5 layer-high volumes in the sediment in fig. 2.2d, where a blurry image suggests a staggered stacking (fig. 2.3e, left) and discernible hexagonal rings

indicate eclipsed stacking due to perfectly aligned particle faces between close-packed layers (fig. 2.3f, left). The results show that both staggered (fig. 2.3e, right) and eclipsed (fig. 2.3f, right) stackings are present; however, we note that the global intensity autocorrelation in fig. 2.2g characteristic of CD structure indicates that local HD stacking faults are rare. Our simulations show that the primary effect of gravity is to facilitate heterogeneous nucleation near the wall, followed by an epitaxial growth process with a preferred direction.

The fact that growth is along the (111) plane, which corresponds to a simple sliding mode that can switch between CD and HD, is likely responsible at least in part for the presence of HD defects. However, this growth process is evidently slow enough that most such defects can anneal out, resulting in a largely monocrystalline structure rather than a highly mixed set of CD and HD grains. This layer-by-layer growth process is also likely responsible for the lack of α-arsenic in our sediment. Although CD is the preferred assembly structure, the minimum free energy structure at sufficiently high packing fractions is the dense packing, α-arsenic, but the formation of CD from hexagonal planes at the lower densities at early stages of the sedimentation prohibits later rearrangement to α-arsenic due to the geometric barriers that emerge as the CD structure is compressed by the formation of higher layers [69]. These results indicate that, without engineering any energetic interactions, we have achieved an entropy- and shape-driven assembly of ATTs into a CD structure.

2.2.3 Modifying structure formation using depletion

The presence of some eclipsed stacking in the sediment suggests the possibility of engineering HD assembly. For ATTs, the mirror symmetry of the eclipsed conformation results in aligned interstices and additional contacts between truncated tips (see fig. 2.3d), a structural difference that we sought to exploit. The nature of these distinctions suggests a natural choice for biasing the assembly: the depletion interaction, an entropic interaction driven solely by excluded volume considerations. Depletion has been used to control self-assembly in various colloidal systems [12, 13, 54], and it is well-known that the primary effect in systems of polyhedral particles is to enhance face-to-face contacts in order to maximize the free volume available to depletants. Alth ough no previous studies of ATTs (or tetrahedra) in the presence depletion have been conducted, these results would suggest that the increased free volume available to depletants in HD relative to CD due to the interstitial alignment would lead to depletants favoring the ecplised stackings characteristic of HD.

To test this, ATTs are suspended in a mixture of silica nanoparticles and salt and sediment within a sealed capillary (see section 2.4). We first examine the assembly of a monolayer, where, similar to the sedimentation case, particles quickly order into hexagonal

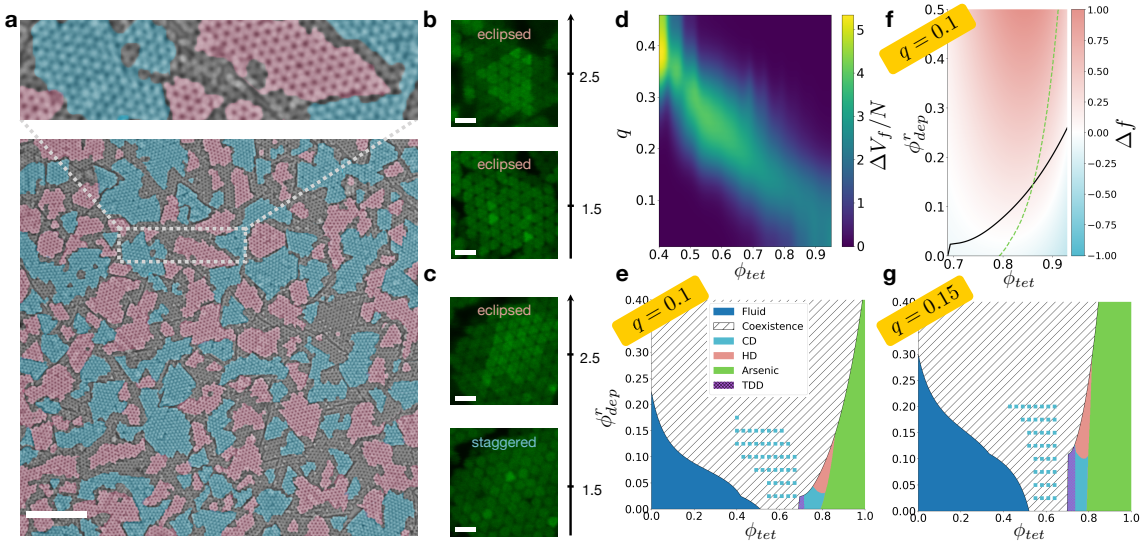

Figure 2.4 | **Depletion-driven formation of 2D hexagonal layers. (a)** Bottom: Even mixture of staggered and eclipsed interlayer conformations are obtained via depletion. False colours highlight the vertically staggered (blue) and eclipsed (pink) areas in this optical microscope image focusing between the first two layers. Scale bar: $20\,\mu\mathrm{m}$. Top: enlarged view of boxed region. **(b-c))** Uncorrelated vertical arrangement evidenced by confocal microscope imaging. Eclipsed stacking between the second and third layers occurs likewise above two first layers in eclipsed (h) or staggered (i) conformations. Scale bars: $2\,\mu\mathrm{m}$. **(d)** Relative free volumes between CD and HD as a function of the radius ratio $q = r_{dep}/r_{tet}$. HD always affords greater free volume to depletants, but the optimal depletant size at which the difference is maximized becomes larger as the system becomes less dense. **(e)** A phase diagram for depletants 10% the size of the ATTs shows a broad coexistence region induced by depletion (the y axis is $\phi^r_{dep} = v_{dep} n^r_{dep} = \frac{4}{3}\pi r^3_{dep} n^r_{dep}$). The region where the TDD structure is preferred is rendered purple with blue (i.e. CD-colored) hashes to indicate that it is structurally nearly identical to CD. The densest packing structure of α-arsenic becomes the favored structure at sufficiently low packing fractions to limit the range of densities where HD may be preferred. **(f)** The difference in free energy densities ($f = \Omega v_0/V$, where v_0 is the volume of a tetrahedron) between HD and CD for these depletants shows that beyond the coexistence boundary (solid black line) CD is typically the preferred structure. Most of the regions where HD has a lower free energy lie within the coexistence bounds, but a small region is present where HD is (slightly) preferred in the solid region below packing fractions where arsenic is preferred (dashed blue line). **(g)** A phase diagram for larger depletants ones shows a shrinking of the coexistence region, but no additional propensity for HD assembly.

two-dimensional sheets with staggered face-to-face contacts. The structure can be examined in great detail in both optical and electron microscopy images, see fig. A.3. We then performed experiments at a higher colloid density for which particles are still abundant after the completion of a monolayer, so that a second close-packed hexagonal layer grows above the first with staggered *intra* layer particle conformations. Differences in vertical arrangements are conveniently captured by focusing in between the two layers, where crystalline domains with different stacking mode are smoothly distinguished. Surprisingly, while small hexagonal motifs form and grow to span a complete layer as expected, fig. 2.4a-c reveals that these layers stack irregularly, resulting in a patchy landscape of interlayer conformations composed of an even mixture of staggered (blue) and eclipsed (pink) stackings

(see also fig. A.2)

HPMC simulations in the presence of depletants [70] confirm that the presence of depletants in this system leads to a mixed system. To understand these results, we turn to Free Volume Theory (FVT) [71], which considers a semigrand ensemble where the system is in equilibrium with a depletant bath with which it can exchange particles. The semigrand potential for such a system of N_c colloidal particles and N_d depletants in equilibrium with a depletant reservoir at a chemical potential of μ_d is:

$$\Omega(N_c, \mu_d, V, T) = F(N_c, N_d, V, T) - \mu_d N_d \tag{2.1}$$

The Widom insertion theorem provides a way to express the chemical potential μ_d as a function of the free volume $\langle V_{free} \rangle$ available to a depletant in a system of hard particles. The resulting expression provides a path between systems with and without depletion, allowing the use of thermodynamic integration to compute the total system free energy. The critical assumption of FVT is that $\langle V_{free} \rangle$ can be approximated by $\langle V_{free} \rangle_0$, the free volume in the limit $N_d = 0$, which results in the following ansatz for the resulting grand potential:

$$\Omega(N_c, N_d, V, T) \approx F(N_c, N_d = 0, V, T) - n^r_{dep} \langle V_{free} \rangle k_B T \tag{2.2}$$

where n^r_{dep} is the number density of depletants in the reservoir.

We use FVT to calculate the free energy density $f = \Omega v_0 / V$ (where v_0 is the colloid volume) in the fluid and in each structure of interest, then employ a common tangent construction to identify phase coexistence boundaries [54, 72] (see section 2.4). The first step is the calculation of $\langle V_{free} \rangle$ (see section 2.4) in various structures; fig. 2.4d shows the difference in free volumes between HD and CD for different depletant sizes (parameterized by the radius ratio $q = r_{dep}/r_{tet}$ where r_{tet} is the circumsphere radius of the truncated tetrahedron) and at different packing fractions of ATTs. As expected, this number is always positive, and the depletant size at which $\Delta V_f = V_f^{hex} - V_f^{cub}$ is maximized is a decreasing function of the packing fraction, i.e. smaller depletants are necessary to differentiate between the two structures at higher packing fractions. Interestingly, we also see that the greatest differences occur for lower packing fractions, because at sufficiently high packing fractions of tetrahedra, the depletants are crowded out of even the larger interstitial spaces in HD. Nonetheless the general trend suggests since ΔV_f is always positive, increasing the number of depletants should always eventually lead to HD assembly as long as the required depletant packing fraction remains within physical bounds (i.e. below a total system packing fraction of 1).

The phase diagram in fig. 2.4e reveals a more subtle picture. Although HD has a

lower free energy than CD for a sufficiently high depletant concentration, these depletant concentrations also greatly widen the coexistence regime, making it much more difficult to form a single phase. The pairwise free energy comparison in fig. 2.4f clearly illustrates that most of the regions in the ϕ_{tet}/ϕ^r_{dep} phase space where HD is favored over CD fall well above the coexistence boundary (shown in black). Moreover, the phase diagram is a ground state (0 temperature) calculation, so the relatively small values of δf near or below the coexistence boundary translate to a high probability of coexistence in real (thermalized) systems even below the 0 temperature coexistence boundary.

The thermodynamic picture is further complicated by the presence of other phases; in particular, we also consider α-arsenic, the densest knowing packing of ATTs [15], and the tetragonal diamond derivative (TDD) family of structures, which are tetragonal variants of CD (with the ratio of lattice parameters $c/a < 1$) [73]. The $c/a = 1$ limit corresponds to CD, and indeed the preferred c/a ratio for ATTs is near unity (although it can vary dramatically for more truncated tetrahedra) [73]; while we tested a range of c/a ratios, in practice the only TDD that is ever the minimum free energy structure has $c/a = 0.95$, which is nearly identical to CD and is therefore shown in purple with CD-colored (blue) hatches in our phase diagram. Further, we note that while α-arsenic is the thermodynamically preferred structure at $\phi_{tet} \gtrapprox 0.8$ (also shown as a dashed blue line in fig. 2.4f), achieving this density effectively requires a fluid-solid transition to a different crystal at lower densities first, and solid-solid transitions to α-arsenic from either CD or HD are effectively geometrically forbidden due to differences in the local topology. Therefore, despite their presence in the phase diagrams, since the predicted TDD structure is effectively indistinguishable from CD and α-arsenic is unlikely to actually assemble, CD and HD are the only two structures that we expect to see in experiment.

Since fig. 2.4d suggests that increasing depletant size should make HD formation more favorable, we construct a phase diagram with larger depletants in fig. 2.4g, where we observe a narrowing of the coexistence regime while otherwise seeing the same general trends. As has been discussed in previous works, FVT tends to underestimate the critical depletant concentrations above which fluid-solid coexistence is observed [54, 74], which is consistent with the discrepancies we observe between simulation and experiment in fig. 2.4e and fig. 2.4g. This additional phase diagram highlights the same problem with coexistence: at higher depletant packing fractions for system packing fractions $\phi \geq 0.64$ (regions where free energy calculations indicate that HD should be favored), simulations stop showing a clear preference for CD assembly, instead showing competition between CD and HD that prevents either structure from forming cleanly.

Note that while fig. 2.4d suggests that larger depletants at lower packing fractions

would most strongly favor HD by maximizing the free volume difference between CD and HD, closer consideration of eq. (2.2) shows otherwise. Since the depletant free energy contribution $n^r_{dep}\langle V_{free}\rangle k_B T$ is proportional not only to $\langle V_{free}\rangle$but also to n^r_{dep}, and since maintaining the same $n^r_{dep}$for larger depletants would require $\phi^r_{dep} = v_{dep} n^r_{dep} > 1$ (because $v_{dep} \propto r^3_{dep}$), larger depletants have a much smaller absolute contribution to the free energies even though the relative depletant free energy difference between CD and HD becomes larger. In other words, because depletion is an entropic, numbers-driven effect, maintaining a comparable depletant driving force with larger depletants would require filling the system with more depletants than can physically fit; conversely, physically valid depletant densities lead to small depletant free energy differences between CD and HD despite large free volume differences (see fig. A.11 and fig. A.12). As a result, these calculations suggest that in fact biasing the assembly from cubic to hexagonal diamond requires finding a very fine balance between various system parameters that may not be feasible in experiment, explaining the observed random stacking of hexagonal layers of ATTs.

2.2.4 Shape-sensitive entropic bonds

As discussed in section 2.2.1, our modified fusion protocol affords previously unavailable control over truncation and face curvature. Although the effect of changing truncation has been well-studied by simulation in this shape family [15], the effect of changing convexity and rounding is far less well understood. However, it is known that rounding (i.e. convex face curvature) weakens direction entropic interactions for faceted particles [55, 75], suggesting that it should disrupt the assembly of CD in this system. Since few synthetic methods allow for precise control over surface curvature, we therefore close this article by investigating how changes in the building block curvature alter their assembly propensities.

The two independent control knobs of particle morphology enables the synthesis of particles with similar truncations but opposite curvatures, thereby providing an experimental platform to test the sensitivity of assembly to the particle face curvature. We focus on two batches of particles, one convex (curvature = $8.1 \times 10^{-4}\,\mathrm{nm}^{-1}$, truncation = 0.52) and one concave (curvature = $3.2 \times 10^{-4}\,\mathrm{nm}^{-1}$, truncation = 0.68) (see fig. 2.5a). In both cases, we attempt assembly both with and without depletion.

Our experiments show that even small deviations from flatness (less than $100\,\mathrm{nm}$, right) have a significant effect on particle assembly. In experiments both with and without depletion, we observe that concave particles form much more ordered phases that convex ones, as illustrated in optical images in fig. 2.5b and confocal images in fig. 2.5 c). These experiments hint at an asymmetric influence of face curvature over entropic bonding. Note

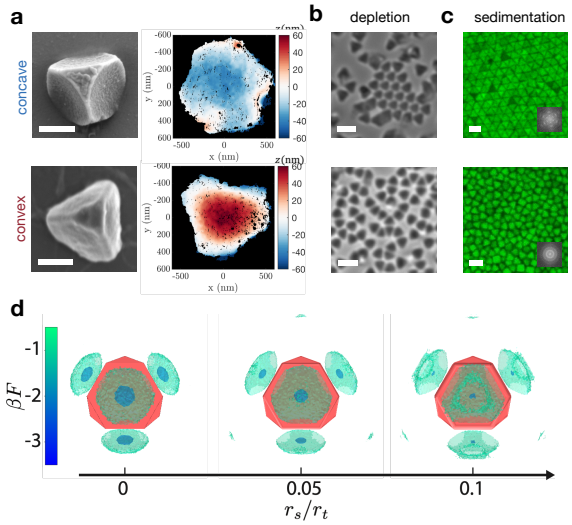

Figure 2.5 | **Tuning the shape entropy of colloidal tetrahedra.** **(a)** Scanning electron images (left) and elevation maps (right) of a tetrahedron with concave faces (top) and one with convex faces (bottom). Scale bars: $500\,\mathrm{nm}$. **(b)** Depletion-driven assembly of the concave (top) and convex (bottom) tetrahedra of (c). Concave tetrahedra display favoured tip to face contact. Convex tetrahedra do not crystallize. Scale bars: $2\,\mu\mathrm{m}$. **(c)** Assembly of the concave (top) and convex (bottom) tetrahedra of (c) by sedimentation. Concave tetrahedra successfully crystallize. Convex tetrahedra fail to crystallize. Insets: the Fourier transforms of the images indicate the hexagonal order (top) and fluid structure (bottom). Scale bars: $2\,\mu\mathrm{m}$. **(d)** Potentials of mean force and torque (PMFTs) of ATTs with increased rounding show increasingly delocalized entropic bonds. A rounding ratius 10% the particle's circumsphere radius is sufficient to almost completely eliminate the $-3k_BT$ energy well and leads to comparable bonding strengths all along the hexagonal faces rather than focusing it in the center.

that the primary effect of rounding on the CD/HD competition is to bring their free energies closer together by virtue of reducing the free volume difference (see fig. A.14), making it less likely that we would see a single grain in a crystal, but this does not explain the lack of crystallization altogether.

To understand why crystallization of convex tetrahedra fails to occur, we computed PMFTs (see section 2.4) from simulations of tetrahedra with various degrees of rounding, see fig. 2.5d. The PMFT measures the ensemble-averaged free energy surface around a particle, and as prior literature would suggest [55], the PMFT clearly shows that the $-3.0\,k_BT$ energy well at the centers of the four hexagonal faces becomes smaller as the amount of rounding increases. Furthermore, we see that the $-1.0\,k_BT$ well spreads to the truncated triangular faces, indicating that as rounding increases the strong preference for contacts between hexagonal faces becomes weaker and the alignment of tetrahedra along their truncated tips becomes stronger.

The reduction in the effective face to face attraction required for forming diamond explains why the convex particles. The fact that concave particles still assemble readily

indicates that similar disruption of emergent direction attractions does not occur. These different behaviours could also be explained by viewing tetrahedra with convex faces as intermediates between Archimedean truncated tetrahedra and spheres. As such, they experience an 'identity crisis' [76], failing to crystallize into either CD or FCC structures. Meanwhile, the assembly behavior of concave particles seems largely determined by the (convex) volume swept out by these shapes, with concavity being at most a secondary concern.

This result indicates that surface curvature is an important consideration in the synthesis of colloidal particles for self-assembly into higher-order structures, and that its effects are more complex than a simple pursuit of zero curvature surfaces. Our experimental findings highlight the crucial role played by surface curvature in entropy-driven self-assembly, a feature rarely captured in prior works. Moreover, since convexity is a far more common product of synthetic methods than concavity, our result suggests the importance of developing methods capable of minimizing the rounding of polyhedral particles intended for self-assembly.

2.3 Conclusion

In this chapter, we demonstrated how the colloidal fusion process can be exploited to synthesize precisely truncated tetrahedra, and we have engineered the entropy-driven self-assembly of colloidal Archimedean truncated tetrahedra into a cubic diamond crystal via sedimentation. We have shown that depletion forces drive the crystallization of ATTs into a random stacking of hexagonal planes, but we suggest possible strategies for engineering systems based on our observations of the behavior of ATTs in the presence of depletion. We further show that this crystallization is highly sensitive to the curvature of the particle faces: while truncated tetrahedra with concave faces crystallize, particles with convex faces remain in a disordered state. These results demonstrate the sensitivity of entropic assembly to minute changes, and show nontrivial structures may be assembled by fine-tuning these features. These results provide important guidance in the future design of building blocks to target specific microstructures.

2.4 Methods

2.4.1 Synthesis of truncated tetrahedra

Synthesis of the building blocks Monodisperse 3-(trimethoxysilyl)propyl methacrylate (TPM)-based emulsion droplets and solid polystyrene (PS) spheres are used to assemble liquid core clusters (LCCs). A detailed description of the synthetic protocols of the two species and assembly of LCCs can be found in [57].

Here, we focus on the modified synthetic route of the TPM-based droplets that we use to build highly cross-linked siloxane networks inside the droplets. Specifically, $100\,\mu\mathrm{L}$ of NH_3 ($28\,\mathrm{wt}\%$) is added to $100\,\mathrm{mL}$ of deionized water, followed by the addition of $200\,\mu\mathrm{L}$ of 3-(trimethoxysilyl)propyl methacrylate (TPM). After being kept under mild magnetic stirring for $2\,\mathrm{h}$, the monodisperse emulsion droplets with a diameter around $1\,\mu\mathrm{m}$ can be checked under optical microscopy, respectively. $400\,\mu\mathrm{L}$ of NH_3 ($28\,\mathrm{wt}\%$) is added to the suspension to speed up the condensation of silanol groups inside droplets without changing their size. The mixture is kept under mild stirring for another $2\,\mathrm{h}$, followed by two cycles of gentle centrifugation/re-suspension in deionized water, at which point it is ready for cluster assembly.

The positively charged PS spheres are made by seeded emulsion polymerization using 2,2'-Azobis(2-methylpropionamidine) dihydrochloride (AIBA, 97%, Sigma Aldrich) as the initiator. The seeded growth step is repeated twice to reach a final particle diameter of $2\,\mu\mathrm{m}$.

Truncated tetrahedra formation To assemble LCCs both silicone droplets and PS particles are first placed in $25\,\mathrm{mM}$ HCl solution to allow for poor wetting. The oil droplet suspension ($0.04\,\mathrm{wt}\%$) is then dripped into the PS suspension ($4\,\mathrm{wt}\%$) at a rate of $\sim 1\,\mathrm{mL\,min^{-1}}$ under mild stirring.

To separate clusters from excessive PS singlets, the mixture is suspended in an approximately $22\,\mathrm{wt}\%$ glycerol, $25\,\mathrm{mM}$ HCl solution and centrifuged at $3500\,g$ for $1\,\mathrm{h}$. The substitution of $25\,\mathrm{mM}$ HCl for water in making the glycerol suspension is crucial to ensure that the poor wetting of the oil droplet on PS is maintained during separation. The sediment of LCCs is then resuspended into $25\,\mathrm{mM}$ HCl solution.

In order to capture faceted tetrahedra, which are unstable intermediates (as noted in section 2.2.4, see also fig. A.4), we use a small amount of tetrahydrofuran (THF) and a long deformation time to extend the time window. This extended window enables quenching and fixing the intermediate, despite no noticeable change in the shape of the overall cluster, see fig. 2.1b. Typically, for faceted tetrahedra, THF solution ($40\,\mathrm{v./v.}$ in water) is dripped into the LCC suspension of $1\,\mathrm{wt}\%$ while vortexing to reach a final THF concentration of $22\,\mathrm{wt}\%$.

Deformation is allowed to take place over the course of $10\,\mathrm{min}$, followed by quenching with deionized water to reach a seven-fold dilution. To recover the colloidal tetrahedra, the liquid cores are first hardened by radical polymerization with 2,2'-Azobis(2-methylpropionitrile) (AIBN, $98\,\%$, from Sigma-Aldrich) at $80\,^{\circ}\mathrm{C}$ for $3\,\mathrm{h}$ and then the PS templates are dissolved in pure THF. After 8 to 10 washing/re-suspending cycles in pure THF, particles are gently transferred back into water with $1\,\mathrm{wt}\%$ F108 for assembly experiments.

For the deformation series shown in the top row of fig. 2.5a, a master batch of tetrahedral clusters were made with lightly cross-linked droplets (growth for $2\,\mathrm{h}$, no extra condensation in base). A small sample was taken out of the cluster suspension and polymerized directly. The dimpled cores recovered from this sample were demarcated as having zero deformation time. THF solution ($40\,\mathrm{v}\%$ in water) was then added to the master batch of clusters to reach a final THF concentration $22\,\mathrm{v}\%$. For every few minutes, a small sample was taken out, quenched with deionized water, and polymerized. This process was repeated until the deformed cores looked convex under an optical microscope. PS templates were removed after polymerization by dissolution in THF and morphologies of tetrahedral cores were examined via scanning electron microscope (SEM). The middle and bottom rows of fig. 2.5a correspond to repeats of the same set of experiments with clusters made of mildly cross-linked droplets (growth for $2\,\mathrm{h}$ in $7\,\mathrm{mM}$ ammonia, aged for $5\,\mathrm{h}$ in $37\,\mathrm{mM}$ ammonia) and heavily cross-linked droplets (growth for $2\,\mathrm{h}$ in $7\,\mathrm{mM}$ ammonia, aged for $12\,\mathrm{h}$ in $37\,\mathrm{mM}$ ammonia).

2.4.2 Sedimentation experiments

To study the self-assembly behaviour of sedimented colloidal tetrahedra we use toluene as the solvent because its refractive index closely matches that of colloidal tetrahedra. Particles are transferred into toluene with $0.5\,\mathrm{wt}\%$ OLOA-1200 (From Chevron) as a stabilizer, and didodecyldimethylammonium bromide ($98\,\%$, from Sigma-Aldrich) as a salt to decrease Debye screening length. Typically, a $60\,\mu\mathrm{L}$ particle suspension is injected into a glass sample chamber made of a cut NMR tube ($1.5\,\mathrm{cm}$ in height and $2.4\,\mathrm{mm}$ in diameter) and glued onto a thin microscopy slide. The inlet is then covered with a microscopy slide and sealed with Norland 63 Optical Adhesive Glue. Particles are left to sediment for several days to allow for crystallization. Normally, particles can be observed over $80\,\mu\mathrm{m}$ in z-direction. When crystallization occurs, the solid phase is in approximately the bottom $20\,\mu\mathrm{m}$.

2.4.3 Depletion experiments

In our depletion experiments we use Ludox HS-40 colloidal silica ($40\,\mathrm{wt}\%$ suspension in H_2O from Sigma-Aldrich) as the depletant. The depletant concentration is tuned to enable soft attraction and therefore crystallization. A typical depletant number density used in our experiments is $7.2 \times 10^{15}\,\mathrm{particle\,mL^{-1}}$. Specifically, $1\,\mu\mathrm{L}$ of $40\,\mathrm{wt}\%$ Ludox is first mixed with $10\,\mu\mathrm{L}$ of $5\,\mathrm{wt}\%$ Pluronic F108 solution and the mixture is vortexed for a few seconds to enable full coating of polymer on silica particles, followed by the addition of $19\,\mu\mathrm{L}$ of $0.1\,\mathrm{wt}\%$ colloidal tetrahedra suspended in $1\,\mathrm{wt}\%$ F108 solution. The suspension is then transferred into a flat $50 \times 2 \times 0.2\,\mathrm{mm}^3$ capillary from VitroCom, which is then sealed with Norland 61 Optical Adhesive Glue to a microscopy slide. We use a Leica DMI 3000 inverted optical microscope equipped with $100\,\mathrm{x}$ Leica oil objective and high-resolution CMOS camera (Hamamatsu ORCA Flash 4.0 CMOS) to image particles in bright field, and a Leica TCS Confocal Microscope to acquire fluorescent images. To better visualize particle assembly, a capillary with crystals formed by depletion is broken and dried in open air for a day. The broken capillary pieces were then gently rinsed with deionized water to remove extra salt and surfactant, and placed on conductive tape for observation by SEM.

2.4.4 Confocal imaging and analysis

Imaging A Leica TCS Confocal Microscope is used to study and characterize the crystal phase. The scanner was set to scan from the bottom to the top layer of the solid phase with a step size of $100\,\mathrm{nm}$. Confocal stacks are analyzed by ImageJ and autocorrelation measurements are performed using Matlab.

Autocorrelation measurements We identify the crystalline structure of the sediment in real space by measuring the three-dimensional autocorrelation $C(\Delta x, \Delta y, \Delta z)$ of the fluorescence intensity signal $I(x, y, z)$. Precisely, the intensity autocorrelation is defined as:

$$C(\Delta x, \Delta y, \Delta z) = \iiint dx\,dy\,dz\,[I(x, y, z) \times I(x + \Delta x, y + \Delta y, z + \Delta z)] \tag{2.3}$$

The actual measurement of C is performed using Matlab and uses its built-in fast Fourier transform (FFT) algorithm to speed up the calculation. The global analysis in fig. 2.2 is performed using the whole stack of volume $74.8 \times 74.8 \times 14\,\mu\mathrm{m}^3$. To investigate the local structures as in fig. 2.3, we measure C using eq. (2.3) with a selected subset of the total intensity with volume $14 \times 14 \times 3.7\,\mu\mathrm{m}^3$. We plot C normalized by its maximal value

$C(0, 0, 0)$.

2.4.5 Colloidal tetrahedron morphology

Truncation parameter t of colloidal tetrahedra is defined as twice the ratio of the length of triangular truncation plane to that of the edge of tetrahedron before truncation, see fig. A.7 (this is identical to the parameterization used in [15]). For each sample, its truncation parameter was determined as the average value of that of ten particles.

To measure the surface curvature of colloidal tetrahedra, high resolution SEM images of the same particle with a $5°$-$10°$ tilt are taken. Elevation maps are obtained by correlating the two images, which is done with MountainsMap, Digital Surf. From the elevation maps, characteristic curvatures of the faces are estimated by approximating the faces with spherical caps, see fig. A.7.

2.4.6 Computer Simulations

Simulations of the sedimentation- and depletion-driven assembly of truncated tetrahedra were carried out using HOOMD-blue [32, 77, 78] using the hard particle Monte Carlo (HPMC) [79] subpackage, which models all Archimedean truncated tetrahedra (ATTs) as hard particles. In simulations of sedimentation, the gravitational potential was simulated using a just-in-time (JIT)-compiled potential that imposes a one-dimensional linear gradient on the potential energy. The parameter controlling gravitational strength g was tuned to match the experimentally observed fraction of the sediment that crystallized. Depletion interactions were simulated using a semi-implicit depletant scheme that simulates a semi-grand ensemble in which hard particles (the tetrahedra) are held in osmotic equilibrium with a reservoir of mutually penetrable hard depletants [70]. To compute the positional autocorrelation functions in fig. 2.2h and fig. 2.3e-f, the `freud` library [33] was used to generate corresponding Gaussian blurs and SciPy's FFT module was used to efficiently compute the correlation function (zero-padded to ensure aperiodicity in x and y).

Simulations of rounded ATTs represent the shapes as the Minkowski sum of the core polyhedron and a sphere with the desired rounding radius. This is geometrically equivalent to sweeping the surface of the polyhedron with a sphere of that radius, and has the effect of bulging the faces while introducing rounding around the edges and vertices. Simulations were run for 1×10^7 Monte Carlo (MC) sweeps, and systems without shape convexity were found to crystallize within 3×10^6 sweeps. Potentials of mean force and torque (PMFTs) [56, 80] were calculated from these simulations using the `freud` library's PMFT

module [33]. The PMFTs were averaged over all trajectory frames after a 1×10^6 sweep equilibration.

Analysis makes extensive use NumPy [81], SciPy [82], pandas [83], and rowan [84]. Figures from theory and computation in this paper were generated using Matplotlib [85]. Data and workflows were managed with the signac framework [86, 87].

2.4.7 Free Volume Theory

Since our application of Free Volume Theory (FVT) follows the procedures and conventions established in [54] and [72], we provide only a brief summary here. Crystal free energies were calculated using the Einstein crystal method of Frenkel and Ladd for anisotropic particles [88–90]. Free energies for different crystal structures were computed for otherwise identical systems, ensuring that the free energy of the corresponding Einstein crystals are identical. Fluid free energies were computed by integration of the equation state, using the Widom insertion theorem to establish a limit of integration via calculation of the chemical potential at a particular packing fraction. The free volumes plotted in fig. 2.4d are computed using an MC volume integration based on the fraction of successful depletant insertions into a system of tetrahedra. Phase coexistence boundaries are found using the common tangent construction described in [72] (see fig. A.13 for examples).

CHAPTER 3

Predictive Modeling of Protein Stability and Assembly

3.1 Introduction

Predicting the assembly behavior and crystallization of proteins is of immense practical value to engineering new materials. Due to their incredible natural variety and the ease of modification, a diverse set of structures can be made from variations on the set of known proteins. In the therapeutic arena, proteins may be used as drug delivery vehicles or as part of gene therapies, and understanding the aggregation and dissociation behavior of proteins is critical to understanding diseases like amyloidosis and Alzheimer's that involve defective proteins [91, 92]. Protein catalysts, or enzymes, could be engineered to treat such disorders, or to improve industrial processes ranging from biofuel generation to food processing [91]. Polypeptides can also be used to direct the assembly of nanoparticles, a process with uses ranging from carbon sequestration to new energy technologies like solar energy conversion [93].

Just as protein folding is an intractable problem that is the subject of active research (despite astounding recent advances in predicting folded proteins [94], much work remains to elucidate the mechanics of the folding process), predictive modeling of protein assembly remains out of reach of our current theoretical and computational modeling capabilities. Coarse-grained modeling provides one popular way to close the gap between current capabilities and the relevant problem scale; as a result, improving protein models has been a driving force for the evolution of both top-down and bottom-up coarse-graining efforts over time [9, 95]. The most common models tend to either involve rigorous bottom-up coarse-graining via approximation of the potential of mean force (PMF), or top-down but still quite small-scale models that focus on collecting atoms into effective interaction beads to reduce the number of interaction sites while matching certain properties [7, 95]. However, the large scale of the problem and the number of interactions to consider makes the potential

of colloid-like approaches to minimal models attractive. Such models approach top-down coarse-graining at a much higher level, seeking to understand only the relevant features and use a very small set of interactions to capture only those features.

Although there is much debate over the validity of such a top-down approach [22, 28], its potential utility is unquestionable. Moreover, recent research efforts have shown that under the right circumstances such a model can indeed be sufficient, especially when truly dominant interactions that govern interprotein interactions can be identified [24–27]. Protein shape is one of the most common features that such a model must capture. The central challenge here is to determine a suitable coarse-graining resolution that captures the necessary length scales, particularly when additional interactions must also be included in the model, and this challenge typically dooms extremely coarse colloid-like models.

In this chapter, I aim to provide further insights into when and how far protein shape can be coarse-grained. I first discuss an experimental system of supercharged green fluorescent protein (GFP) and how it can be modeled by embedding point charges into a hard, nonconvex polyhedron. I show that this approach is sufficient to capture the stability properties of various assemblies of this protein, but that it is insufficient to assemble these proteins into a structure from a fluid. I then propose a means by which the lessons from this system may be applied to develop a machine learning model to discriminate between the role of different interactions in the assembly and crystallization of a particular protein. I develop a complete framework to collect data for, construct, and train such a model, and demonstrate its application. I close the chapter with some suggestions for potential future applications of this model.

3.2 Assembly of Supercharged GFP

This section was adapted from Simon, A. J., Zhou, Y., Ramasubramani, V., Glaser, J., Pothukuchy, A., Gollihar, J., Gerberich, J. C., Leggere, J. C., Morrow, B. R., Jung, C., Glotzer, S. C., Taylor, D. W. & Ellington, A. D. Supercharging enables organized assembly of synthetic biomolecules. *Nature Chemistry* **11,** 204–212. ISSN: 17554349 (Mar. 2019). Results are summarized for brevity, with most details of the experimental results omitted except those necessary to discuss the simulation results. My contribution to this work was the development of a new method for simulating these proteins and the analysis of the resulting simulations.

3.2.1 Introduction and Experimental Results

This work demonstrates a robust strategy for assembling well-defined quaternary structures of proteins. Our novel supercharged protein assembly (SuPrA) technique exploits monomeric GFP with excess acidic or basic amino acids. Although these results are demonstrated for a few variants, our primary focus is on the Ceru+32 and GFP-17 positive and negative proteins. Dynamic light scattering (DLS) shows $\sim 6\,\mathrm{nm}$ particles in solutions of like-charged monomers (e.g. only Ceru+32 or GFP-17) indicating that monomers do not form any complexes. These results change at higher salt concentrations, where DLS shows approximately $11\,\mathrm{nm}$ diameter particles in $100\,\mathrm{mM}$ to $300\,\mathrm{mM}$ NaCl solution and much larger $\sim 1300\,\mathrm{nm}$ diameter particles in $<50\,\mathrm{mM}$ NaCl solution.

Adding salt to a $50\,\mathrm{mM}$ NaCl solution causes a rapid shift in the size of the particles from $\sim 1300\,\mathrm{nm}$ to $11\,\mathrm{nm}$, indicating that the μm-sized particles are held together by relatively weak interactions between discrete, well-defined structures. These results suggest that electrostatically-mediated assembly is responsible for the larger particles present in the mixed solutions. Confocal microscopy shows that these larger particles are amorphous and lack any clear structure. Therefore, we focus on the smaller nm-scale structures. Negative stain electron miscroscopy (EM) shows that these particles are well-defined barrel-like protomers composed of two symmetric, 8-protein rings where the major axis of the protein was oriented along the axis of the barrel.

As will be discussed in section 3.2.2, we employ MC simulations to gain deeper insight into these rings, constructing candidate structures based on the 8-fold symmetry apparent in the EM data. Since most naturally occuring protein oligomers are highly symmetric, we imposed additional symmetry criteria upon these arrangements to limit our set of candidate structures. However, we observed no significant difference in the stability of these potential candidates, suggesting that our data was in some way incomplete.

We resolved this discrepancy using cryo-EM images, which showed that while the general structure of the 8-member ring was accurate, the true structure had D4 symmetry in which half the proteins were instead oriented perpendicular to the barrel axis, while the others were oriented at an angle (see fig. 3.1). We analyzed this structure and found four distinct contacts that were primarily implicated in the stability of the protomer that were composed of mutant residues in the supercharged variants. These contacts include two distinct types of contacts between proteins in the same ring and two other inter-ring contacts. However, a large number of the supercharged residues played no part in any contacts, suggesting that the large number of mutations provided the protein an ample variety of potential contacts from which to choose the most favorable. We further verified these results by testing additional supercharged GFP variants and found robust protomer assembly. These

results suggest that SuPrA is a simple and general technique for engineering proteins into larger molecular assemblies.

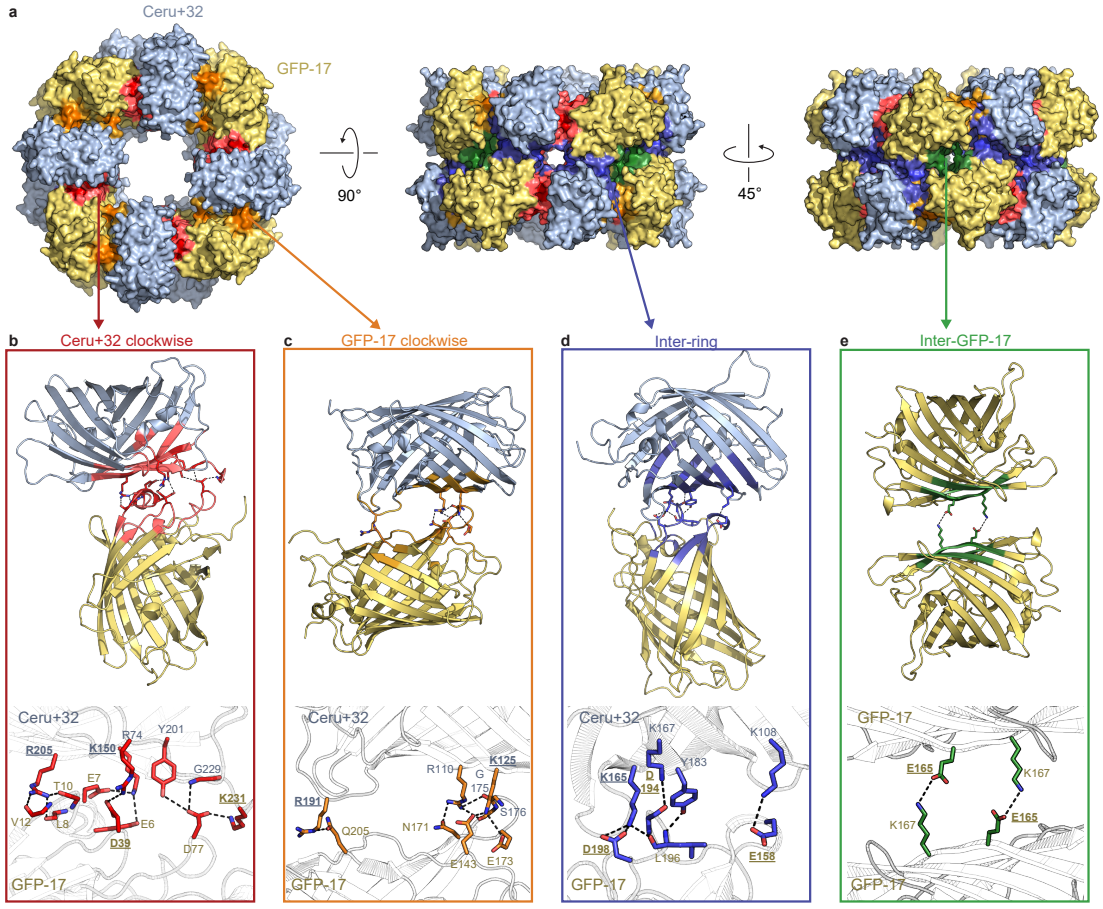

Figure 3.1 | **a**) Surface representation of the atomic model. Interfacial regions (within 7 Å of residues participating in inter-monomer interactions, as detected by PDBePISA [96]) are shaded red, orange, blue and green, respectively. **b–e**) Details of the four interfaces between Ceru+32 and GFP-17 in the protomer: 'Ceru+32 clockwise' interface (b); 'GFP-17 clockwise' interface (c); 'inter-ring' interface (d); 'inter-GFP-17' interface (e). Dotted lines correspond to hydrogen bonds. Ceru+32 and GFP-17 monomers are coloured light blue and gold, respectively. Underlined residues are the charged residues introduced to engineer the supercharged variants.

3.2.2 Modeling and Simulation

The primary goal of our simulations of this protein complex were to determine the true structure of the protomer and implicate specific contacts in its stability. We initially attempted to use all-atom molecular dynamics (MD) simulations to study this system of proteins, but these indicated that achieving protomer assembly would be impossible on reasonable time scales. In these simulations we observed performance of $\sim 3.395\,\mathrm{ns\,d^{-1}}$. For the

experimentally observed GFP diffusion coefficient of $87\,\mu\mathrm{m}^2\,\mathrm{s}^{-1}$ [97], the time required for a single GFP protein to diffuse $11\,\mathrm{nm}$ (the approximate protomer diameter) would be $\frac{(11\,\mathrm{nm})^2}{87\,\mu\mathrm{m}^2\,\mathrm{s}^{-1}}\frac{1\,\mathrm{d}}{3.395\,\mathrm{ns}} \approx 410\,\mathrm{d}$. This observation led us to pursue coarse-grained simulations focusing on protomer stabilization over atomistic assembly simulations.

To coarse-grain the system, we modeled candidate protomer structures by representing Ceru+32 and GFP-17 proteins as nonconvex polyhedra corresponding to their molecular surfaces. The position of each protein in the protomer is defined by the centroid of the atomic positions in the atomistic representation. To determine the orientation of each protein, we calculated the covariance matrix between the atomic positions of the protein at that site and a corresponding protein placed at a reference orientation. We used a singular value decomposition of this matrix to determine the rigid body rotation required to match the actual protomer protein's orientation [98]. We construct a coarse-grained protomer representation in which molecular surfaces are placed at these positions and orientations. Molecular surfaces are constructed for each of the proteins constituting the protomer using the MSMS tool[99] with a probe density of $1.0\,\mathrm{vertex}\,\mathrm{\AA}^{-1}$. We performed MC simulations [79] of protomers composed of molecular surfaces, which are treated as hard (i.e. non-overlapping) particles. Screened point charges are placed at every atomic site, and a Metropolis acceptance criterion is applied to the combined hard shape and charge potential.

Charges interact via the Yukawa potential:

$$V(r) = \left\{ \begin{array}{cc} \lambda_b Z_i Z_j \frac{\exp(-\kappa r)}{r} & r < r_{cut} \\ 0 & r < r_{cut} \end{array} \right\} \tag{3.1}$$

where κ is the inverse Debye screening length given by Debye-Hückel theory:

$$\kappa = \left(\frac{e^2 \sum_i c_i Z_i^2}{\epsilon_r \epsilon_0 k T} \right)^{1/2} \tag{3.2}$$

We hypothesized that the resolution of this HPMC+charge model would be sufficient to model the stability of these proteins. Critically, by using a MC approach, we could take advantage of acceleration techniques, namely cluster moves, to accelerate the convergence of these simulations to their equilibrium state in order to assess stability without necessarily considering the nature of the pathways. In particular, we employed a geometric cluster algorithm [100] that dramatically increased the time scales accessible to us.

Since the EM data suggested that all proteins were oriented along the protomer axis, we started constructing possible protomer structures with the initial assumption of 2^{16} total possible configurations (each protein either up or down). However, symmetry considerations eliminate nearly all of these candidates; if we consider only nearest-neighbor interactions

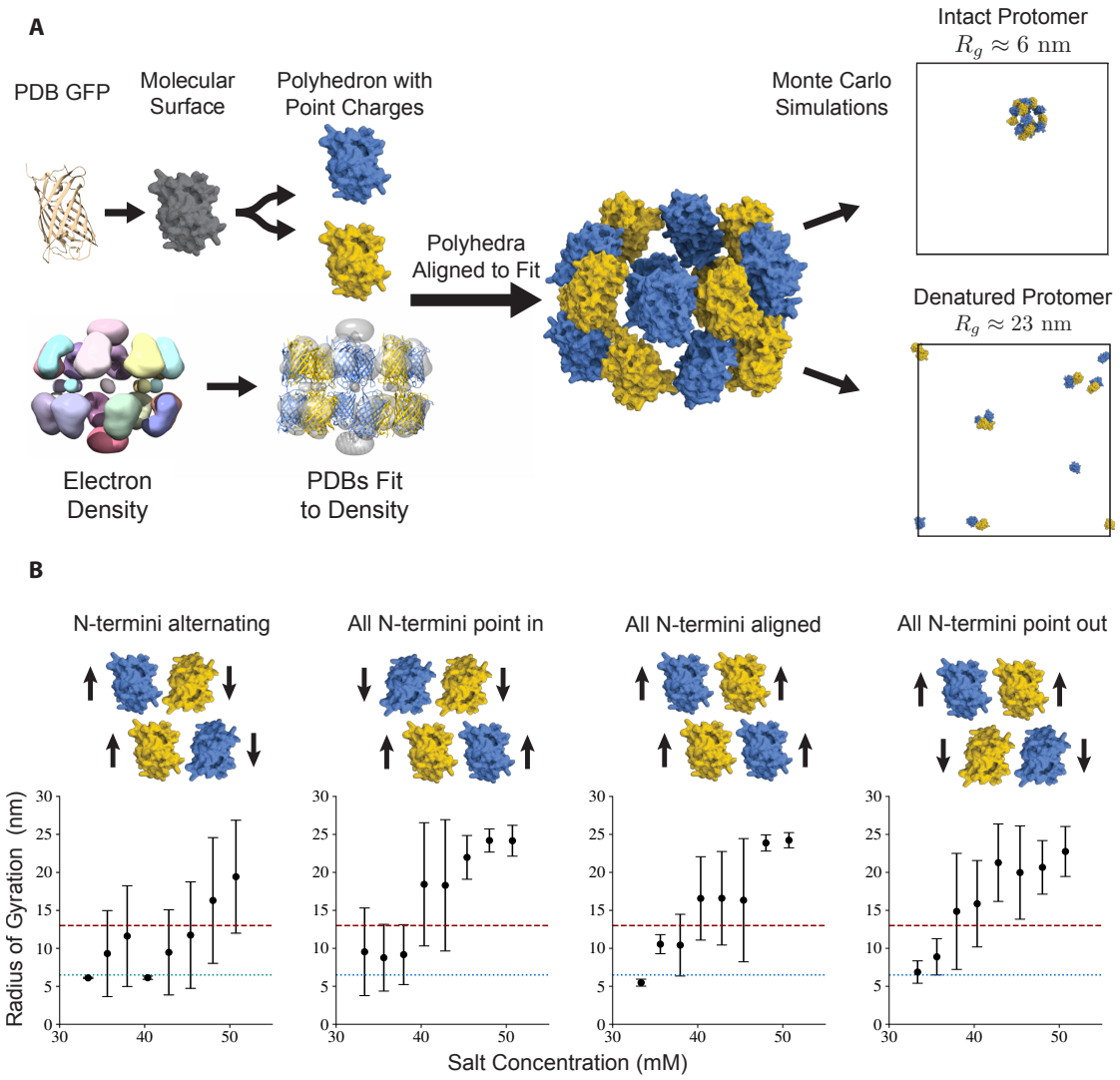

Figure 3.2 | **a**) Surface representation of the atomic model. Interfacial regions (within 7 Å of residues participating in inter-monomer interactions, as detected by PDBePISA [96]) are shaded red, orange, blue and green, respectively. **b–e**) Details of the four interfaces between Ceru+32 and GFP-17 in the protomer: 'Ceru+32 clockwise' interface (b); 'GFP-17 clockwise' interface (c); 'inter-ring' interface (d); 'inter-GFP-17' interface (e). Dotted lines correspond to hydrogen bonds. Ceru+32 and GFP-17 monomers are coloured light blue and gold, respectively. Underlined residues are the charged residues introduced to engineer the supercharged variants.

between proteins, we reduce the set of possibilities to the four shown in fig. 3.2. To assess the the relative stability of each of these candidates, we performed simulations at various values of κ and measured the radius of gyration R_g of the protomer to see which configuration exhibited stability over the largest range. Surprisingly, we found that all putative configurations were nearly identical in their stability range; moreover, all of these ranges dramatically underestimated the experimentally observed value.

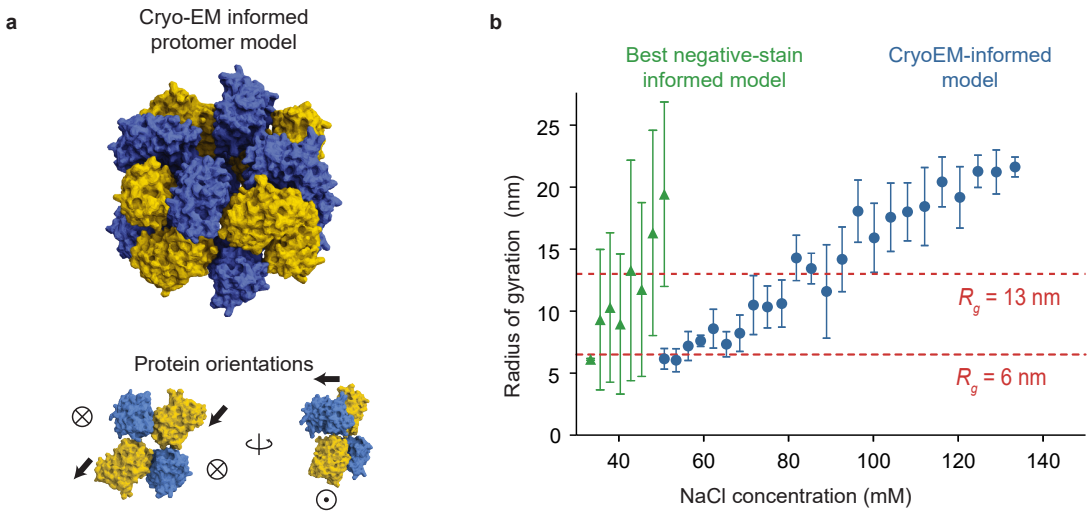

Figure 3.3 | **a**) Simulation of the Ceru+32/GFP-17 protomer with protein arrangements based on the cryo-EM data. Arrows indicate the Ceru+32/GFP-17 orientation within the protomer. Circles with crosses indicate termini pointing into the page, and circles with dots are pointing out of the page. **b**) Computationally predicted average radius of gyration versus NaCl concentration of the best negative stain informed protomer model (green triangles) and cryo-EM-informed model (blue circles). Data points and error bars are computed as the mean and standard deviation of the radii of gyration of five simulations conducted at each value of NaCl concentration. While the models informed solely from the negative stain electron microscopy denature below $50\,\mathrm{mM}$ NaCl, the structure inferred from the cryo-EM data remains stable up to $100\,\mathrm{mM}$. While this still underestimates the experimentally observed stability region, it demonstrates that the simulations successfully distinguish between the actual and other unstable configurations.

Collection of the cryo-EM data, which was motivated by these unexpected simulation results, allowed us to construct a much more precise model with orientations specified according to the resulting electron density map. The resulting simulations exhibited a much wider range of stability than the original simulations, providing a clear indication that this new proposed structure was more stable than any of our previous suggestions fig. 3.3. The result of this work was a clear demonstration of both the potential power and the limits of such a highly simplified protein model. Although the model successfully captured the relative stability of different configurations, it did not capture the true stability range shown from experiments. Moreover, this model remained too complex to feasibly simulate assembly due to both the large number of pairwise charged interactions to compute and the high complexity of performing overlap checks between arbitrary nonconvex meshes. These results motivated further research into ways to capture these properties of proteins both more efficiently and more effectively in simulation.

3.3 Using Machine Learning to Predict Protein Assembly

One immediate possible conclusion from the previous section is that such colloid-like models are simply not capable of achieving the level of detail required to accurately simulate these proteins. Indeed, papers like [22] strongly endorse the view that colloid-based models of proteins have sharp limits While such criticisms are typically based on the coarsest possible representations of globular proteins as spheres, they are typically also valid for many more detailed colloidal protein models because these models are based on intuition and ad hoc assumptions. While top-down coarse-graining approaches like the Martini model [101] can still be connected easily to the underlying atomistic representation, colloid-like models like those in [24], [25], and [26] must be justified based on whether the results of simulations match expected or observed behaviors.

Fortunately, the enormous amount of data now available on proteins suggests an alternate means to constructing such models in a more principled matter: using data-driven machine learning (ML) approaches to directly predict assemblies from protein properties, and in the process to quantify the importance of particular features on a given assembly based on known structures. Such an approach could encode numerous properties of a protein into a set of input descriptors to predict properties of the crystal the protein would assemble into, which would also allow direct evaluation of the relative importance of such descriptors. Rather than immediately attempting to predict protein crystallization, which is an extremely complex problem (as illustrated the previous section), I start by focusing on the simpler task of understanding how the symmetries of protein crystals are related to their constituent proteins.

It is well known that protein crystallization overwhelmingly produces crystals with one of a small number of possible symmetries as classified by the space group of the protein crystal [102]. This preference is strongly correlated with measures of the entropy of these structures, suggesting that the preference is largely statistical. However, this explanation provides no insight into when or why proteins crystallize into different space groups, or how the properties of a certain protein bias it towards or away from the most statistically preferred space groups. Since this symmetry preference has important implications for the possible protein structures that may be designed, I focus on training models to predict the space group a protein will crystallize into based on their shape.

3.3.1 Machine Learning from Protein Shape

Training these models requires deciding how to define the shape of a protein, a choice with significant implications because proteins are inherently flexible meshes. Any rigid represen-

tation of a protein therefore invariably sacrifices some fidelity because chain flexibility may be a necessary ingredient for some assemblies, so any model based on a rigid representation immediately becomes inapplicable to model the assembly of some set of proteins. Focusing only on this subset of proteins, however, I employ the same molecular surface approach used in the previous section. More precisely, I use the solvent-excluded surface (SES) [99], a representation commonly used in the literature that is also the basis for common metrics for shape complementarity [16], making it the most suitable choice as a basis for predicting quaternary structures.

The next step is determining a means to develop a model that can map from the SES to the predicted space group. Recent successes in classifying anisotropic colloidal particles with ML suggest that one possible approach is to compute a set of descriptors of the SES and train a model to recognize the relationship between these descriptors and the resulting space group [103]. However, standard geometric metrics used for convex colloids are far too limited to describe the complex, nonconvex meshes that represent proteins. To address this issue, I instead use a model capable of learning from a protein reduced surface mesh directly: a convolutional neural network (CNN).

Neural networks (NNs) are a powerful tool that in essence construct arbitrarily complex nonlinear mappings from input data to output predictions [104]. Although these networks are highly versatile, their most common usage is in supervised learning: a number of data points for which the output is known are provided to the network (the training set), which then optimizes the mapping to maximize predictive accuracy of the outputs, and is finally validated by making output predictions from previously unseen inputs (the test set). CNNs specialize this model to more naturally handle locality in spatially meaningful input data. Primarily designed for grid-like data (although they have since been generalized), CNNs embed locality into the mapping by performing local convolutions on input grids in the first few layers of the mapping, allowing them to learn how relationships between neighboring regions in the input data are correlated with the output.

This feature has made CNNs a popular tool for image classification, and they can be easily adapted to data sets that have natural image-like representations. As such, CNNs are a natural choice for predicting properties of protein crystal structures from protein shapes. To represent the protein surface as a grid-like image, I voxelize the SES into a binary occupancy-based 3D voxel grid such that each grid cell indicates the presence of the surface, generating a standardized representation for CNN learning. This representation can be made arbitrarily precise based on the resolution of the voxelization.

3.3.2 Embedding Descriptors in the Solvent Excluded Surface

Another critical feature of CNNs is their use of channels, i.e. the association of multiple pieces of data with each grid point; in standard image analysis, channels represent RGB colors. I exploit this by calculating additional features on the SES and then interpolating them onto the voxelized grid and placing them in additional channels. By modifying or removing certain channels, I can also use this model to determine how important different features are to accurately predicting protein crystals, results that I can use to inform physics-based models of protein crystallization. As seen in section 3.2.2, charged interactions play a critical role in many protein assemblies. The other significant contributor frequently included in analyses of protein assembly is hydrophobicity [105], so I choose to include that as well.

Since protein assembly typically occurs in water, a high dielectric solvent that usually contains ions, any evaluation of the electrostatic field about a protein must account for the effects of the environment. Assuming that the effect of these ions can be approximated well by the mean electrostatic potential of a Boltzmann-distributed collection of ions, the exact solution to the electrostatic field about the protein is given by the Poisson-Boltzmann equation, a nonlinear partial differential equation [106]. The Poisson-Boltzmann equation may be cast into an Euler-Lagrange equation where the integral of the resulting Lagrangian is the electrostatic free energy of the system [107, 108], proving that the Poisson-Boltzmann equation has a unique solution corresponding to the electrostatic potential of the system. In practice, most biomolecular simulations use a linearized version of the Poisson-Boltzmann equation due to the numerical complexity of solving the nonlinear version, and in fact most adopt a simplified Lagrangian that models the solute (the protein) as a collection of spheres interacting via a Coulomb potential (the Coulomb field approximation), resulting in a vastly simplified expression for the solvation free energy known as the generalized Born approximation [106, 109]. In this case, however, accuracy is more important than efficiency because the electrostatic surface only needs to be computed once as an input to the model rather than iteratively at each time step during integration of the equations of motion. As a result, I employ a direct solution to the Poisson-Boltzmann equation, using the `pygbe` tool [110] to perform GPU-accelerated calculations.

I embed hydrophobic information using the Kyte-Doolittle scale [111], a standard measure of hydrophobicity. This scale is based on the transfer free energies of of amino acids between different solvation states. Although other hydrophobicity scales have been developed based on slightly different data sets, and agreement between different scales is semi-quantitative at best, there is little agreement on which scale should be generally preferred. As a result, I chose the Kyte-Doolittle scale as it is the most frequently used. The

scale assigns values to each amino acid in a protein, and I then perform a local averaging about neighboring amino acids to account for their local environments in the protein. Both the computed electrostatic field and the hydrophobicity values are then interpolated onto the voxelized grid as additional channels.

3.3.2.1 Developing a Machine Learning Pipeline

Although feasible in principle, developing a CNN to analyze protein shapes in this manner faces numerous hurdles. In one sense, the quantity of data is limited: there are $\sim 150\,000$ proteins in the Protein Data Bank (PDB), a relatively small number of data points compared to the millions that are used to train high-fidelity NNs. Unfortunately, in another sense the quantity of data is extremely large: although the number of data points is relatively small, each data point is a three-dimensional image and therefore extremely large, resulting in a total dataset on the order of $10\,\mathrm{TB}$. Not only does analyzing these data require building a data pipeline capable of effectively feeding the data to the network, it also requires finding a way to efficiently process this data because CNNs are expensive, especially in three dimensions.

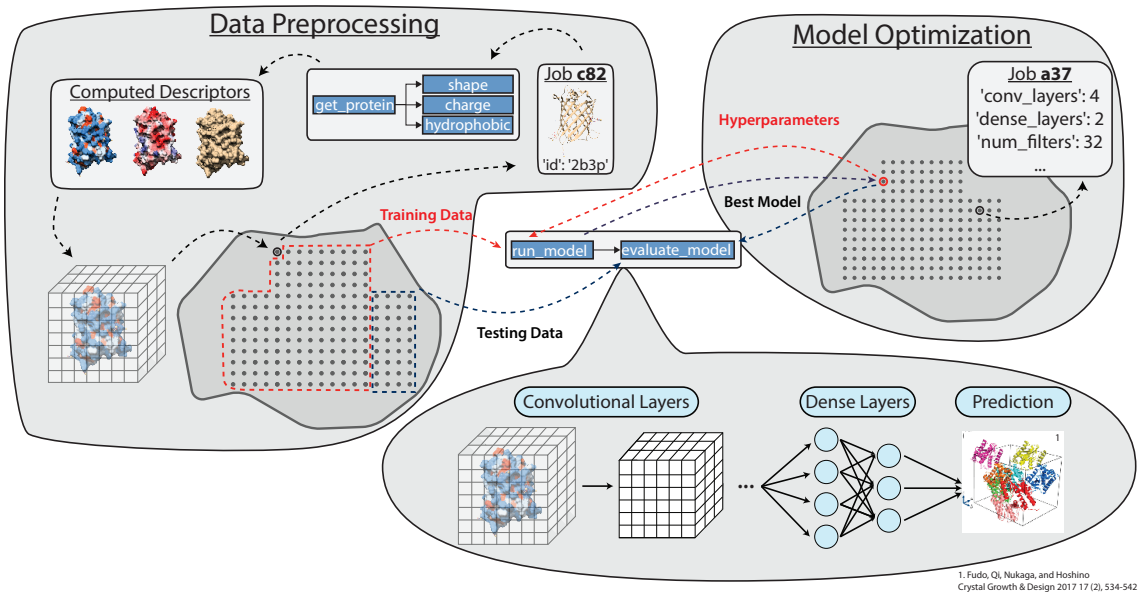

Figure 3.4 | ML from protein structures requires a sophisticated collections of tools to scalably manage the data and ML Each "data point" is a protein from the PDB, which is stored within an on-disk database managed by signac [86, 87]. Each protein is then processed independently to calculate various relevant features, such as its reduced surface, the electrostatic charge distribution, and the hydrophobicity of various regions. These data are then saved back into the same data space, from which they are read on the fly as part of training a convolutional neural network to predict the space group of a protein's crystal structure. A separate on-disk database is used to manage the hyperparameter optimization of this model, running numerous variants on the neural network to find the one that gives the best predictive performance.

Figure 3.4 shows the model and data pipeline that I constructed to handle this problem. Networks were built using TensorFlow [112], and data was managed using signac [86, 87]. These models were trained and tested on NVIDIA V100 graphics processing units (GPUs) on the Oak Ridge Leadership Computing Facility (OLCF) Summit supercomputer, one of the world's largest supercomputers. Use of these resources was critical to achieving the necessary scale for this model.

To address the issues with data size above, I stored the images on disk in a sparse array representation, reducing their size by roughly 2 orders of magnitude (down to a total of about 150 GB). Sparse 3D convolution is not supported by TensorFlow (or any other widely available tool at this time), an issue I solved by exploiting the configuration of the compute nodes on Summit. These nodes are composed of 42 central processing units (CPUs) and 6 GPUs, meaning that 7 CPUs are available for every GPU. To simultaneously limit disk usage and data transfer speeds using the sparse array representation on disk while still performing dense convolutions using TensorFlow on the in-memory data, I used the CPUs to perform a sparse-to-dense transformation of the data on-the-fly, constantly pipelining and preprocessing the data while the expensive convolutions were performed on the GPUs.

To analyze all of these proteins in a reasonable amount of time requires using hundreds of GPUs in parallel to process the incoming images. I parallelized my model using `horovod` [113], which uses the Message Passing Interface (MPI) to communicate between many GPUs and allows models to easily scale to extremely large data sets. Usage of this tool incurs additional complexity, however; in particular, reading the same data from disk from many parallel processes can result in deadlock due to contention. To avoid such problems, I ensured that each Message Passing Interface (MPI) rank only reads a fraction of the total data. Since these subsets of the data are randomly chosen, any correlations that happen to exist within a subset would be learned within that MPI rank, leading to biases in the model. To avoid this problem, I randomized the data by using MPI to randomly transfer a subset of data from each rank to another randomly selected rank at the end of each epoch.

3.3.2.2 Preliminary Results

I trained and tested various CNN architectures with this data pipeline. In this case, hyper-parameter optimization amounted to choosing various properties of the network such as the number and size of layers, the channels, and parameters relating to the convolution. Moreover, I tested this approach for various combinations of channels, including models with subsets of the three channels. However, the results showed that as currently constituted this model has limited predictive capability.

Learning based on shape (see fig. 3.5) alone leads to high accuracy on the training set,

Figure 3.5 | The results of the machine learning model using only a single binary channel encoding the shape. The resulting model successfully learns the training data easily, but as expected these predictions do not generalize to the test data set at all. The failure is expected because we do not believe protein shape alone should be sufficient in most cases to predict the space group (or most other relevant crystal structure features).

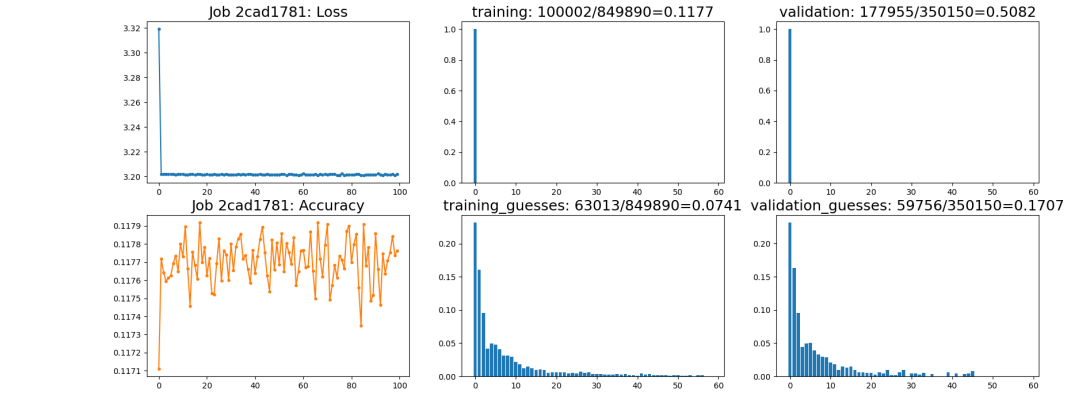

Figure 3.6 | The results of the machine learning model using three channels encoding shape, charge, and hydrophobicity. The resulting model fails to learn after the initial step despite various normalizations and other tricks. The failure of the model to make any predictive improvements suggests that much more care must be used in curating and pruning the input data into a format that is more appropriate.

but the resulting model fails to generalize to the test data, performing no better than a trivial prediction based on the distribution of space groups in the PDB. This result may indicate that shape alone is an insufficient predictor of space group; however, the results for the full three channel model suggest otherwise (see fig. 3.6). These results indicate that the model essentially fails to learn from the input data, even on the training set, indicating that more care is required either in the curation of the input data or in the choice of model parameters. Possible next steps are discussed in more depth in section 8.2.1.

CHAPTER 4

A Mean-Field Approach to Simulating Anisotropic Particles

This chapter is reproduced from Ramasubramani, V., Vo, T., Anderson, J. A. & Glotzer, S. C. A mean-field approach to simulating anisotropic particles. *The Journal of Chemical Physics* **153,** 084106 (Aug. 2020).

4.1 Introduction

Particle simulation has become a crucial component of the scientific toolbox for investigating a wide range of physical phenomena, with various simulation techniques to choose from based on which ones are best suited to the phenomena of interest [89, 114]. A core element for such simulations is the pair potential, which governs the pairwise interactions between any two particles. Pair potentials are ubiquitous in physics and chemistry, describing everything from the gravitational interaction of celestial bodies to the electrostatic forces between atoms and molecules. The straightforward application of pair potentials representing Coulombic and van der Waals forces to point particles representing atoms has been enormously successful in predicting a wide array of behaviors in highly complex systems, from biomolecules to organic compounds and metals [3, 6, 27, 115–119]. However, despite algorithmic developments and improved hardware, the range of applicability of this approach for molecules or particles of complex geometry remains limited by the complexity of computing all relevant interactions for sufficiently large numbers of particles [8–10, 27].

Coarse-graining, the process of collectively handling many atomistic degrees of freedom, is a powerful strategy to increase the time and length scales accessible via simulation [7, 120]. Examples of this include the bead-spring polymer model [121, 122] and the Martini force field for proteins [7]. However, coarse-grained models still utilize isotropic pair potentials similar in form to those used in all-atom simulations, depending upon bonded

networks of isotropic particles to represent nontrivial geometries, such as proteins [95, 123]. While such approaches assume that the internal flexibility and interactions within such networks are important, for many phenomena these models can be simplified by treating each such network as a single rigid body [124], for example by using sets of overlapping spheres to represent a rod or cube [125, 126]. Despite substantially reducing complexity and admitting more efficient simulation techniques [127], though, these methods ultimately still require computing a large number of interactions. Moreover, recent studies have shown that even coarser representations of particle shape are frequently sufficient to predict thermodynamic and kinetic behavior [12–14], a discovery that motivates the ongoing pursuit of new coarse-graining approaches and simulation methodologies for studying anisotropic particles.

Numerous different methodologies have emerged for the simulation of anisotropic particles depending on the phenomena of interest. MC methods have been extremely successful in elucidating the assembly behavior of shaped particles [12–14]. Since MC methods do not require force fields, potential energies need not be represented by smooth or even continuous functions. This flexibility has been critical to previous studies of shaped particles, which have made extensive use of the hard particle approximation to bypass the need for even an analytical expression to represent interactions between particles [79]. However, because MC simulations rely on a statistical sampling rather than a time integration of equations of motion, in principle they only provide information on equilibrium (thermodynamic), not kinetic, behavior unless we make further assumptions. Furthermore, the hard particle approximation captures but one aspect of the behavior of attractive shapes since no overlap is permitted, and parallelizing HPMC simulations to the degree possible in MD remains an unmet challenge. As a result, an MD method for simulating anisotropic particles remains eminently desirable.

Perhaps the best-known method for MD simulations of anisotropic particles is the Gay-Berne (GB) potential, which rescales the distance and energy in the 12-6 Lennard-Jones (LJ) potential by the relative orientations of the pair of particles to simulate ellipsoids [128–130]. The potential is an analytic function that can be computed efficiently, but no known generalization of the Gaussian overlap formulation exists for arbitrary geometries, illustrating the central problem for anisotropic pair potential development. One method for circumventing this difficulty is using event-driven molecular dynamics (EDMD), which can be much faster than force-based MD in some cases [131, 132] and has been used to great effect to simulate systems ranging in complexity from simple hard spheres [133] to fibril assembly from proteins [134]. However, although EDMD has been used to study the assembly of hard cubes [11], there is no known generalization for arbitrary geometries.

An alternate approach to dynamical simulations of shaped particles is the discrete element method (DEM), in which geometries are decomposed into discrete pieces whose interactions can be independently computed and then summed. This method has been used to great effect in the granular materials community [135], and it has been adapted to conform to the energy and momentum conserving schemes preferred in other particle simulation communities [136]. While DEM can be used to study both dynamical and equilibrium behavior, it, too, suffers from certain limitations. DEM is generally restricted to working with shapes that admit a decomposition into a discrete collection of elements, and while it readily admits parallelization of the numerous calculations, the approach scales poorly to complex geometries where the number of discrete interaction sites becomes prohibitively expensive. As a result, we seek a method that can ameliorate these performance concerns while still accurately representing both smooth (e.g. ellipsoidal) and discontinous (e.g. polytope) geometries.

In this paper, we introduce a framework for deriving anisotropic pair potentials. The resulting anisotropic pair potentials generalize existing isotropic pair potentials in a way that largely combines the strengths of the previously discussed methods. Since these pair potentials can be used in either MD or MC simulations, the method allows the study of both dynamical systems and systems out of equilibrium. Moreover, simulations using these pair potentials can take advantage of both the parallelization schemes used in MD and the cluster moves and advanced sampling techniques used to overcome kinetic barriers in MC. Moreover, by choosing an appropriate pair potential the method can be used to study both attractive and purely repulsive particles. For discrete geometries such as polytopes, the non-smoothness associated with parallel faces and sharp edges and vertices introduces discontinuities into the potential energy landscape with respect to the orientational coordinates, but we show that these discontinuities can be ameliorated with a local averaging that achieves energy conservation comparable to DEM while retaining advantageous performance characteristics.

We organize this paper as follows. In section 4.2, we describe the derivation of anisotropic pair potentials by applying a mean-field approximation to a form of Poisson's equation that describes a wide range of interactions. The resulting pair potentials are applicable to a wide range of shapes, including the ellipsoids and polytopes that are our principal focus in this work. As an example, we derive an anisotropic generalization of the LJ potential – the anisotropic Lennard-Jones (ALJ) potential – along with a purely repulsive anisotropic Weeks-Chandler-Anderson (AWCA) variant. Section 4.3 discusses a practical implementation of this ALJ potential for use in MD simulations on both CPUs and GPUs. In section 4.4 we validate this implementation and measure its performance, focusing on the

AWCA potential since there are directly comparable methodologies (primarily DEM) that have been used for simulations of purely repulsive shapes in the past. We also demonstrate the utility of this method for studying particle self-assembly by demonstrating the assembly of some common colloidal crystal structures. Finally, we summarize and consider avenues for future work using this potential in section 4.5. This work makes extensive use of NumPy [137] and rowan [84], and figures in this paper were generated using Matplotlib [85] unless otherwise noted. Data and workflows were managed with the signac framework [86, 87].

4.2 Theory

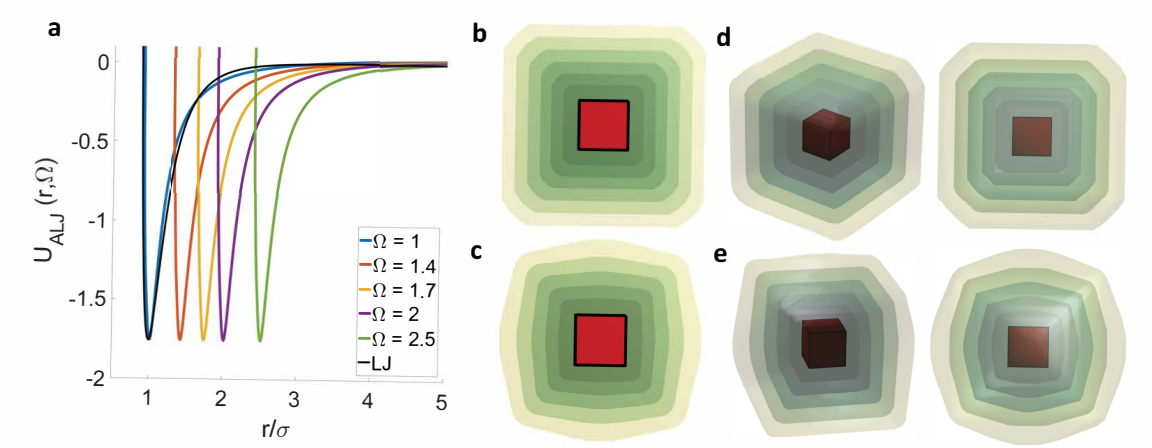

Figure 4.1 | The ALJ potential is derived by solving eq. (4.4) using an LJ input potential $U_{in} = U_{LJ}$. a) The ALJ potential is effectively an LJ potential shifted by the value of Ω at a given orientation. The standard LJ potential is shown in black for reference. b-c) The isocontours of the ALJ potentials for a square (b) and a cube (c) retain the particle's shape even at intermediate distances. This consistency is crucial to ensuring that the potential generates a globally (rather than locally) anisotropic field. d-e) Without the appropriate rescaling of ϵ_s in eq. (4.6), the isocontours quickly become more spherical (disc-like in 2D) further away from the shape.

Consider a system of N anisotropic particles occupying a volume V. The force on a reference particle with position $\vec{r}$ and orientation α from the potential energy field generated by all the other particles in some fixed configuration can be written as

$$\nabla U(\vec{r}, \alpha) = \int_V \nabla U(\vec{s}, \beta)\rho(\vec{s})dV \tag{4.1}$$

where ∇U is the vector force field (the derivative of the underlying potential of interest), ρ defines the distribution of particles within the volume V, and s and β are the positions and orientations of the particles relative to the reference position and orientation.

For simplicity, we restrict ourselves to shapes that can be constructed by the anisotropic dilation of a sphere contained in that shape. Mathematically, this motivates defining a new

coordinate system (r, θ, ϕ) such that $x = rf_x(\theta, \phi)$, $y = rf_y(\theta, \phi)$, $z = rf_z(\theta, \phi)$, where r is the radial distance from the sphere center and f_x, f_y, f_z are some functions encoding the angular dependence of x, y, and z, respectively. This parameterization exists if and only if there is a sphere of radius r (not necessarily unique) for which a unique mapping $(\theta, \phi) \rightarrow (x, y, z)$ exists. Locally, the magnitude of the dilation of the sphere to the shape at any point (r, θ, ϕ) is given by $\Omega(\theta, \phi) = \left(f_x^2 + f_y^2 + f_z^2\right)^{1/2}$. This parameterization exists for all convex shapes, and more generally for any shape for which a sphere (not necessarily unique) may be chosen such that $\Omega(\theta, \phi)$ is unique relative to the center of that sphere. An example of a nonconvex shape that admits such parameterization is a star; conversely, an example for which this parameterization is impossible is a U-shaped particle (see section 4.3 for how to simulate such a particle in practice).

In this work, we define the insphere of a shape as the largest sphere whose center coincides with the centroid of the shape that is also contained inside the shape. In the subsequent derivation, the coordinate system is always defined relative to this insphere. As a clarifying example, for a cube whose insphere radius is 1, $\Omega = 1$, $\sqrt{2}$, and $\sqrt{3}$ at the face centers, edge centers, and corners, respectively. Such an approach has been successfully employed both to develop an equation of state for hard polyhedra [138] and to predict ligand shell morphology about an anisotropic particle [139]. Transforming to these new coordinates, we find

$$\nabla U(\vec{r}^{\,\prime}, \alpha^{\prime}) = \int_{V^{\prime}} \nabla U(\vec{s}^{\,\prime}, \beta^{\prime}) \rho(\vec{s}^{\,\prime}) r^2 \Omega(\beta) dV^{\prime} \tag{4.2}$$

where the Jacobian determinant of the coordinate transformation is simply $r^2\Omega(\beta)$ and Ω is a function of the relative orientation of each particle. Taking the divergence of both sides of eq. (4.2) results in a form of Poisson's equation:

$$\frac{\partial^2 U}{\partial r^2}(\vec{r}^{\,\prime}, \alpha^{\prime}) = \frac{\partial U_{in}}{\partial r}(\vec{r}^{\,\prime}) \rho(\vec{r}^{\,\prime}) r^2 \Omega(\vec{\alpha}^{\,\prime}) \tag{4.3}$$

where U_{in} represents the radial component of the potential U. This component can be separated due to two important features of eq. (4.3). First, because we assume that U is conservative, all non-radial components of the divergence vanish by symmetry. Second, because we restricted ourselves to shapes that can be modeled by anisotropic dilations of a sphere, U is separable into its radial and angular components, and the θ and ϕ terms in the radial derivative cancel. A crucial consequence of these two features is that all orientational dependence on the right-hand side of the equation is confined to the Ω term, allowing us to consider the derivative of only the radial part of the potential on the right side.

Therefore, we can consider a self-consistent field solution to eq. (4.3) using an isotropic

pair potential of interest as the *input potential* U_{in}, where U_{in} could be any of the isotropic pair potentials commonly used in simulations. There is no known analytical solution to eq. (4.3) because no general expression exists for Ω across the class of anisotropic particles. However, in simulations we only need to evaluate U for a specific configuration, for which we can employ iterative approaches to determine Ω for a given pair of particles even if no analytical solution is known for computing the distance between that particular pair of shapes. As a result, we can treat Ω as a constant for a given configuration and solve eq. (4.4) for the effective pairwise interaction between anisotropic particles given an input potential U_{in}.

With a constant Ω, eq. (4.3) may be solved numerically given an expression for the density, so we now make three further approximations to derive such an expression. First, rather than solving for the potential in a particular fixed configuration, we instead perform an ensemble average to replace U with the potential of mean force U_m, allowing us to approximate the system density distribution as $\rho = \rho_o e^{-\beta U_m}$. Second, although U_m is a configurational integral over all positions and orientations, we assume that the field strength at any point in space is dominated by the closest point (θ_c, ϕ_c) and perform a first-order Taylor expansion of U_m about this point. Third, we Taylor expand the exponential itself, yielding the following differential equation:

$$\frac{\partial^2 U_m}{\partial r^2}(r, r_c) \sim \frac{\partial U_{in}}{\partial r}(r)\rho_0 \left(U_0 - U_m(r, r_c)\right) r^2 \Omega(\alpha) \tag{4.4}$$

where U_0 is the first term in the Taylor expansion. For a system of interacting anisotropic particles, the Taylor expansion of U_m amounts to assuming that the field strength at any point in space is dominated by the closest point (θ_c, ϕ_c) on each particle to that point in space. The corresponding distance r_c, which we term the "contact" distance, is a function of the orientation α, but the formulation of eq. (4.4) allows us to instead view this equation in terms of the parameters r and r_c, both of which are easily computed from simulations.

We are now in a position to solve eq. (4.4) for a given isotropic pair potential. For our purposes, we choose the traditional LJ potential. When $U_{in} = U_{LJ}$, the numerical solution to eq. (4.4) is – not surprisingly – similar in shape to an LJ potential, shifted along the x axis by Ω (fig. 4.1a). This suggests employing the LJ functional form as a basis function for defining approximate *analytical* solutions for U_m. Using this basis function and a change of variables allows us to solve for U_m. Requiring that U_0 reflect the same anisotropic shape as U_m provides a condition that can be used to solve for U_0, resulting in the following approximate form for the ALJ potential $U_{ALJ} = U_0 + U_m$:

$$U_{ALJ} = 4\epsilon_s \left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6} \right] + 4\epsilon \left[\left(\frac{\sigma_c}{r_c}\right)^{12} - \left(\frac{\sigma_c}{r_c}\right)^{6} \right] \tag{4.5}$$

where we call the U_0 term the *central* interaction (because it depends only on r) and the U_m term the *contact* interaction (because it depends on r_c). σ_c is a free parameter that controls the relative strength of the contact interaction and determines how well the potential represents the shape of interest. In the limit $\sigma_c \to 0$, the shape will be sharply preserved, but the resulting potential will be extremely hard and require very small time steps to simulate. Conversely large values of σ_c will lead to an effectively softer (but much more rounded) representation of the shape. The requirement that U_0 must preserve the anisotropic core shape in eq. (4.5) (figs. 4.1b-c) results in defining $\epsilon_s = \epsilon\,(L_1/L_2)$, with

$$\begin{aligned} L_1 &= \left(\frac{\sigma_{ij}}{r - M_{ij}}\right)^{12} - \left(\frac{\sigma_{ij}}{r - M_{ij}}\right)^{6} \\ L_2 &= \left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6} \end{aligned} \tag{4.6}$$

where σ_{ij} is the average of the insphere diameters of particles i and j and $M_{ij} = (\Omega_i(\theta_c, \phi_c) - 0.5\sigma_i) + (\Omega_j(\theta_c, \phi_c) - 0.5\sigma_j)$. Figs. 4.1d-e show the isocontours of U_{ALJ} if this rescaling is ignored and we simply set $\epsilon_s = \epsilon$, in which case these potential energy surfaces bulge the faces of the shape in a way that does not reflect the underlying shape.

The core insights of our theory lie in eqs. (4.2) and (4.5). The success of the coordinate transformation eq. (4.2) in reducing eq. (4.1) to the tractable form of eq. (4.5) demonstrates that for a wide class of anisotropic particles (those for which the coordinate transformation is valid) and a wide class of potential forms (those for which the first-order Taylor expansion about the contact point is a good approximation), the total interaction reduces from a volume integral to a sum of two point interactions. While our primary usage of this construct in this work is to develop a new framework for computational simulations, it could also be used in future theoretical treatments of the interaction of anisotropic particles.

Although this potential includes both attractive and repulsive interactions, as discussed in section 4.1 the most commonly used methods for simulations of shapes at present have largely focused on hard shapes. For comparison with these existing methods, we define a short-ranged repulsive variant of our potential analogous to the Weeks-Chandler-Anderson (WCA) potential. This AWCA potential is simply a truncated and shifted ALJ potential, with the additional note that central and contact components of the potential are truncated at their respective minima (note that this also applies to the L_1 and L_2 terms defining ϵ_s). The results in all future sections use the AWCA potential rather than the ALJ potential unless noted otherwise.

4.3 Implementation

We have implemented the ALJ pair potential in the simulation engine HOOMD-blue [77, 78]. As described in eq. (4.5), the ALJ potential is the sum of a central potential U_0 and a contact potential U_m evaluated between the closest points on a pair of particles (A, B). To find the closest contact points $\{a \in A, b \in B\}$, we employ a modified version of the Gilbert-Johnson-Keerthi (GJK) distance algorithm. We define torques $\vec{\tau_i} = \vec{r_i} \times \vec{F_m}$, where $F_m = -\nabla U_m$ and the moment arms are given by $\vec{r_a} = a - q_a$ and $\vec{r_b} = b - q_b$ (where q_i is the centroid of particle i). As noted in section 4.2, this pair potential is only defined for shapes that are amenable to the coordinate transformations described there. However, in practice arbitrary nonconvex particles may be simulated by using rigid bodies composed of convex ALJ particles.

Finding center-center interparticle distances is handled by the existing machinery in HOOMD-blue used by other pair potentials. The most complex part of implementing this anisotropic pair potential is therefore developing a sufficiently performant and robust implementation of the GJK algorithm. In order to discuss our implementation and some of its particular features, we provide a brief introduction to GJK here that covers the crucial details. We refer the interested reader to refs. 140 and 141 for more extensive descriptions of GJK (the following sections closely follow the notation of ref. 140).

4.3.1 Implementing GJK

The GJK algorithm [142, 143] for computing the minimum distance between two arbitrary convex sets is based on two crucial geometric concepts: Minkowski sums and support functions. The central idea is that given two convex sets A and B, the vector joining the closest points on the two shapes is equal to the vector defining the closest point to the origin on the Minkowski sum of A and $-B$ (defined as $A - B = \{a - b : a \in A, b \in B\}$); consequently, when two shapes overlap, the origin must be contained in $A - B$. Support functions are the critical ingredient that provide an implicit representation of $A - B$ via its supporting hyperplanes, bypassing the otherwise prohibitively expensive explicit computation of the Minkowski sum. The support function $h_K(\vec{x}) = sup\{\vec{x} \cdot y : y \in K\}$ of a convex set K provides the distances between its supporting hyperplanes and the origin, and the corresponding support mapping $s_K(\vec{x})$ defined by the relation $h_K(\vec{x}) = s_K(\vec{x}) \cdot x$ gives the corresponding point on the hyperplane.

In essence, GJK is a descent algorithm that constructs a sequence of simplices $(W_0, W_1, W_2, ...)$ contained in $A - B$ and a corresponding sequence of vectors $(\vec{v}_0, \vec{v}_1, \vec{v}_2, ...)$ that is guaranteed to converge to the point of minimum norm $\vec{v}^*$. At each iteration, the

point $\vec{w}_k = s_{A-B}(\vec{v_k})$ is added to W_k, and $\vec{v}_{k+1}$ is set to the closest point to the origin in the augmented W_k. The smallest subset of this set that contains $\vec{v}_{k+1}$ then becomes W_{k+1}. Finding this subset efficiently is a critical part of GJK and is discussed in more detail below.

We implemented a version of GJK with a few noteworthy features that stem from our specific requirements. In addition to the vector $\vec{v}^*$, we also require the points $\vec{a}^* \in A$ and $\vec{b}^* \in B$ that correspond to the endpoints of $\vec{v}^*$ to apply the torques. To find these points, we note that

$$W_k \subseteq \{s_{A-B}(\vec{v})_i : i <= k\} \tag{4.7}$$

$$v_{k+1} = \sum_{w_i \in W_k} \lambda_i w_i \tag{4.8}$$

s.t. $\sum_i \lambda_i = 1$ and $\forall i \ (\lambda_i >= 0)$. These equations imply that $\vec{v}_{k+1}$ is a convex combination of $\{s_{A-B}(\vec{v}_k)\}$. Defining the index set $I_k = \{i : \vec{v}_i \in W_k\}$, we can find the necessary points using

$$a_{k+1} = \sum_{i \in I_k} \lambda_i s_A(\vec{v}_i) \tag{4.9}$$

$$b_{k+1} = \sum_{i \in I_k} \lambda_i s_B(\vec{v}_i) \tag{4.10}$$

Therefore, to find $\vec{a}^*$ and $\vec{b}^*$ we track the support evaluations for the two shapes s_A and s_B corresponding to the current $\vec{v}_k$. When the algorithm converges, we use these evaluations and the computed λ_i values to recover the contact points $\vec{a}^*$ and $\vec{b}^*$, and the difference gives us $\vec{v}^*$.

Accurately estimating $\vec{a}^*$ and $\vec{b}^*$ in addition to $\vec{v}^*$ using the classical GJK algorithm is infeasible due to numerical inaccuracies in the original method for determining W_{k+1} from W_k, which is known as the Johnson subalgorithm (JH). JH solves a linear system of equations to find the minimal subset W_{k+1} from W_k, but near-degenerate configurations are known to cause severe robustness issues for the method, and we found that even the most robust termination conditions[140] for JH led to solutions that, in the worst cases, deviated from $\vec{v}^*$ by up to 5%. More critically, in many cases even much smaller errors in $\vec{v}^*$ led to dramatically incorrect approximations of $\vec{a}^*$ and $\vec{b}^*$, frequently leading to torques in the wrong direction and causing unacceptably severe violations of energy and momentum conservation. To solve this problem, the original GJK paper proposes an expensive backup

procedure that essentially amounts to an exhaustive testing of all possible points in the set [142], but this procedure is prohibitively expensive for our purposes, particularly on GPUs where it introduces severe warp divergence.

To ameliorate this issue, we implemented the recently developed Signed Volumes subalgorithm (SV) [144]. This subalgorithm is substantially more robust and avoids the incorrect torques resulting from JH. It also significantly improves performance, both because it accelerates convergence and because it requires far less caching of intermediate calculations, reducing memory requirements in otherwise compute-bound GPU kernels. We observed speedups of nearly 30% on the GPU even for simple polyhedral geometries where the total number of iterations is tightly bounded by the sum of the number of vertices of both shapes. We expect even larger improvements for curved geometries where SV improves convergence; however, due to the stability issues encountered in simulations using JH, we did not conduct this performance comparison.

To improve energy and momentum conservation we enforce a strict ordering on evaluations of GJK. On the GPU, HOOMD-blue computes all interactions twice to avoid communicating between threads, but due to limitations of finite precision arithmetic, differences between $gjk(i,j)$ and $gjk(j,i)$ are enough to lead to violations of momentum conservation even with convergence tolerances on the order of machine precision. To avoid this issue, we enforce that for any pair $U(i,j) = U(j,i)$ by evaluating $gjk(i,j)$ if $i < j$ and otherwise evaluating $gjk(j,i)$. This ensures that Newton's third law is obeyed ($F(i,j) = -F(j,i)$) and momentum is conserved. We tested alternative estimators for the true distance, including $\min(gjk(i,j), gjk(j,i))$ and mean$(gjk(i,j), gjk(j,i))$, but we found no measurable difference in the energy conservation of the system over the tested timescales. We therefore chose the current minimum index heuristic for computational efficiency since it only requires a single evaluation of the distance algorithm.

4.3.2 Polytopes

The contact point approximation breaks down for polytopes because there can be multiple pairs of points between two polytopes whose pairwise distances are arbitrarily similar. Although forces computed remain reasonable because the contact distance is a smooth function, changes in the contact point can be arbitrarily large relative to the simulation timestep, introducing a discontinuity in the energy with respect to the orientational coordinates of the system. For a simple case illustrating this discontinuity, consider the interaction between a pair of edge-aligned squares in 2D. In this configuration, an arbitrarily small rotational move of either particle about its respective centroid would result in a shift of the contact point

from one vertex to the other. Since this shift is not a function of the integration timestep, the change in torque can become arbitrarily large relative to the move size, leading to significant violations of energy conservation.

To mitigate this issue, instead of only computing interactions about the contact point, we compute a sum of all interacting features *a la* DEM [136]. In 2D, we compute all vertex-edge interactions between polygon edges, whereas in 3D we compute both vertex-face and edge-edge interactions between polyhedra. To avoid unnecessary computation of negligible interactions, we construct the vector joining the centroid of one shape to the contact point on the other shape (found using GJK) and identify the face this vector passes through. In all relevant cases, this face accounts for all interactions that are necessary to consider. This method therefore leads to energy conservation comparable to DEM while ensuring that performance scales with the number of vertices in the largest face of the polytope rather than scaling with the total number of vertices (which is the principal reason for poor performance scaling in DEM).

Note that for pathological geometries where multiple faces on a shape are arbitrarily close to parallel, averaging over a single face may not be enough. In this case, a local search of the vertex graph to find and test distances for all facets neighboring the closest facet pair would be necessary to ensure perfect equivalence with DEM. Since none of the current shapes of interest meet this criteria, we omit this additional smoothing procedure here. Currently, this averaging is enabled by default in the implementation in HOOMD-blue but may be turned off by users in Python scripts in situations with less restrictive energy conservation requirements. All presented results are gathered with the full face averaging unless stated otherwise.

4.4 Results

4.4.1 Validation

4.4.1.1 Ellipsoids

We first validated our method for smooth shapes, where the contact-point formalism should hold exactly. In fig. 4.2, we show long-time energy conservation for a system of AWCA ellipsoids; over the timescales observed we see no measurable drift. It is worth noting that simulations using the GB potential to represent equivalent ellipsoids result in relative fluctuations that are 1-2 orders of magnitude smaller. This difference can largely be attributed to that fact that, since the surface component of the AWCA potential is extremely hard (it is essentially a LJ potential with a very small σ that has been shifted to the surface), small

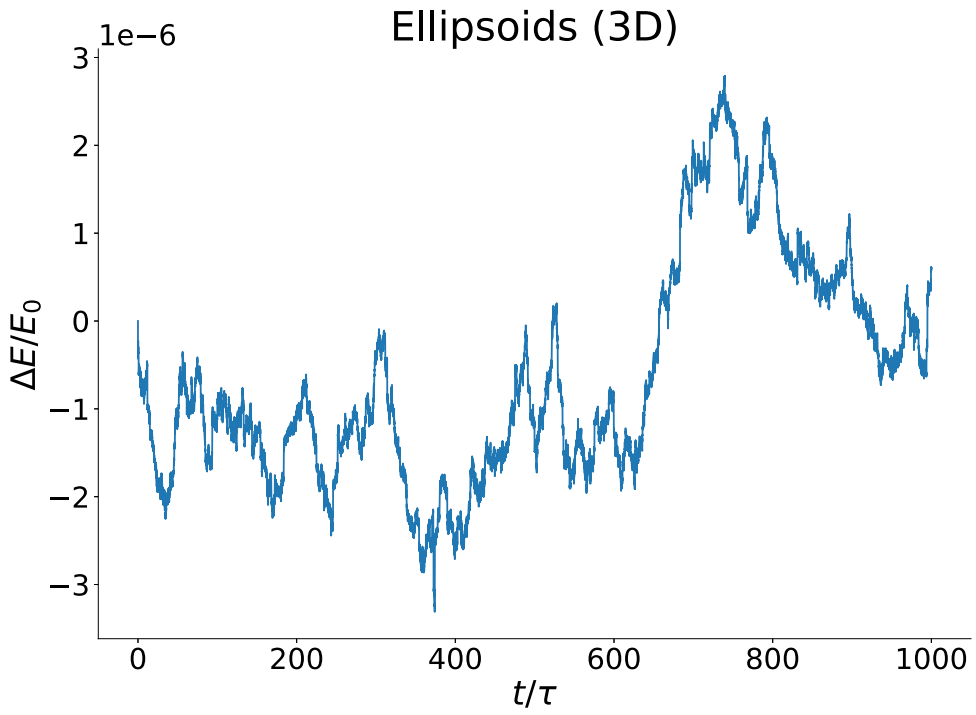

Figure 4.2 | The total energy of a system of ellipsoids with an aspect ratio of 2 for $dt = 1 \times 10^{-4}\tau$ shows no measurable drift over 1000τ, where τ is the unit of time in the simulation.

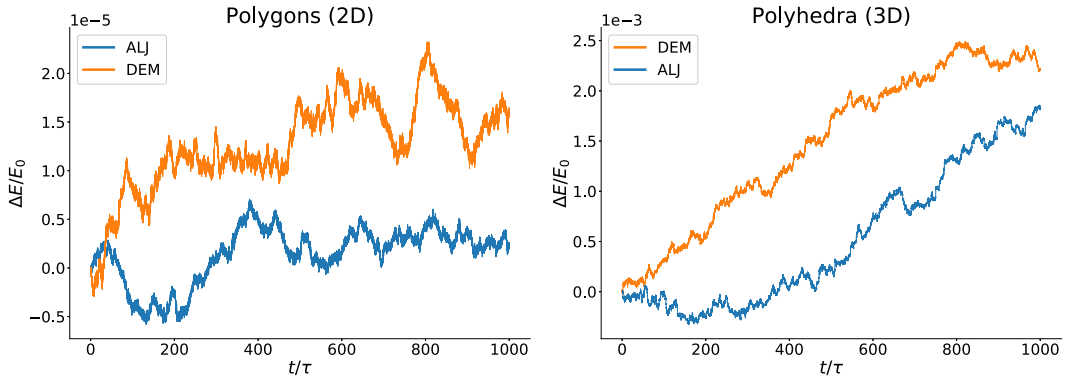

Figure 4.3 | The long-time energy conservation of polytopes simulated using both the AWCA potential with face averaging and DEM are very similar over 1000τ, where τ is the unit of time in the simulation. In two dimensions, neither method shows significant drift on the timescales tested for a system of squares. In three-dimensional simulations of tetrahedra, some drift is evident due to the inherent discontinuity associated with parallel edges in three dimensions, which causes discontinuities in the evaluated torques. However, we do find that the AWCA potential achieves comparable conservation to DEM (and as discussed in section 4.4.2, at a fraction of the computational cost). We choose squares and tetrahedra to show the relative behavior of two shapes with the same number of vertices in two and three dimensions.

movements result in much larger changes in energy. There may also be small effects due to the use of GJK to compute the contact point, since any numerical error in this calculation could cause larger shifts in the computed distance from one timestep to the next than the true change. However, our implementation is accurate up to machine precision [144], and we observed no measurable drift over the tested timescale, suggesting that any such fluctuations would be very minor if present. Note that the true shape of the ellipsoid is determined by the sum of the underlying principal axes and the rounding radii; for instance an ellipsoid specified with $a = 3, b = 2, c = 1$ and $\sigma_c = 0.1$ would effectively be an ellipsoid with principal axes $a = 3.1, b = 2.1, c = 1.1$. Any desired ellipsoid can still be achieved by setting the principal axes of the internal ellipsoid appropriately.

4.4.1.2 Polytopes

We next validated the method for AWCA polytopes. All simulations of polyhedra in this paper were conducted using contact spheres of size $\sigma_c = 0.15\sigma$, where σ is defined as the insphere diameter of the shape and is used to evaluate the central potential. For instance, for a unit volume cube, the diameter of the insphere is unity so $\sigma_c = 0.15$. Although, this choice makes the potential very hard and necessitates a very small dt, we chose it to retain very accurate representations of our shape. It has previously been shown that judicious choice of rounding radius can still result in the desired assemblies while also allowing timesteps one to two orders of magnitude larger [136], so such choices can be made to accelerate simulations on a case-by-case basis.

Without the averaging described in section 4.3.2, system energies drift significantly even over relatively short times in both 2D and 3D simulations of polytopes. Simulations in HOOMD-blue can be set to completely ignore anisotropic degrees of freedom by setting the parameter `aniso=False` in the integrator. When doing so, we observe energy conservation on par with LJ sphere simulations, indicating that the discontinuity is indeed entirely in the rotational motion of the system.

In fig. 4.3, we show the result of simulations that employ the face averaging described in section 4.3.2. In general, the energy conservation of the AWCA is on par with that of DEM. While proper face averaging is sufficient to avoid energy drift in two dimensions, we note that that in three dimensions both DEM and the face-averaged AWCA show significant long-time drift (see fig. 4.3b). This drift is the result of the edge-edge interactions, which cause small discontinuities in the torques for near-parallel edges because the closest point jumps from one vertex to the other (essentially the same issue that arises in simulations without face averaging). However, the edge-edge interactions are necessary to account for all possible collision configurations: for instance, a pair of shapes whose closest points

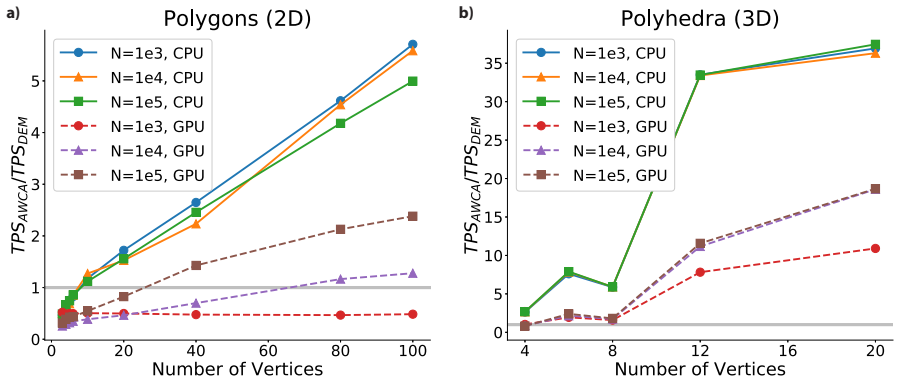

Figure 4.4 | Here we compare the performance in time steps per second (TPS) of the AWCA potential to DEM for a range of system sizes on both the CPU and the GPU. a) The performance for the two methods is generally on the same order of magnitude for two-dimensional systems of polygons, with DEM being up to a factor of 2 faster for small systems of simple polygons while AWCA becomes faster as system sizes increase and polygons gain vertices. b) The performance of the AWCA potential in three dimensions quickly outstrips that of DEM, with between one and two orders of magnitude improvement in the performance for large systems of the most complex shapes tested (dodecahedra). The three-dimensional data shows a jagged, nonlinear trend in the performance ratio due to the fact that 8-gons (cubes) and 20-gons (dodecahedra) both have some non-triangular faces, whereas the other tested shapes have exclusively triangular faces. The AWCA method performs averaging over only the nearest face, and in general calculating the potential between two polygonal faces with N vertices involves $2N + N^2$ face-vertex and edge-edge calculations. Since DEM performs a comparable amount of work in all cases since it averages over all features in all cases, the relative impact of more complex faces is larger for the AWCA potential, resulting in a sublinear performance in those two cases. CPU benchmarks were performed on a single Intel® Xeon® Gold 6154 @ 3.0GHz, and all GPU benchmarks were performed on an NVIDIA Quadro V100. This figure was generated using Pandas [83].

lie on perpendicular edges would exhibit incorrect collision dynamics if only vertex-face interactions were considered.

Since any attempt to smooth this interaction imposes an effective distortion or rounding on the shape, we prioritize faithfully reproducing the shape since (as discussed in ref. 136) the measured energy drift is acceptable for the coarse-grained simulations for which this method (like DEM) is most likely to be used. In fact, the AWCA energies in these plots are nearly identical to the DEM energies, which conforms to our expectations: in repulsive systems, the cutoff of the AWCA potential is short enough that the contact term dominates most pair interactions. As a result, the behavior of the two methods is essentially identical (although the AWCA potential is much faster in many cases as we will show). However, we note that this is not the case when the attractive, long-ranged ALJ potential is considered; in that case, the center-center interaction becomes the dominant contribution because at large separation distances the sharply decaying contact term quickly becomes negligible.

4.4.2 Performance

We next discuss the performance of our method. Since our method is implemented in HOOMD-blue, it inherits most of the advantageous properties such as MPI parallel execution on multiple nodes and strong scaling on GPUs[78]. The main distinction between our potential and the typical pair potentials in HOOMD-blue is that the ALJ potential is substantially more compute- and memory-intensive to evaluate. As a result, whereas in typical simulations of LJ particles (for instance) significant portions of the simulation time are taken up by tasks like neighbor list builds, in ALJ simulations the energy evaluation becomes the bottleneck, so its intrinsic properties become very important to the feasibility of simulations.

For the simulation of polytopes, we again compare to the DEM method since it is the most similar method from this perspective. There are two main differences that we expect to govern the performance differences between the two methods. The first difference is in the operation count of the two methods. DEM computes interactions between all pairs of features, so in general there are $\mathcal{O}(N^2)$ distinct interactions to compute between two N-polytopes. Meanwhile, the AWCA potential uses GJK to find the relevant interactions to compute, so there are fewer total interactions computed; however, GJK is expensive and adds a constant additional cost for every pair potential evaluation. Therefore, in general we expect DEM to perform better for small systems of simple shapes where computing all interactions in parallel is cheaper than evaluating GJK, whereas the AWCA potential should scale better to very complex shapes because the number of distinct interactions increases much more slowly than in DEM (as we see in fig. 4.4).

The second major difference is that the two use different parallelization schemes. The AWCA is parallelized like any other pair potential, i.e. it is evaluated in parallel for every pair of particles. In DEM simulations, all possible pairs of interaction sites between each pair of particles are known *a priori*, so DEM can take advantage of this information to exploit an additional level of parallelism and compute all possible *interaction pairs* in parallel (of which there are multiple per particle pair). Since the AWCA potential uses GJK to determine the face to average over each time the potential is evaluated for a given pair, this level of parallelization is not possible. Therefore, we expect the AWCA to be especially fast compared to DEM on the CPU, whereas this advantage will be reduced on the GPU.

Figure 4.4 investigates the practical effect of these tradeoffs for various system sizes on both CPU and GPU architectures. In two-dimensional systems, the performance of the two methods is of the same order of magnitude. DEM is approximately twice as fast for triangles and remains faster than the AWCA potential for all shapes with 6 or fewer vertices (triangles to hexagons). For shapes with more than six vertices, the AWCA potential becomes faster

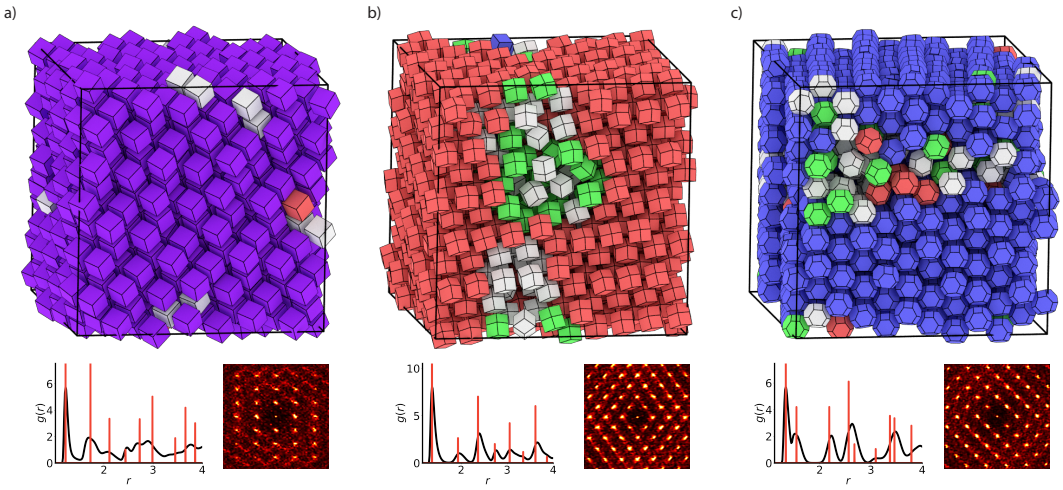

Figure 4.5 | MD simulations with the AWCA potential can be used to assemble crystal structures of anisotropic particles. Here, we validate this method by demonstrating the assembly of three of the most prevalent crystal structures found in such systems. a) A system of cubes in a nearly perfect simple cubic (SC) crystal (purple) with a few particles with local face-centered cubic (FCC) ordering (red). b) A system of rhombic dodecahedra in an FCC crystal with a small number of particles identified in local hexagonal-close packed (HCP) environments (green). c) A system of truncated octahedra in a body-centered cubic (BCC) crystal. All simulation snapshots were generated using Ovito [63] and colored using the built-in polyhedral template matching (PTM) method [145]. Red lines in the accompanying radial distribution functions (RDFs) indicate the RDF of the corresponding ideal structure. Diffraction patterns are taken along high symmetry axes of the structure.

on the CPU with a relative speedup of approximately 2x for most realistic shapes. On the GPU, however, DEM retains a performance advantage up to higher vertex counts because of its more fine-grained parallelism. As a result, for smaller systems of 1000 particles, DEM is essentially always faster by about a factor of 2. As system sizes increase and the GPU becomes saturated, however, the AWCA potential becomes faster since it has to do strictly less computation.

The performance difference is much more stark for three-dimensional systems. On the GPU, performance for the two methods is nearly equal for tetrahedra, and cubes and octahedra both experience a speedup of 2-3x when using the AWCA potential. However, more complex shapes run at least 5 times faster and usually more than 10 times faster regardless of system size. On the CPU, the AWCA potential is essentially always faster, and for complex systems the performance difference exceeds 1.5 orders of magnitude. The reason for this difference between two and three dimensions is that in three dimensions there are substantially more interactions to account for. In two dimensions there are exactly $2N^2$ vertex-edge pairs for which to perform calculations, but in three dimensions the total number of interactions is $2NF + 2EE$, where N is the number of vertices, F is the number of faces,

and E is the number of edges. Not only does this number grow much more quickly for a given N-polytope in three dimensions, but each distance calculation is more expensive than the corresponding calculations in two dimensions, accounting for the different performance characteristics between 2D and 3D. In both 2D and 3D, once system sizes are sufficiently large to saturate the GPU, we see that the relative performance of the AWCA potential increases as the effect of DEM's greater parallelism can no longer keep up with the total number of calculations to perform. System size is far less impactful on the CPU since the total number of operations scales identically for both methods.

We omit quantitative comparison to the GB potential for ellipsoids in this paper except to note that at present the AWCA potential is 1-2 orders of magnitude slower. This performance difference can be almost entirely attributed to the cost of using GJK to find the contact distance. The GJK algorithm could easily be replaced with an exact solution for uniaxial ellipsoids or more efficient specialized approximate solutions for biaxial ellipsoids [129]. Moreover, since the neighbor list constructed by HOOMD-blue is based on the major axis of the ellipsoid, it includes many ellipsoids that in fact should not interact given their orientations. One possibility for accelerating this would be filtering the neighbor list using, for instance, oriented bounding box (OBB) overlap checks [146, 147], but this method would need to be carefully implemented and profiled to understand the tradeoffs since such OBB overlap checks are also expensive calculations. Since the GJK algorithm maintains a lower bound for the distance as it iterates, the algorithm could also be modified to terminate early if the lower bound is ever larger than the distance cutoff; however, the resulting warp divergence renders this method relatively ineffective on the GPU. Since the immediate applications of our ALJ potential are for studying polyhedral particles, we leave such optimizations for future works specifically focused on studying ellipsoidal particles.

4.4.3 Assembled Crystals

As discussed in section 4.1, some of the most successful methods for studying anisotropic particles are HPMC methods that enable the study of purely entropic systems driven by excluded volume effects alone. Since such methods have been used in prior works to elucidate the assembly propensities of various shapes [14], we validate our method by showing that MD simulations using the AWCA potential can correctly reproduce these behaviors. We ran simulations of 1000 particles of various shapes in the NVT ensemble using a Nose-Hoover thermostat as described by Martyna et al [148]. Simulations were conducted at a temperature of unity in units of $\frac{k_B T}{\epsilon}$. Particles are modeled using the vertices of a unit volume polyhedron, the σ of the AWCA interaction is set to the insphere radius of

the polyhedron, and the contact interaction length is $\sigma_c = 0.15\sigma$. Particle volumes, masses, and inertia tensors were calculated from the Minkowski sum of the core polyhedral particle and the contact sphere. We initialized systems in a dilute gas and compressed them to an effective packing fraction of 0.55. Particles were assumed to be of unit density.

In fig. 4.5, we show the results for shapes with well-known assembly propensities, namely the shapes corresponding to the Voronoi cells of SC, BCC, and FCC crystal structures. The cubes in fig. 4.5a form a SC structure, the rhombic dodecahedra in fig. 4.5b form an FCC structure, and the truncated octahedra in fig. 4.5c form a BCC structure as expected. The crystal structures are identified using polyhedral template matching (PTM), which is also used to color the particles. RDFs and diffraction patterns (computed using freud [33]) are included as further evidence of structure formation.

4.5 Conclusion

Studies of anisotropic particles have appeared frequently in the literature in recent years. The role of shape in various phenomena has been examined by a number of methods, but thus far researchers have struggled to find a general framework within which such shapes could be studied using classical MD simulations. In this paper, we presented such a framework and showed how it can be used to generalize a wide range of MD pair potentials to account for anisotropic shapes. We showed how this method can be used to study the behavior of nearly-hard particle shapes, and we discussed its performance relative to other methods currently in use.

Our method enables the study of mixtures of arbitrary shapes, substantially generalizing existing dynamical methods and providing information that cannot be obtained from existing MC techniques. Concave shapes can be studied by constructing rigid bodies based on the convex decomposition of such shapes, and such rigid bodies would require far fewer particles than would be necessary using rigid bodies made of spheres. We are currently using this new method to study how particle anisotropy modulates dynamical behavior and will publish this research in a future article. Among other uses, we also foresee that this potential will be used to study the assembly behavior of attractive anisotropic particles and the properties of heterogeneous mixtures of anisotropic particles in the near future.

CHAPTER 5

Brownian Dynamics of Anisotropic Particles

This chapter is reproduced from Ramasubramani, V., Vo, T., Anderson, J. A. & Glotzer, S. C. Brownian dynamics of anisotropic particles. *In Preparation*.

5.1 Introduction

Colloidal and micron-sized particles are known to exhibit random motion when dispersed in solution. This phenomenon has been a centerpiece of modern physics ever since it was first observed in plant organelles by the botanist Robert Brown, in whose honor it is now named Brownian motion. Although this observation occurred in the early 1800s, it was not until nearly a century later that scientists like Einstein and Smoluchowski published seminal papers that formalized these observations into the language of mathematical physics. The contributions of these works were twofold: first, they related the long-time diffusion constant of a particle to its friction coefficient, now commonly referred to as the Stokes-Einstein (SE) relation [150, 151]; and second, they established that that diffusion constant could be estimated via experimental observation of the mean-squared displacement of a Brownian particle.

Such profound findings were not only crucial validation of the atomic theory of matter, they opened the floodgates to various subsequent works that greatly expanded on both the predictability and applicability of Brownian processes. First principles developments such as the Langevin equation enabled direct analysis of Brownian particles [152], and further extensions led to the incorporation of Brownian motion into modern mathematical theories of stochastic processes [153]. Concurrently, extensions were made to generalize the initial theory for translational motion to rotational diffusion via a rescaling of the long-time diffusion constant [154] as well as by direct coupling between the two for ellipsoidal particles,[155, 156] culminating in the now widely used Stokes-Einstein-Debye (SED) relation. Further generalizations were proposed for arbitrary anisotropic particles [157, 158],

but these results remained relatively obscure outside of a few theoretical studies [159–162] until they were experimentally validated in via detailed digital video microscopy [163].

Most of these works generalize the diffusion-friction relation, focusing on the idea of rescaling the long-time diffusion constant of anisotropic system while leaving the core differential form of the Langevin equation unaltered[157–162]. This approach assumes that the *characteristic* long-time diffusion behavior of anisotropic particles is identical to that of spherical particles, and that dynamic variations arising from particle anisotropy only manifest in the short-time regime. However, recent advances in computational and high resolution experimental techniques have led to novel findings pointing not only at deviations away from the ideal SE and SED behavior at long times, but also at the presence of secondary regimes within mean squared displacements (MSDs) for anisotropic particles. While the former is indicative of the expected breakdown of the long-time diffusion assumption requiring a rescaling of the diffusion constant, the latter hints at an inability of the Langevin equation to capture secondary modes of interactions intrinsic to anisotropic systems. Common throughout these breakdowns is the idea of "dynamic heterogeneity" where deviations away from ideal behaviors arise due to regions of high and low translational mobility within the system.

Recent works indicate that hard anisotropic particles, which interact solely via excluded volume interactions, may provide an ideal model system for investigating these breakdowns of traditional theories. These particles exhibit emergent attractions upon crowding as the particles aim to maximize their entropy by selectively choosing orientations that maximize translational freedom [56, 80]. Such orientational preferences result in precisely the types of dynamic heterogeneity required for violations of classical theories of Brownian motion, suggesting that hard anistropic particles are a minimal model with which to investigate extensions of these theories to anisotropic particles.

Here, we employ regular polygons as model systems to elucidate the source of deviations away from ideal Brownian motion. We start by showing how emergent, directional attractions in a system of shaped particles intricately couple translational and rotational motion such that particles' characteristic modes of translational motion are directly affected. Specifically, we find that these couplings introduce a secondary, subdiffusive regime into the MSD that is particularly evident at intermediate packing densities. In order to capture the observed simulation results, we propose an extended Langevin theory to explain the source of this additional regime. We then employ the theory to directly study the interplay between translational and rotational motions by comparing its predictions of each respective relaxation times with those measured from simulations across the various polygonal systems studied. Our study further reveals an interestingly distinct split between two different classes

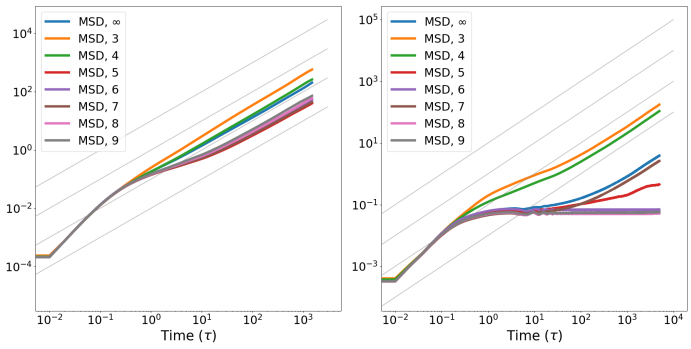

Figure 5.1 | The MSD of hard polygons at $\phi = 0.68$ (a) and $\phi = 0.75$ (b). Both panels (a) and (b) show that at intermediate packing fractions the MSD exhibits a subdiffusive regime prior to the onset of diffusion. The density at which this emerges depends on the particular shape under consideration. The blue line is the MSD of spheres. The MSD is computed using the `freud` analysis package [33].

of shapes – those that tile space and those that do not – that is also captured by the proposed theory.

5.2 Results

To study the motion of anisotropic particles, we conduct MD simulations in the *NVT* ensemble [148] using the HOOMD-blue simulation package[32]. Here, we employ a recently developed framework for the MD simulation of anisotropic potentials [34]. Briefly, the framework models interactions between two polygons as the sum of two WCA interactions [17], a central interaction $\mathcal{WCA}(\sigma, \epsilon)$ between polygon centroids and a contact interaction $\mathcal{WCA}_c(\sigma_c, \epsilon)$ between the two closest points on these polyhedra. This contact interaction behaves like a shifted WCA potential from the surface of the shape (for more details, see 34). Specifically, we consider unit area n-gons where n ranges from $3 - 9$, where we set σ to the insphere diameter of the particle and $\sigma_c = 0.15\sigma$. In HOOMD-blue's reduced unit system of $\mathcal{D}$ (distance), $\mathcal{E}$ (energy), $\mathcal{M}$ (mass), we set $\epsilon = 1\epsilon$ and a reduced temperature of $T^* = kT\mathcal{E}^{-1}$ (where kT has units of $\mathcal{E}$). Polygon area $\mathcal{A}$ is determined by the Minkowski sum of a unit area n-gon and a disk of diameter σ_c, and packing fractions are determined as $N\mathcal{A}L^{-2}$ where $N = 1024$ particles and L is the edge length of the square simulation box. Simulations are conducted with a timestep of $10^{-4}\sqrt{\mathcal{M}\mathcal{D}^2\mathcal{E}^{-1}}$.

To demonstrate effect of anisotropy on translational motion, we compute the MSD of n-gons across a range of packing fractions. At low packing fractions, where collisions are infrequent and all particles behave like disks on average, the MSD collapses onto the isotropic limit. Outside the dilute limit, however, we see clear differences between the shapes.

The most prominent differences between these polygons and disks is that a subdiffusive regime emerges prior to the onset of caging in systems at intermediate packing fractions. This regime is present in the MSD of all polygons, but its onset occurs at different packing fractions for different shapes: in particular, polygons with at least 5 sides all show this transition around $\phi = 0.65$, while triangles and squares exhibit the transition at a higher packing fraction of $\phi = 0.75$. Assemblies of these shapes are typically highly defective due to the presence of glide planes [164–166], implicating local cage formation and escape mechanisms in the emergence of this subdiffusive behavior. Since most of these polygons are known to display an intermediate bond-ordered phase between the fluid and the solid [167], the propensity for cage formation and persistence can be measured using the bond-orientational parameter

$$\psi_{k,p}^{a} = \frac{1}{p} \sum_{b \in NN_p(a)} \exp(\boldsymbol{i} k\theta) \tag{5.1}$$

to quantify k-fold symmetry in the local environment of particle a using the p nearest neighbors. In fig. 5.2, we indeed see that the transition in the $\psi - \phi$ curve occurs at the two different characteristics packing fractions for the two different types of shapes, indicating that the rise in local directional ordering is responsible for the onset of the corresponding dynamic transition.

The crucial role of directional ordering in this transition is suggestive, but disks transition smoothly from ballistic to diffusive at all packing fractions despite also having a hexatic phase, so directional ordering alone is evidently insufficient to induce the transition observed in fig. 5.1. By design, the only difference between polygons and discs is the presence of relative orientations between particles. Thus, the subdiffusive, secondary mode must be a result of translation-rotation coupling – a general feature of systems with nontrivial rotational degrees of freedom. Notably, the subdiffusive behavior exists for all polygons irrespective of the types of phase transitions that occur [167], indicating that the dynamic changes are not caused by a particular thermodynamic transition. While some previous works have captured such an intermediate regime via direct corrections to the long-time diffusion constant for hard disks, such modifications are invariably ad hoc changes that do not address the underlying cause: such an additional regime is not supported by and cannot be predicted from theories based on the traditional Langevin equation.

To systematically address this problem, we employ the Mori-Zwanzig projection formalism to develop a generalized Langevin equation that accounts for rotation-translation coupling, then solve this equation for an MSD that directly embeds rotation-translation coupling. We start with the standard generalized Langevin equation derived via the projection formalism:

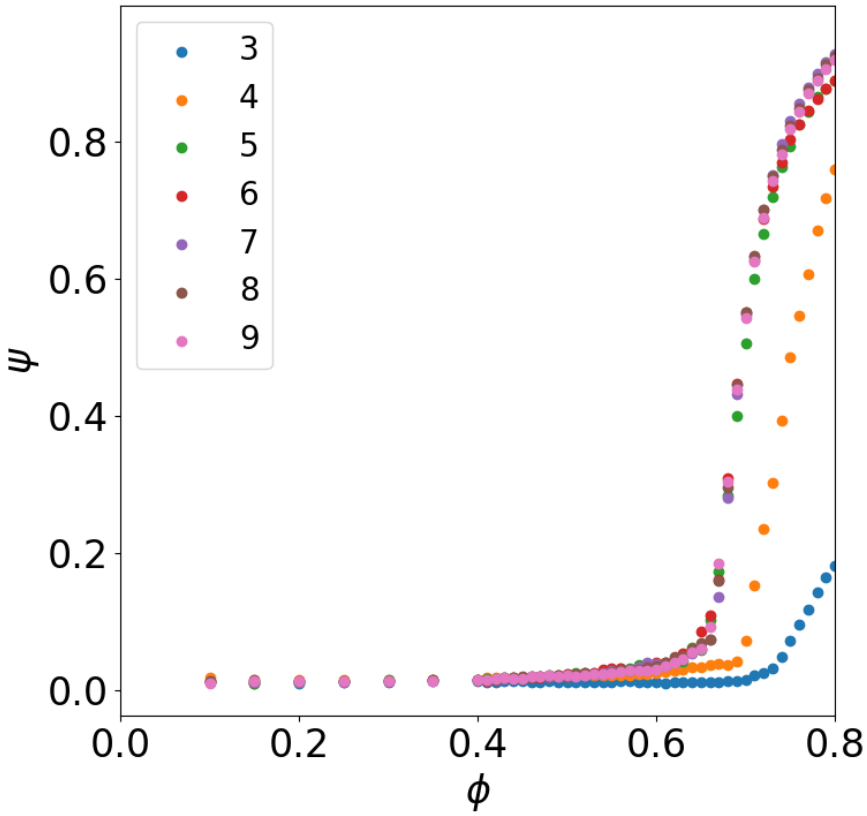

Figure 5.2 | The plot of the system-average ψ after simulations at different packing fractions equilibrate shows a clear separation of the shapes into two separate groups. Most shapes show a clear transition near $\phi = 0.68$, while triangles and squares instead show this transition at $\phi = 0.75$. These packing fractions correspond to those at which the MSD for these shapes exhibit the new characteristic behavior. We use $\psi_{4,4}$ for squares, $\psi_{6,3}$ for triangles, and $\psi_{6,6}$ for all other shapes.

$$\frac{\partial}{\partial t}\mathbf{A}(t) - i\boldsymbol{\Omega}\cdot\mathbf{A} + \int_0^t \mathbf{M}(t-s)\cdot\mathbf{A}(s)ds = \mathbf{f}(t) \tag{5.2}$$

where $\boldsymbol{\Omega}$ is the frequency matrix, $\mathbf{M}$ is the memory kernel, and $\mathbf{f}$ is the random noise. The classical Langevin equation is the result of selecting the velocity of a Brownian particle as the relevant "slow" observable, and systematically collecting all other irrelevant "fast" variables into the random noise term. Here, we follow the same approach for an anisotropic Brownian particle, which requires a multivariable observable of position, velocity, and angular velocity – $\boldsymbol{A} = \left[\vec{r}, \vec{v}, \vec{l}_p\right]$ where $\vec{l}_p$ is the projected form of the angular velocity $\vec{l}$ to ensure orthogonality: $\vec{l}_p = \vec{l} - (\vec{r}, \vec{l})(\vec{r}, \vec{r})^{-1}\vec{r}$. Applying the projection framework to this set of observables gives a system of three differential equations (see appendix B for full detail); to attain a single Langevin-like equation, we focus only on the velocity equation:

$$m\dot{\boldsymbol{v}}(t) + \xi \boldsymbol{v}(t) + \gamma_c \left(\alpha / I_n\right)^{-1/2} \boldsymbol{v}^2(t) = f(t) \tag{5.3}$$

where I_n is the moment of inertia of any given regular n-gon, f is the fluctuating force, α is a proportionality constant relating linear and angular velocities, γ_c is a translational to rotational coupling constant, and ξ is a friction coefficient.

In deriving this expression we have made use of the same approximations used for the classical Langevin equation (focusing only on the real components and assuming a constant memory kernel). In addition, to eliminate terms in angular velocity from the expression, we derived a relationship between the linear and rotational momenta of hard polygons (see appendix B), which results in the unfamiliar term in $\boldsymbol{v}^2$. Additionally, we note that in the limit of $\gamma_c \to 0$ (i.e. no rotational mode for coupling), we regain the original Langevin equation.

Like in the original Langevin equation, while most these terms can be evaluated analytically within the framework of this theory (see appendix B), the friction coefficient ξ and the coupling constant γ_c (which encodes a relationship between translational and rotational friction) must be estimated separately, a task we return to later. First, we seek a solution for this equation to study how the new $\boldsymbol{v}^2$ term changes the characteristic behavior of the MSD. No exact analytical solution exists for eq. (5.3); however, an approximate solution can be obtained via Taylor expansion about the intermediate time regime to give the following functional form for the MSD of hard polygons (see appendix B for derivation):

$$\begin{gathered}\left\langle r^2(t)\right\rangle = \frac{1}{4\gamma^2}\left\{B_1 + B_2 e^{-\xi t}\left[e^{-\xi t} + \left(\frac{1}{\xi^2 e^2}\right)t + \frac{2}{\xi^2}\right]\right\}^2 + \left(\frac{B_3}{\xi}\right)t + \frac{1}{2}\frac{B_3^2\gamma^2}{\xi^4} \\ B_1 = 4\left[\frac{B_3^2\gamma^4}{\xi^4}\right]^2 + B_2\left[1 + \frac{2}{\gamma^2}\right],\ \ B_2 = -\frac{1}{2}\left[\frac{\xi^7}{B_3^3\gamma^6}\right]\left[2\xi + \frac{2}{\xi} - \frac{1}{\xi^2 e^2}\right]^{-1},\ \ B_3 \sim \frac{\langle m v^2(t)\rangle}{m}\end{gathered} \tag{5.4}$$

where we have defined $\gamma = \gamma_c \left(\alpha / I_n\right)^{-1/2}$ for ease of notation and $\left\langle r^2(t)\right\rangle \equiv \left\langle \left(r(t) - r(t_o)\right)^2\right\rangle$.

It is instructive to look at some qualitative features of eq. (5.4). In the long time limit, we see that $\left\langle r^2(t)\right\rangle \sim \left[B_3 \xi^{-1}\right] t$. Expanding the exponential dependency in eq. (5.4) and isolating linear terms reveals a secondary linear time dependence of the form $\left\langle r^2(t)\right\rangle \sim \left[(e^{-2} - 2\xi)\left(2\xi^{-1} - B_1 B_2^{-1}\right)\left(\sqrt{2}B_2\xi\gamma\right)^{-2}\right] t$. Combining both yields

$$\langle r^2(t)\rangle \sim \left\{ \frac{B_3}{\xi} + \frac{(e^{-2} - 2\xi)\left(2\xi^{-1} - B_1 B_2^{-1}\right)}{2B_2^2\gamma^2\xi^2} \right\} t \tag{5.5}$$

By inspection, the first term on the right hand side of eq. (5.5) is identical to SE results and the second term is a coupling between rotation (γ) and translation (ξ). Thus, the behavior of eq. (5.4) can be interpreted as perturbations about the SE limit due to intermediate rotational-translational coupling, giving rise to a net "average" diffusion constant that approaches the SE limit at long times. This physical picture is analogous to previous theoretical approaches employed for Brownian motion of thin rods and ellipsoids. In the limit of no coupling, $\gamma_c \to 0$, $B_2 \to \infty$, and eq. (5.5) converges to the traditional SE results of $\langle r^2(t)\rangle \sim [B_3\xi^{-1}]\, t \sim [kTm^{-1}\xi^{-1}]\, t$. For $t << 0$, the exponential terms vanish and eq. (5.4) is dominated by the t^2 dependence. At intermediate times, however, the terms involving negative exponentials become non-negligible, indicating a source for the subdiffusive regime.

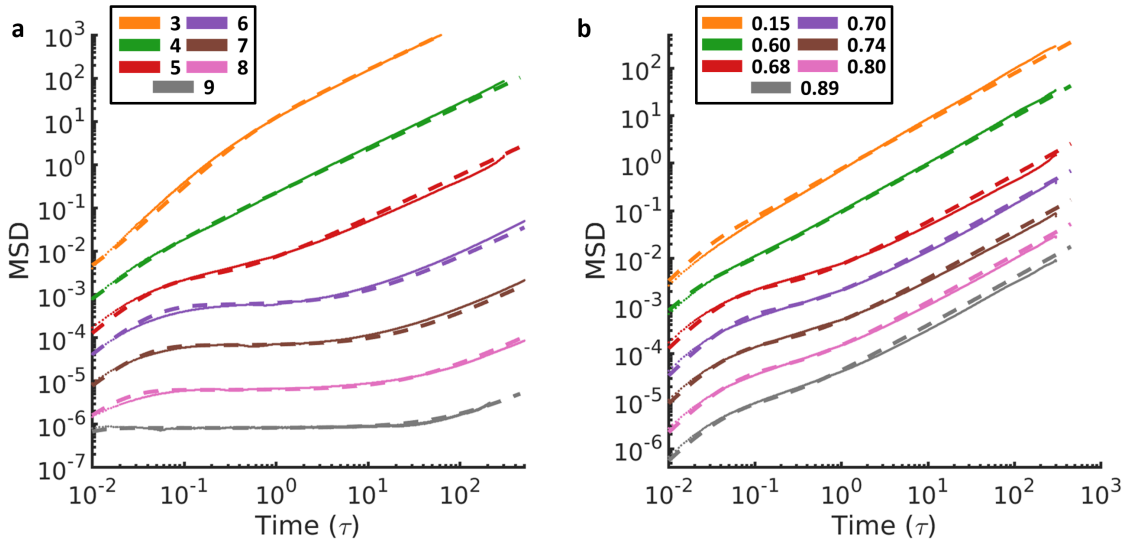

Figure 5.3 | Comparison of measured MSD with theory prediction. a). MSD for n-gons at packing fraction $\rho = 0.68$. b). MSD for 5-gon at various packing fractions. Scatter points are from simulation, dashed lines indicate theory predictions. MSDs are artificially offset in order to clearly show comparison between theory and simulation.

As shown in fig. 5.3a, our theory is able to capture the secondary regime observed in simulations as well as its emergence as a function of increasing packing fraction. To gain deeper insight into the behavior of this system, we now inspect more closely the characteristic relaxations in translational and rotational modes. While comparison of the MSD provides a rough estimate of how well eq. (5.4) captures the relevant dynamics for systems of hard polygons, a more rigorous test of the theory is to evaluate how well it

predicts characteristic relaxation times in autocorrelation functions. Since our primary interest is the translation-rotation coupling evident in the MSD, we focus on estimating τ_{trans} and τ_{rot}, the translational and rotational relaxation times. These quantities are easily computed from simulations by computing autocorrelation functions and fitting exponential decays to determine characteristic time scales.

A plot of the ratios of the two relevant timescales as a function of packing fraction across all n-gons studied is shown in Fig. 5.4a. In the limit of low packing fraction, this ratio is very large, reflecting that velocity correlations take a long time to die off in sparse systems with few collisions, whereas rotational correlations decay quickly in the absence of nearby particles preventing free rotation (assuming that initial angular velocities are finite, for instance distributed according to a Maxwell-Boltzmann distribution). Conversely, this ratio tends to zero for high packing fraction as rotations are now arrested and velocities are effectively random due to the high number of local collisions. One particular feature of interest is what appears to be a crossover in the relaxation ratios near $\rho \sim 0.75$ between 5-gons and 6-gons. In our derivations of Eq. 5.3 – 5.4, the moment of inertia I_n completely defines the shape dependency. Since I_n changes monotonically as a function of the number of sides for regular polygons, extracting the relevant timescales from Eq. 5.4 will not capture such a crossover (see appendix B). In order to address this deficit, we note that the local ordering that a particle feels directly influences the balance between translational and rotational relaxation times. One possible extension to our theory in order to impart some knowledge of local caging into the governing equations involves the reintroduction of the frequency term – $i\boldsymbol{\Omega} \cdot \mathbf{A}$. Doing so results in the following modification of Eq. 5.3

$$m\dot{\boldsymbol{v}}(t) + \left[\xi + \eta \left(\alpha / I_n\right)^{-1/4}\right] \boldsymbol{v}(t) + \gamma_c \left(\alpha / I_n\right)^{-1/2} \boldsymbol{v}^2(t) = f(t) \tag{5.6}$$

where η is a composite term encapsulating the projection of the rotational mode of a polygon onto its positional mode (see appendix B). The general solution to Eq. 5.6 is analogous to Eq. 5.4 with ξ terms replaced with $\xi_p \sim \xi + \eta \left(\alpha / I_n\right)^{-1/4}$. In order to approximate the characteristic relaxation times using theory, we perform a simple scaling argument and look for the crossover time for which the rotational-translational coupling term is dominant (second term in Eq. 5.5). This provides an approximation for τ_{rot}. A similar balance for when the standard diffusive term, B_3/ξ_p, dictates long-time dynamics provides the translational characteristic time τ_{trans}. The functional forms for τ_{trans} and τ_{rot} are thus derived to be

$$\tau_{trans} = \xi_p^{-1} \mathcal{W}\left(\frac{B_1 B_2^{-1} - 1 + \left(\xi_p^{-1} - \xi_p^{-2}\right)\left(e^{-2} + 2\right)}{\gamma B_2^2 B_3 \xi_p^{-2}}\right) \tag{5.7}$$

$$\tau_{rot} = \xi_p^{-1} \mathcal{W}\left(\frac{B_1 B_2^{-1} - 1 + \left(\xi_p^{-1} - \xi_p^{-2}\right)\left(e^{-2} + 2\right)}{\left(e^{-2} - 2\xi_p\right)\left(2\xi^{-1} - B_1 B_2^{-1}\right)\xi_p^{-3}} \right) \tag{5.8}$$

where $\mathcal{W}(x)$ defines the Lambert W-function.

Computing the ratio $\tau_{trans}\tau_{rot}^{-1}$ using Eq. eqs. (5.7) and (5.8) for all n-gons across different packing fractions shows good agreement between theory and simulation (fig. 5.4b), suggesting that frequency corrections can serve as a first-order perturbative correction for relaxation behaviors within the secondary regime.

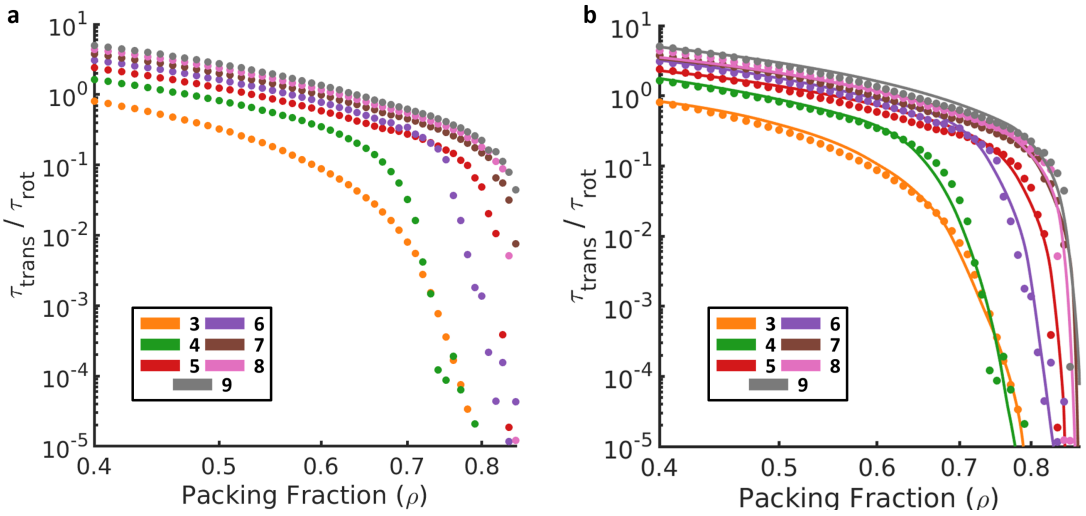

Figure 5.4 | Ratio of translational to rotational characteristic time as a function of packing fraction. a). Measured $\tau_{trans}\tau_{rot}^{-1}$ from simulation. b). $\tau_{trans}\tau_{rot}^{-1}$ as predicted from Eq. 5.7-5.8 (solid lines) overlaid with simulation (scatter points).

Interestingly, the ratio between translational and rotational relaxation drops off more quickly and with a different characteristic slope for 3-, 4-, and 6-gons as compared to other polygons. By definition, the earlier drop in $\tau_{trans}\tau_{rot}^{-1}$ indicates that these particles stabilize their orientations at lower packing fractions relative to the packing fractions where they stabilize their positions. The behavior of hexagons is particularly interesting: while ψ_6 in fig. 5.2 is more in line with that of other shapes, the relaxation time ratio fig. 5.4 indicates that the characteristic behavior of hexagons is closer to that of triangles and squares. This suggests that while the absence of glide planes forces hexagons to localize their positions at lower packing fractions than triangles or squares, the fact that their symmetries are commensurate with tiling space also leads to a concurrent stabilization of orientations at much lower packing fractions.

5.3 Conclusion

In summary, we employ our recently developed framework for MD simulation of anisotropic particles [34] to study the Brownian dynamics of 3- to 9-gons across a range of packing fractions. We observe the emergence of a secondary, subdiffusive regime before the onset of full caging across all polygons studied. In order to explain the observed trends, we utilize mode-coupling type approaches to directly build the effect of translational and rotational coupling into the Langevin equation commonly used to study Brownian motion. Our new theory provides a generalized equation for the MSD whose functional form is able to capture the trends observed simulation. We then employ our framework to study relaxation times across all n-gons, showing that our approach can additionally capture the correct translational and rotational behaviors. Lastly, we note a distinct split in the characteristic trends of polygons into two shape classes suggesting that some shapes exhibit different levels of emergent directional interactions upon crowding polygons. Even for simple, regular polygons, our study reveals that a more intimate coupling of translational and rotational motion is required to fully understand their dynamical interactions. These results hint that rich, newer physics can potentially lie in wait upon extension to 3-dimensional systems. Proper understanding and elucidation of such hidden interactions not only enrich our understanding of anisotropic dynamics, but can also provide crucial insights needed for future assembly and engineering applications of shaped building blocks.

CHAPTER 6

How to Professionally Develop Reusable Scientific Software — and When Not To

This chapter is reproduced from Adorf*, C. S., Ramasubramani*, V., Anderson, J. A. & Glotzer, S. C. How to Professionally Develop Reusable Scientific Software—And When Not To. *Computing in Science Engineering* **21.** (*these authors contributed equally to this work), 66–79. ISSN: 1521-9615 (Mar. 2019). CSA and I worked together on all parts of this work.

6.1 Introduction

As computational science continues to play a rapidly growing role across nearly all scientific disciplines, quality scientific software becomes ever more critical to research productivity[168]. The development of such software is impeded by the fact that most scientific software is developed by non-experts for whom the primary product is not the software, but the resulting science. The existing incentive structure, which is geared towards rapid publication of scientific results, tends to motivate the creation of single-use software. In addition to being highly targeted at a specific application, single-use software often also disregards established software development best practices such as proper version control and documentation, impeding its reuse for future studies and ultimately hindering long-term scientific progress.

Software solutions for many-body simulations present a useful case study of this development pattern. Over time, the range of scales and physical phenomena of interest have led to the creation of numerous packages with overlapping yet distinct feature sets. Despite this fragmentation, however, certain packages have attracted much larger user communities than others, suggesting a slow progression towards specific reusable solutions.

We present an approach, *lazy refactoring*, for accelerating this progression while ensuring that code development is always targeted at an *immediate* scientific objective. This

approach—especially suited for researchers, who need to reconcile sustainable software development with the need to make immediate scientific progress—advocates that initial development should always lead to single-use code, but this code is refactored into a reusable solution *as soon as* two further uses for it are found [169]. We discuss how software standards such as modular design and well-defined interfaces, tools such as version control system (VCS) and continuous integration (CI), and code proper documentation help facilitate lazy refactoring. With this context, we offer a concrete methodology to help determine when and how to write new code.

As an example of lazy refactoring, we present the development progression of software developed by the Glotzer Group at the University of Michigan. We are a diverse, collaborative research group comprised of 30+ graduate students, post docs, and research scientists from chemical engineering, materials science, physics, and other disciplines. Roughly half our members join the group with little to no simulation experience, and very few have developed professional quality code prior to joining. As such, we are a representative sample of the computational research community within academia. We show how lazy refactoring evolved alongside our development efforts and demonstrate its use to create a domain-specific, reusable, and loosely integrated software stack. We close with a brief discussion of how we train group members to foster the presented techniques.

6.2 Developing computational solutions

The goal of lazy refactoring is to minimize the total resources used on tool development for a computational problem within the context of a preexisting *software ecosystem*, which refers to the set of all available software. While much of this code will come in the form of clearly defined, well-documented, and consistently maintained software *packages*, the ecosystem will also contain harder to reuse special-purpose codes that may nevertheless be useful beyond their original intent. This second kind of software, commonly referred to as a *prototype*, generally lacks clean interfaces and documentation, and does not have explicit names or release versions. While packages are easily reusable, prototypes offset this benefit by being faster to implement and requiring less effort to maintain. Borrowing from well-established agile software development techniques such as extreme programming (XP), lazy refactoring ensures that the ability to respond rapidly to changing requirements and maintaining working software are prioritized over planned design and comprehensive documentation [170]. One critical difference here is that the programmer of scientific software aimed at solving a particular scientific problem is usually also the only customer.

In principle, total development efforts are minimized by using existing code whenever

possible, and, when new code is required, expending effort to make code reusable only when reuse is expected. From this perspective, solving a computational problem starts by decomposing it into components that each require a software solution. This process helps identify which parts are already solved within the ecosystem, which ones can be addressed by adapting existing tools, and which ones require entirely new code. Then, the missing components can be assessed to determine which ones merit developing reusable packages and which ones should be developed as prototypes. We use the term *adapter* to refer to code whose primary purpose is to interface between other packages.

In practice, however, making an optimal decision about when to develop reusable code is generally impossible. The problem scope may be ill-defined and future project plans change, invalidating total resource estimates and making it difficult to determine which components are worth developing as packages. In this light, it is clear that developing reusable code packages any time reuse is foreseen is often an inefficient use of resources.

Instead, we recommend the more agile lazy refactoring approach, in which *all* missing code is developed as a prototype in the most convenient manner possible, e.g., in Python [171]. Refactoring of this code should always be motivated by its imminent application to new problems, but refactoring should be preferred over new prototype development once such applications are identified. Specifically, we advocate following the **Rule of Three**: a prototype should be refactored as soon as a *third* application is found. We illustrate this approach in fig. 6.1, where there are two alternatives to address the additional software needs of Problem B after previously solving Problem A. If no prototypes exist, the problem should be solved with rapid development of prototypes **x** and **y**, tailored to the specific problem (*Alternative A*). If two or more prototypes already exist, then the gap should be closed by refactoring these into package **d** (*Alternative B*). We argue that, on average, this approach minimizes total software development effort since the resources required for the design, implementation and maintenance of software packages are only expended once the future reuse potential is sufficiently apparent.

The advantages of lazy refactoring are manifold:

1. Total resource investment into solving one problem is minimized.

2. Refactoring a prototype into a package is usually easier than writing a package from scratch because it postpones developing interfaces and defining the scope until the problem is better understood.

3. The package can be validated against the original prototype.

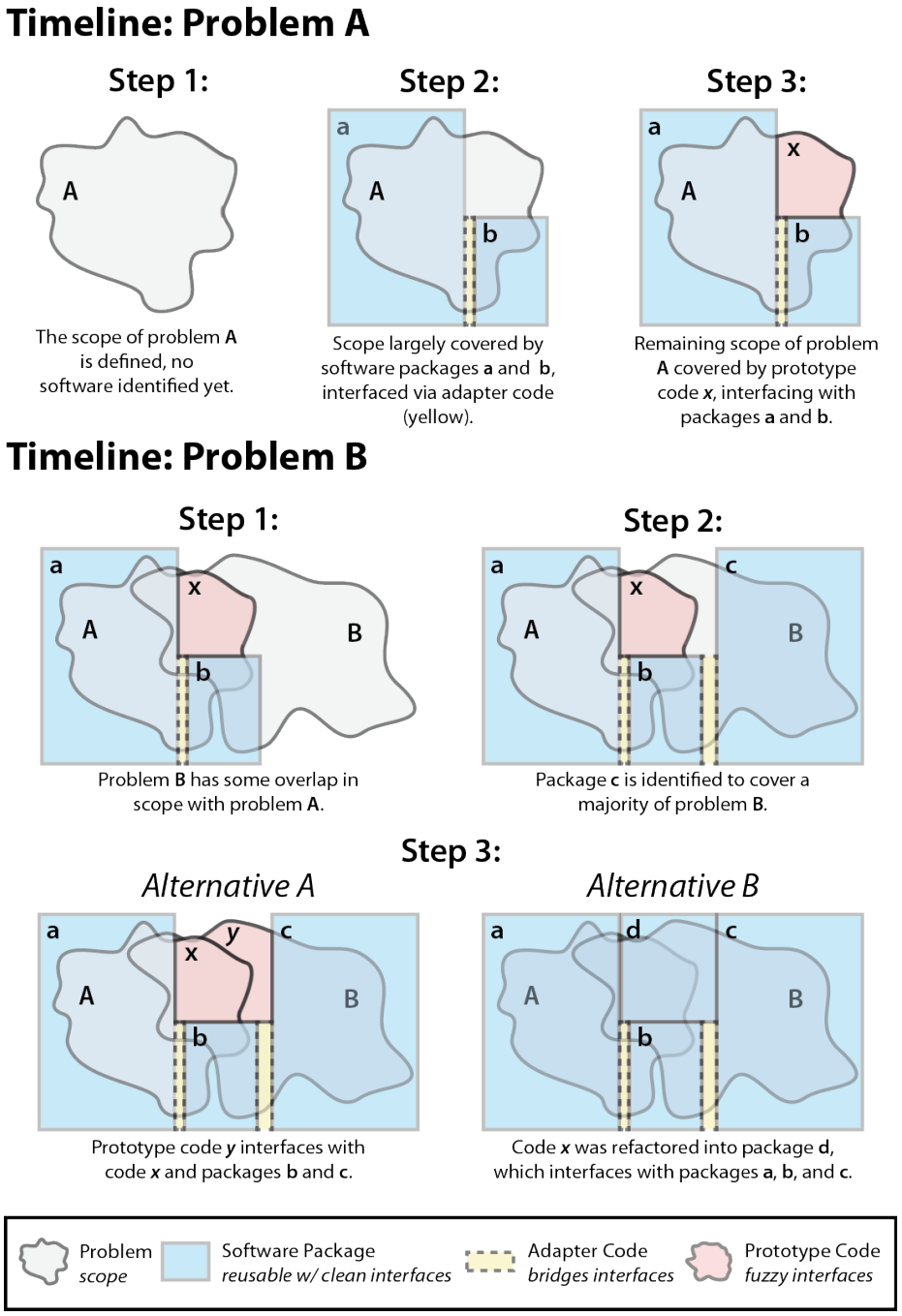

Figure 6.1 | This figure illustrates the code development progression associated with solving two related computational problems. Problem **A** is solved using packages **a** and **b** in combination with some adapter code and the prototype code base ***x***. Problem **B** is determined to overlap with problem **A**, and large portions of it are already solved by packages **b** and **c**. There are two alternatives for addressing the remaining software needs for problem **B**. The first alternative is to reuse ***x*** and develop prototype ***y*** to fill the gap (*Alternative A*). The second approach is to refactor prototype ***x*** into package **d**, which interfaces with packages **a**, **b**, and **c** and fully solves problem **B** (*Alternative B*).

4. The decision to refactor can account for actual projects that arise rather than attempting to estimate future reusability.

Lazy refactoring does carry significant risks: key personnel involved with the original development may no longer be available for refactoring, or the prototype may grow so large and opaque that refactoring itself presents a significant barrier. Therefore, the Rule of Three should be applied rigorously such that refactoring occurs in a timely manner and the original developers can be heavily involved in the refactoring process. This method requires that some quality controls be imposed on even initial prototypes to ensure that refactoring remains possible.

6.3 Principles, Tools and Practices

In this section, we expound upon several principles, development tools and best practices that aid in effectively applying lazy refactoring. As we will show, these principles, tools, and practices are also useful barometers for evaluating preexisting software solutions, as will be shown later.

6.3.1 Principles

Modularity, the core design principle that we advocate, is a well-known standard enshrined in the UNIX philosophy that naturally leads to the development of tools with clearly delimited scopes and well-defined Application Programming Interfaces (APIs), each of which confer significant benefits. Code with limited scope can be developed more quickly, and it may preclude further time expenditures if the initial work proves sufficient for the task at hand. The API-driven design model also ensures that different components can be easily used independently of one another, making it easier to generalize and refactor specific pieces or integrate them with other tools. This feature is especially crucial in science because specific research applications often require combining previously unrelated, highly domain-specific software.

The first step towards modularity is defining a clear division between software components. Defining these divisions immediately suggests the appropriate scopes for each component, the specific interoperability requirements, and the necessary API to meet these requirements. In section 6.5 we show how our group's stack is largely composed of modular packages with well-defined APIs that employ standard data formats and operational paradigms for easy integration.

Our other guiding principle is the public release of software source code. Although some code may be privately maintained to preserve a competitive advantage in the short term, the need to validate scientific results and the need to integrate disparate software tools make the ability to view source code more valuable than ever. These are strong incentives for open-source scientific software development.

6.3.2 Tools and Best Practices

We employ a number of tools and best practices to ensure that our code adheres to the principles expressed in the previous section. VCSs, which enable the robust management of a project's evolution over time, are especially important in scientific code development due to its collaborative nature. VCSs enable scientists to work in parallel while maintaining stable, working versions of code at all times. This collaborative process is much better suited to the decentralized model of distributed version control systems (DVCSs), such as Git, than to the centralized approaches of traditional VCSs like Subversion (SVN). The benefits of a controlled, shareable central repository can be recovered by hosting code on public repositories such as GitHub, Bitbucket, or GitLab.

Some code quality checks can be automated using code linters, such as Flake8 for Python, splint for C, or cppcheck for C++. Higher level checks can solicit input by using, for instance, pull request-based workflows to require human reviews before new code is added to an existing code base. Code should include unit and integration tests, which increase confidence in code correctness and reduce the likelihood of introducing breaking changes. Code should also employ CI, the automatic execution of tests to ensure that any errors or issues are caught before spreading to stable code versions. Many CI services, such as Jenkins, CircleCI, and Bitbucket Pipelines, integrate with repository hosting services and are free to publicly available repositories, further incentivizing open-source development.

To maximize reusability, software must be widely disseminated and painless to install. At the very least, source code should be publicly available, but standardizing modes of distribution through, e.g., package managers, is the preferred mode of making the software readily available. For example, most open-source packages published by our group are distributed via the Python Package Index (PyPI) and Anaconda cloud repositories. For distribution on specialized high-performance computing (HPC) environments, we currently host Docker images on the Docker container hub, which we deploy using Singularity on environments such as the Texas Advanced Computing Center (TACC) Stampede2, Pittsburgh Supercomputing Center (PSC) Bridges, and San Diego Supercomputer Center (SDSC) Comet clusters available through Extreme Science and Engineering Discovery Environment

(XSEDE) [172].

Software must come with effective documentation, which includes both guidance for high-level usage and more detailed information on the package's various components such as its API. Documentation quality can be greatly improved by taking into account user feedback, which can be obtained through surveys and focus group sessions along with more asynchronous mechanisms like issue trackers, mailing lists, and chat rooms. As with CI, online documentation hosting services such as ReadTheDocs are free for open-source software to simplify the publishing of high-quality documentation.

Finally, software must be appropriately licensed. Terms and conditions for using unlicensed code may be unclear, so software (whether open- or closed-source) must have licenses that permit reuse. For open-source software, for instance, the Open Source Initiative (OSI) approves certain licenses as providing sufficiently free usage and modification. Our group typically licenses software under the BSD 3-clause or the MIT license.

6.4 Applying lazy refactoring

While we have described an agile approach to code development for a research problem, to this point our formulation has been purposely abstract, providing only a conceptual description and high-level guidelines. We now describe the practical application of lazy refactoring to solving a particular problem. Using the ideas espoused by the previous section as a guide, we identify specific attributes by which existing software may be assessed and that should be incorporated into any new software. We then present a process for rigorously and systematically determining exactly when and how much new code development is merited, using the identified attributes to assess the usability and integrability of external code bases. Note that the needs of a typical project will evolve over its lifetime, and will therefore require applying this heuristic multiple times.

6.4.1 Critical Attributes

Any software that is considered for integration into a problem solution workflow should be assessed according to the following critical attributes:

1. *Scope*: Does it solve the problem at hand?

2. *Integrability*: How well does it integrate into the existing software stack, including external tools upon which this stack relies, and how much work would be required for integration?

3. *Stability*: Does it have a well-defined, static API, is it largely error-free, and is it likely to be maintained into the foreseeable future, or at the very least, over the project lifetime?

4. *Security*: Does it pose any security risks?

These attributes are discussed in detail in the following subsections.

6.4.1.1 Scope

In order for software to be useful, it must solve the problem at hand. Although this statement appears trivial, its simplicity hides some important nuances. A complete solution must satisfy, for instance, scalability and performance requirements. These factors are context-dependent; for example, simulations may need to scale to leadership-class HPC platforms and run fast even for systems with millions of particles. Such considerations may disqualify a candidate code base that solves the problem in a limited set of cases.

6.4.1.2 Ease of integration

The next step is determining how easily the software can be integrated into existing workflows. As discussed earlier, license compatibility is a significant factor in this determination and may be a reason to prefer open-source software. Open-source software has the additional benefit that the code can be inspected and adapted if needed (see section 6.4.2.2).

In addition to license compatibility, software compatibility also encompasses modularity, data types, and file formats. In order for different components of a software pipeline to work together, they must communicate in some common language, i.e., they must be able to exchange data in a standardized manner. For example, all numerical analysis software implemented in Python should make use of NumPy[171], which is the *de facto* standard for arrays of numerical data within the Python ecosystem.

6.4.1.3 Stability

High quality software must also be sufficiently stable. This means that API and code paths are set, further dependencies are unlikely to be introduced, and existing features are provided with the implicit or explicit promise that they will continue to work. Furthermore, stable packages must be well-supported and have good documentation, and they should have active and responsive developers. Additional indicators of stability are active mailing lists, issue trackers, and forums.

6.4.1.4 Security

Last but not least is the consideration of security risks posed by potential external software solutions. For instance, any software that requires a privileged execution context may be problematic, especially in HPC environments. Additionally, software that communicates with internet servers, including cloud computing services, may be unsuitable for handling sensitive data. These services are commonly not compliant with rules and regulations implemented by the Health Insurance Portability and Accountability Act (HIPAA) in the U.S. and similar legislation in other countries.

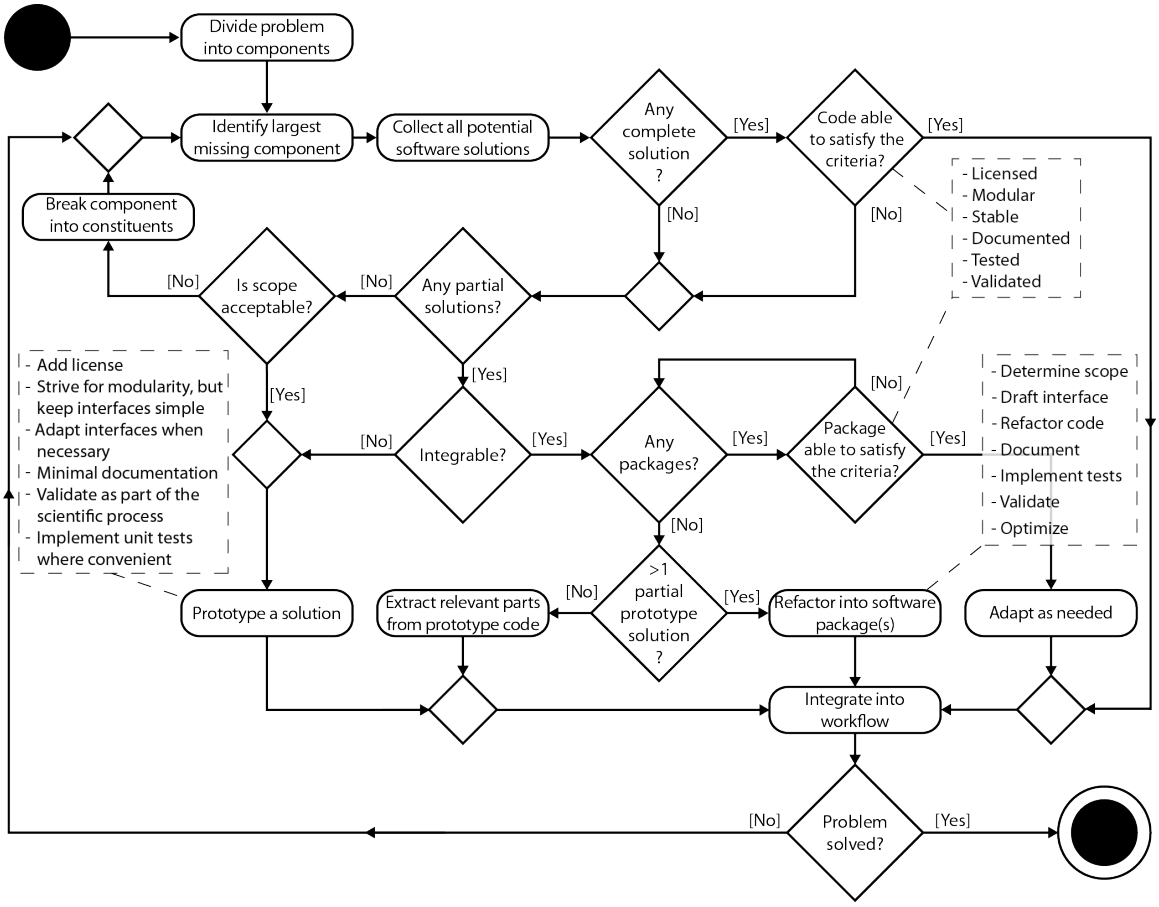

Figure 6.2 | This flow chart depicts the core elements of a basic decision tree for the development of workflows that solve a specific computational problem. In summary, we break down the problem into individual components that may or may not have existing solutions. Existing solutions are evaluated for use, and where appropriate, prototype solutions are either written or existing prototypes are refactored into packages. The decision on what solutions to include is largely based on the criteria outlined in section 6.4, which serve as basic guidelines for evaluating existing solutions for their integrability into the proposed workflow.

6.4.2 General Heuristic

We now formulate a decision tree to guide the software-related decisions involved in solving a scientific problem (see fig. 6.2). This process involves decomposing a problem into components that can be solved sequentially. A critical part of the process is the usage of preexisting *partial* solutions, which are often originally developed as single-use prototypes and are ripe for refactoring. Although such refactoring can be difficult, eventually it will pay dividends as the quality of the software ecosystem increases and software becomes easier to refactor and adapt. NumPy is one example of how two partial solutions, Numarray and Numeric, were consolidated into an improved package that shares both their strengths.

In some circumstances, writing partially redundant software may be justifiable. Git is one example of a highly successful package that was developed despite the existence of other VCSs because those tools failed to support highly distributed, branch-heavy workflows. Any such development, however, should be driven by a clear shortcoming of existing software.

6.4.2.1 Checklist for software integration

Once potential complete or partial solutions have been identified, they must be individually checked to see whether the following criteria can be satisfied:

1. Ensure that the software is appropriately licensed to allow integration and modification.
2. Ensure that the architecture and interfaces are designed for easy integration.
3. Determine whether the software is actively maintained and whether code or documentation contributions would be accepted.
4. Check whether the API is stable to estimate future maintenance effort.
5. Confirm that all interfaces, the architecture and—in case modifications are required—the source code are sufficiently documented.
6. Establish that the majority of the code base is properly tested and validated.

In many cases, software that does not satisfy these criteria can be brought into compliance with sufficient modification, either by the user (e.g. by writing documentation, adding tests, or maintaining a fork) or by contacting the maintainer (e.g. to add a license). If such modification ultimately proves insufficient to satisfy these criteria, however, the software should be discarded as a solution for the problem at hand. Note that all criteria outlined above

extend to dependencies as well, so code bases with many dependencies must effectively clear a higher bar.

6.4.2.2 Guidelines for package adaptation

For packages that provide a near-complete solution but require some adaptation to completely address the relevant part of the problem, we have developed the following guidelines:

1. Plan development: Unless it conflicts strongly with the protection of intellectual property, obtain feedback on the proposed modifications and extensions from the urrent package maintainer.

2. Testing: Before modifying any code, implement any missing tests to ensure adequate coverage of code paths that may be modified.

3. Validation: Add integration tests as needed to completely verify the correctness of the original software when applied to the problem at hand.

4. Modularity: Refactor the code base such that all parts that require modification are mostly separated from those that do not.

5. Adaptation: Write only as much code as is needed to address the problem at hand without aiming for too much generality.

6.4.2.3 Steps for refactoring prototypes into packages

If there are no applicable packages, but there are *multiple* existing prototypes, then these should be refactored into a package. Before refactoring, all related prototypes and packages should be analyzed for interfaces, overlaps, strengths and weaknesses, with a special focus on relevant packages that failed the tests of the checklist for software integration. Such software, which may have failed due to, e.g., licensing issues, likely contains a great deal of expertise and know-how that is invaluable for the design of any new package. It may also be worthwhile to seek additional information on these packages through public forums or issue trackers, or by reaching out to developers or experienced users for further insight. The new package may then be implemented according to the following steps:

1. Determine the scope and the software architecture. Identify the core logic and then build layers around it for additional functionality such that dependencies only point inwards as described in the The Clean Architecture [173] principle.

2. Draft the end user interface before implementation. Ensure that the most prominent use cases are supported by the drafted interface design.

3. Replace the prototype code with the package step by step within the original applications. Ensuring that the new code produces identical results at each step is an important part of the validation process.

4. Document all functions and interfaces well such that a user can understand and use the package without needing to consult any of the developers.

5. Write unit tests for all core functions and add integration tests for the main applications of the code.

6. After ensuring function and correctness, optimize critical code paths (*and only those paths*) if needed.

6.4.2.4 Guidelines for developing a new prototype solution

If new software is required, we impose the following minimal standards on the resulting prototype code to simplify its future refactoring if needed:

1. License: Any code base should be licensed. Such a license may, for instance, explicitly permit internal reuse and require acknowledgment of the original author. To simplify matters, a research group could agree on a general license that is assumed to apply to all code that is not explicitly licensed otherwise. Organization-wide policies regarding intellectual property will apply.

2. Modularity: Code should be developed as modularly as possible to simplify potential refactoring at a later stage. However, this modularity should not come at the expense of simple, problem specific interfaces that streamline solutions to the problem at hand.

3. Stability: All code should be part of a version-controlled repository. Interfaces can be changed whenever convenient.

4. Documentation: The code should have enough internal documentation to allow developers to modify it as well as sufficient API documentation for prospective users familiar with the code base.

5. Validation: The code base should be validated as part of the research process, ideally against known results.

6. Testing: Unit tests are not necessary unless they aid in the development process.

6.5 The Glotzer Group Software Stack

We now trace the growth of the core software developed within our group. Although our initial development followed a far less structured pattern than lazy refactoring, the challenges we encountered and the experiences we accumulated informed the guidelines that we now follow. By describing our internal software ecosystem (see fig. 6.3), we aim to motivate our approach and provide a blueprint for how to use it to create a powerful, sustainable, and integrated software stack for domain-specific computational research.

For brevity, we restrict ourselves to software that is directly involved with the generation, organization and transformation of data. Therefore, we omit discussion of, for instance, all software used for producing illustrations or presentations. We also avoid discussing operating systems aside from noting that we primarily use UNIX-like systems such as various Linux distributions and Mac OS X.

6.5.1 Simulation

The genesis of our lazy refactoring approach lies in the consolidation of "single-use codes" written for various particle simulation techniques into a package called HOOMD-blue [77, 78]. HOOMD-blue evolved from HOOMD [77], a general purpose MD program developed for bead-spring polymer models that was the first MD package to run completely on GPUs. HOOMD-blue exemplifies the paradigm of growth by refactoring and incorporation of external code. Today, HOOMD-blue runs many flavors of MD and MC simulations on both CPUs and GPUs and it supports a rich variety of phenomena that were not originally envisioned. These features were all developed separately for specific research purposes, and the original implementations underwent many revisions before incorporation into a public release of HOOMD-blue.

Consolidating our development efforts has reduced development time, increased reproducibility, and improved code quality, allowing us to improve upon our original prototypes. Moreover, we have incorporated lessons learned from modifying preexisting toolkits that were not modular enough to allow easy modification or integration with our workflows. HOOMD-blue's highly structured, object-oriented design maximizes ease of modification, enabling the continued integration of new tools such as the HPMC[79] module from a pre-existing code base [174].

With internals written in C++ and Compute Unified Device Architecture (CUDA), HOOMD-blue is one of the fastest particle simulation tools available; it is one of NVIDIA's official benchmarks for new GPU hardware. Unlike toolkits such as the Large-scale Atomic/ Molecular Massively Parallel Simulator (LAMMPS) [175] or the GROningen Machine

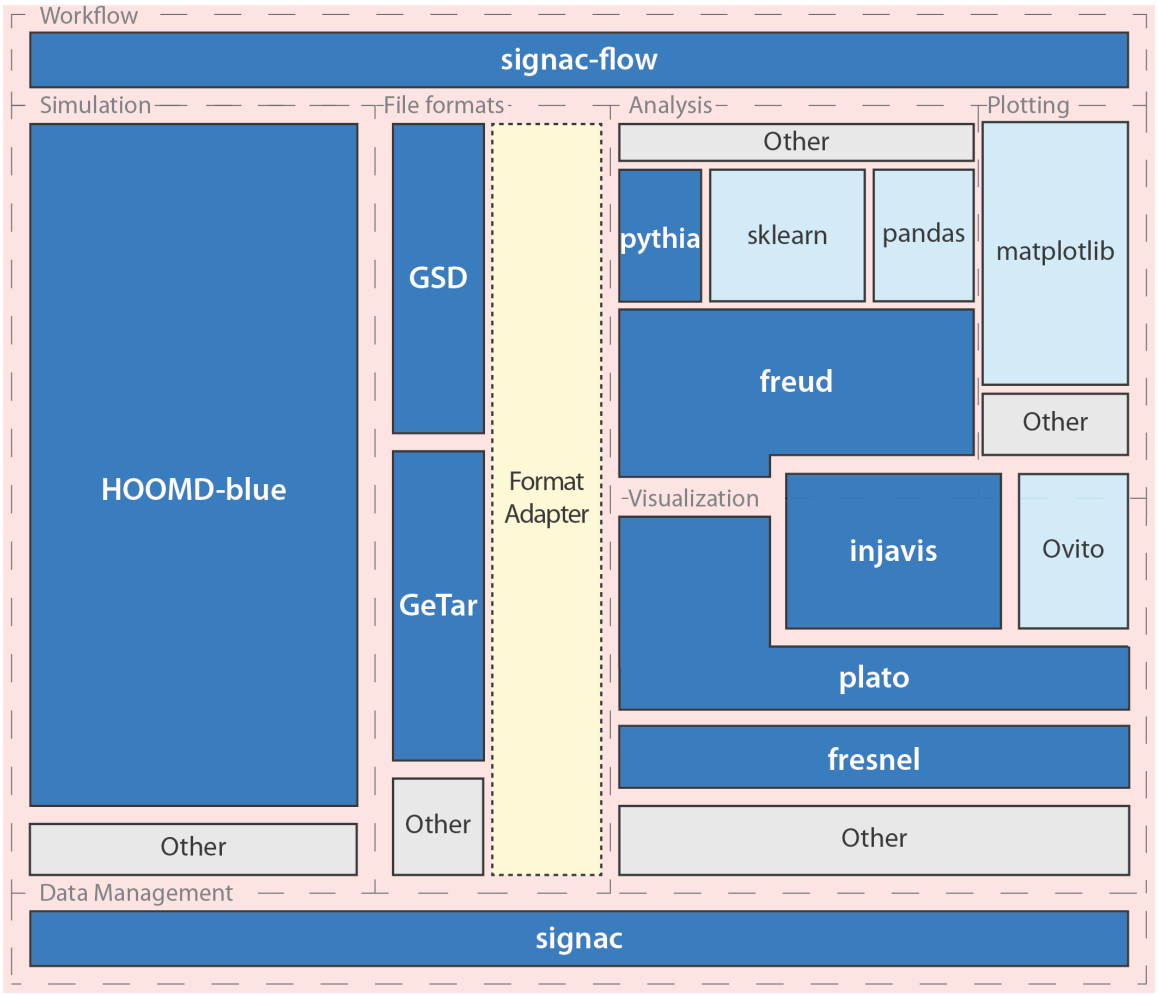

Figure 6.3 | We illustrate here the relationships between the various components of our software stack. Software packages are grouped according to their overarching functions. Interactions are denoted by adjacency. Software that is developed within our research group is shown in dark blue with white labels. Workflows for computational projects are typically organized using signac-flow, which links together simulation, analysis, and plotting. The vast majority of our simulation needs are provided by HOOMD-blue, but we occasionally utilize other tools, for instance to perform atomistic simulations of proteins. Which visualization toolkit is used depends on the specific simulation, and occasionally also on whether the toolkit has the required analysis capabilities built-in. More generally, the problem at hand dictates the appropriate analysis software packages. We use various plotting tools, especially Matplotlib, to visualize the evolution of both raw quantities (such as total system energy) as well as the outputs of more complex analyses. All data generated throughout this process, including raw simulation trajectories as well as the outputs from complex analyses, is stored and managed using signac.

for Chemical Simulations (GROMACS) [176], which use a specialized file format for simulation configuration, HOOMD-blue offers a full-featured Python interface for greater flexibility and ease of integration with other tools. An open-source toolkit, HOOMD-blue has benefited greatly from numerous contributions from its user base; at the time of writing, 29 developers external to the group have contributed to HOOMD-blue. HOOMD-blue also uses a number of external code bases, including: pybind11 for exporting C++ classes to Python; CUDA, cub and Thrust to utilize GPUs; the MPI to scale across multiple nodes; LLVM for JIT compilation; the Eigen linear algebra library; and cereal to serialize data for communication.

6.5.2 Data Analysis

For generic data analysis tasks, our group makes liberal usage of existing software, such as components of the SciPy ecosystem. In addition to NumPy, which we use extensively and have incorporated into many of our own open-source packages, we also use other SciPy tools for, e.g., optimization and working with spatial data. Jupyter notebooks are central to our workflow, and we typically use the Matplotlib library[85] for plotting. For machine learning tasks, we typically develop methods using scikit-learn, Keras, and TensorFlow.

For analyses more specific to particle simulations, however, we found that many of the standard analysis methods we use—including correlation functions, order parameters, and more—had only been implemented in various prototype codes. Eventually, we collected our prototypes into `freud`[33], a Cython, C++, and Intel Threading Building Blocks (TBB)-accelerated tool that, like HOOMD-blue, provides a transparent Python API to a high-performance back-end. While many methods have been refactored and added to `freud`, the package maintains a consistent API across these methods. Tasks that do not fit in `freud` have been placed in other, more specialized tools.

6.5.3 Visualization

Our simulation visualization codes have undergone the most consistent process of repeated refactoring of prototypes and partial solutions. Our simulations typically generate long trajectories of complex systems that require powerful software and hardware for effective visualization. HOOMD-blue provides output in formats that can be interpreted by well-known simulation visualization packages such as Visual Molecular Dynamics (VMD) [177], PyMol, and Ovito[63], but these tools are generally incapable of visualizing anisotropic particles, which require interpretation of their orientations in addition to their positions. Therefore, we used injavis, a previously developed Java visualization and analysis tool

that represented particle data using XYZ-coordinates. Our group members extended this software to account for particle orientations, and we enabled HOOMD-blue to provide output data in this format. As a largely Graphical User Interface (GUI)-centric application, however, its analysis capabilities can be difficult to integrate into scripted workflows.

The goal of refactoring the functions in injavis into a more scriptable tool was a driving force for the development of `freud`. To leverage `freud`'s analysis capabilities for visualization, we developed a visualization toolkit called plato that outsources its analysis functions to `freud`. This division makes plato easier to adapt for new visualization tasks, resulting in a more modular, maintainable software infrastructure. In addition to plato, we developed fresnel to generate publication quality images using a GPU-accelerated path tracing engine.

6.5.4 Data and Workflow Management

A comprehensive simulation study typically involves conducting many simulations, often using HPC clusters to achieve meaningful system dimensions and time scales. Although HOOMD-blue scripts accommodate user-defined parameters, the user remains responsible for reliably associating these parameters with the simulation outputs and subsequent analyses. Many group members developed prototype solutions , but they suffered from numerous drawbacks with respect to scalability, flexibility, and interpretability.

To resolve these problems, we developed signac [86, 87], an open-source framework for constructing complex workflows on large, heterogeneous data spaces. Inspired by these prototypes, signac uses well-formed JavaScript Object Notation (JSON) parameter files to associated data files with their identifying parameters, providing a robust and full-featured database interface to data stored directly on the filesystem. The system is designed for the high performance filesystems inherent to HPC environments and supports the highly parallel file I/O operations required for, e.g., MPI-enabled HOOMD-blue simulations. Since all data and metadata are stored directly on the filesystem, they can also be easily transferred using tools like `rsync` or GLOBUS [178]. The signac framework includes signac-flow, a tool for managing and automating complex workflows operating on a signac data space. Workflows designed with signac-flow are immediately portable to HPC environments, making it possible to design and test workflows on local resources and then immediately submit them to a cluster scheduler.

6.5.5 Software Integration

The smooth functioning of our overall pipeline also depends on a number of smaller packages that perform more limited but equally important tasks. A good example of applying lazy refactoring is rowan [84], an open-source Python package for quaternion operations. A standard method for representing particle orientations in 3D, quaternions are used throughout our code base; however, individual packages have historically each had their own implementations. This fragmentation strongly suggested the need for standardization, particularly because many individuals also reimplemented these methods for their own, more *ad hoc* uses. Although quaternion packages already existed within the Python ecosystem, they suffered from numerous drawbacks with respect to performance and flexibility that made them unsuitable for incorporation into our code bases. rowan is the result of refactoring our own quaternion codes into a single package providing a unified API for working with quaternions at a uniform level of generality appropriate for our needs.

6.6 Training and Support

Our group has built an infrastructure around our software to provide users with training and information on how to take maximal advantage of our ecosystem. In addition to the documentation associated with each of our packages, we have created comprehensive integrated documentation that explains how to utilize our stack as part of a cohesive workflow, including a crash course to incrementally acquaint new users with this stack. We use Slack to maintain internal chat rooms for instant technical support, specialized discussions of science or software, and coordinating development.

We also aim to make our software useful to the broader community. We publish documentation through ReadTheDocs, distribute software via Anaconda and PyPI, and have published papers on some of our packages to further publicize them. We have also presented our software at the Scientific Computing with Python Conference (SciPy) [87], the annual American Physical Society (APS) March meeting, the triennial Foundations of Molecular Modeling and Simulation Conference (FOMMS), and the annual meeting of the American Institute of Chemical Engineers (AIChE), to name a few. In addition to these presentations, which are one avenue for feedback, we also use surveys and focus groups to gather user feedback. Biannual hackathons where all group members participate in improving our code help to improve the distribution of code ownership.

6.7 Conclusions

Advances in computational science are heavily dependent on the ongoing evolution of the scientific software ecosystem. Although quickly produced *ad hoc* solutions may seem sufficiently expedient in the moment, scientific progress can often be accelerated by a judicious use of existing software. Consequently, time spent properly designing and developing reusable software solutions may prove worthwhile if the reuse potential is identified sufficiently early in the development process. Properly estimating the future reuse potential of a piece of code is a daunting task, however, requiring a degree of foresight that is often impossible in scientific applications. Moreover, assessing the existing software options when embarking on a new project is far from trivial.

To address the need for a systematic approach to these problems, we have outlined a near optimal approach to determining when and how to develop reusable software that works well for us. The lazy refactoring approach is designed to balance sustainably improving the scientific software landscape with making immediate scientific progress. It does this by advocating for individual researchers to

1. evaluate existing software for its reuse potential prior to any code development,
2. adapt existing code bases for the problem at hand,
3. refactor existing code bases into proper packages whenever there are more than two use cases,
4. develop rapidly evolving prototype code strictly focused on solving the problem at hand in all other cases.

Lazy refactoring is particularly well suited to the academic research group. To illustrate its application in this context, we have shown how it has been applied over time by our research group to develop a suite of independent but highly interoperable and powerful tools targeted at a specific class of scientific problems. Through this, we provided a concrete example of how lazy refactoring may be used to significantly accelerate net scientific output in any subfield of computational science.