RISM THEORIES FOR POLYMERS AND MULTI-SITE MOLECULES:
APPLICATIONS TO POLYMER BLENDS NEAR SURFACES
AND HYBRID THEORY/SIMULATIONS

by

Caleb A. Tormey
© Copyright by Caleb A. Tormey, 2016
All Rights Reserved
A thesis submitted to the Faculty and the Board of Trustees of the Colorado School of Mines in partial fulfillment of the requirements for the degree of PhD (Applied Chemistry).

Golden, Colorado
Date ______________________

Signed: ______________________
    Caleb A. Tormey

Signed: ______________________
    Dr. David T.W. Wu
    Thesis Advisor

Golden, Colorado
Date ______________________

Signed: ______________________
    Dr. David T.W. Wu
    Professor and Head
    Department of Chemistry
Polymers are one of the most important materials in modern industry. An area of increased interest is understanding polymer blend behavior at surfaces. Understanding of these processes can help to improve material properties such as adhesion, viscosity and surface chemical composition. Current research into blends of polymers with the same monomer makeup but different architectures have been an area of increasing research. Previous work has examined blends of linear and branched systems. However, blends of polymers with cyclic chains are of increasing interest as new synthetic techniques have allowed cyclic blends to be more easily studied. Cyclic polymers have interesting properties because of their lack of end groups and more compact size when compared with linear polymers of the same molecular weight (MW). Wall Polymer Reference Interaction Site Model (Wall-PRISM) studies were conducted to compare with Neutron Reflectivity (NR) studies of cyclic/linear blends near surfaces. To accomplish this a new cyclic model was developed based on the linear Non-overlapping Freely Jointed chain model. The Wall-PRISM calculations were also performed over a range of densities and with different surface stiffnesses to study effects on chain packing at the surface.

Even with increased computational power, large polymer systems can often still prove costly to simulate. Self-consistent PRISM (SC-PRISM) is a state-of-the-art hybrid approach that simulates only a single chain with effective interactions calculated by theory to mimic the influence of the surrounding chains. The recent introduction of Two-Chain SC-PRISM, simulating two chains with effective interactions, showed improvement of results over SC-PRSIM for polyethylene melts when compared with MD studies. However, numerical problems with using polymer chain simulation together with PRISM theory have kept Two-Chain SC-PRISM from being applied to a wide range of polymer systems. I have developed new methods for incorporating the simulation data to work with PRISM theory. This new
method was tested on several small molecule systems with both hard sphere and attractive Lennard-Jones potentials and showed overall improvement over the single molecule version of RISM/PRISM. This new method was then extended to allow for attractive potentials, overcoming another limitation of conventional RISM/PRISM. With the new method tested and verified on smaller molecular systems, a foundation is laid for applying the Two-Chain SC-PRISM method to more complex polymer systems such as polyolefin blends and those with attractive interactions such as polymer electrolyte systems.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABSTRACT</td>
<td>iii</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>ix</td>
</tr>
<tr>
<td>LIST OF TABLES</td>
<td>xvi</td>
</tr>
<tr>
<td>LIST OF SYMBOLS</td>
<td>xvii</td>
</tr>
<tr>
<td>LIST OF ABBREVIATIONS</td>
<td>xviii</td>
</tr>
<tr>
<td>ACKNOWLEDGMENTS</td>
<td>xix</td>
</tr>
<tr>
<td>DEDICATION</td>
<td>xx</td>
</tr>
<tr>
<td>CHAPTER 1 INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>CHAPTER 2 BACKGROUND</td>
<td>6</td>
</tr>
<tr>
<td>2.1 Surface segregation of linear/nonlinear polymer blends</td>
<td>6</td>
</tr>
<tr>
<td>2.2 Experimental methods for measuring surface segregation in polymer blends</td>
<td>6</td>
</tr>
<tr>
<td>2.3 Factors that influence polymer blend surface segregation</td>
<td>9</td>
</tr>
<tr>
<td>2.4 Theoretical and computational study of architecturally driven surface segregation in polymer blends</td>
<td>11</td>
</tr>
<tr>
<td>2.4.1 Self-Consistent Field Theory</td>
<td>11</td>
</tr>
<tr>
<td>2.4.2 Wall-PRISM and Monte Carlo methods</td>
<td>15</td>
</tr>
<tr>
<td>2.5 Reference Interaction Site Model (RISM)</td>
<td>16</td>
</tr>
<tr>
<td>2.6 Polymer Reference Interaction Site Model (PRISM)</td>
<td>17</td>
</tr>
<tr>
<td>2.7 Self-Consistent PRISM (SC-PRISM or One-Chain SC-PRISM)</td>
<td>18</td>
</tr>
<tr>
<td>2.8 Two-Chain Self-Consistent PRISM (Two-Chain SC-PRISM)</td>
<td>19</td>
</tr>
</tbody>
</table>
2.8.1 Summary .................................................. 20

CHAPTER 3 RISM/PRISM THEORIES ................................. 23
3.1 Integral equation theory ...................................... 23
  3.1.1 Reference Interaction Site Model (RISM) ..................... 25
  3.1.2 Polymer Reference Interaction Site Model (PRISM) .............. 26
  3.1.3 Closure equations ........................................ 28
  3.1.4 Picard iteration solution for PRISM equations ................. 29
3.2 Wall-PRISM theory ........................................... 31
3.3 Self-Consistent PRISM ......................................... 35
  3.3.1 Solvation potential ....................................... 36
  3.3.2 Self-Consistent PRISM method ............................ 37
3.4 Two-Chain SC-PRISM .......................................... 39
  3.4.1 Two-Chain theory ....................................... 39
  3.4.2 General Two-Chain numerical scheme ....................... 43
3.5 Two-Chain Monte Carlo simulation ........................... 45
  3.5.1 Direct sampling ........................................ 46
  3.5.2 Window sampling ........................................ 48

CHAPTER 4 SURFACE SEGREGATION OF LINEAR/CYCLIC BLENDS .... 53
4.1 Non-overlapping freely jointed chain polymer model ............ 54
  4.1.1 Linear chain - new derivation of NFJC intramolecular correlation functions ........................................ 54
  4.1.2 Cyclic chain - derivation of new model ...................... 60
4.2 Wall-PRISM results ........................................... 63
4.2.1 Resolution smoothing to compare with experimental density profiles . . . 69
4.2.2 Soft wall derivation ................................................. 71
4.3 Summary .......................................................... 74

CHAPTER 5 MULTI-SITE TWO-CHAIN .................................. 78
5.1 Two-Chain SC-PRISM numerical hurdles .................................. 78
5.2 The “tweaking” method to project out unphysical noise from simulation
correlation functions .................................................. 87
5.2.1 Alternate self-consistent scheme .................................. 95
5.3 Overlapping heterodimers ................................................ 99
5.3.1 Model I results ..................................................... 100
5.3.2 Model II results ..................................................... 102
5.3.3 Model III results ..................................................... 102
5.3.4 Density range for Model I ......................................... 105
5.4 Lennard-Jones heterodimers ............................................. 106
5.4.1 Model II LJ ......................................................... 108
5.4.2 Model III LJ ......................................................... 108
5.5 Trimer systems ........................................................ 114
5.5.1 Sulfur dioxide ....................................................... 115
5.5.2 Linear hard sphere trimers ....................................... 120
5.5.3 Testing density range ............................................. 121
5.6 Summary ............................................................ 126

CHAPTER 6 TWO-MOLECULE RISM WATER ............................. 128
6.1 Coulomb potential and appropriate divergence ......................... 128
## LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure 2.1</th>
<th>Polymers at substrate and air/vacuum surfaces, with monomers depicted as spheres (and bonds between monomers not shown). The enrichment of one type of polymer over another will show up in the comparisons of the density profiles.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 2.2</td>
<td>Neutron Reflectivity experimental geometry.</td>
</tr>
<tr>
<td>Figure 2.3</td>
<td>Reflectivity curves for a blend of 20 volume % pom-pom branched hPS with 80 volume % linear dPS. The inset shows the fitted results for the volume fraction profile of the 6-pom, $\phi_{pom}$, at a given depth $z$.</td>
</tr>
<tr>
<td>Figure 2.4</td>
<td>Model illustrating pompom polymer with variable linear connecting sections used in SCFT study by Hu as well as the star polymer model.</td>
</tr>
<tr>
<td>Figure 2.5</td>
<td>Comparison of SCFT and NR results for the integrated excess of the various 6-pom polymers at the surface of a blend with linear polymers. All chains are 36k in molecular weight. The bulk volume fractions of linear and pompom polymer are $\phi_{linear} = 0.8$ and $\phi_{pom} = 0.2$ respectively.</td>
</tr>
<tr>
<td>Figure 2.6</td>
<td>The radial distribution functions, $g(r)$, between monomers for $N = 24$ and $N = 66$ polymer melts, where $N$ is the number of monomer units in the polymer chain. The monomers interact with the repulsive Lennard-Jones potential with parameters for the non-bonded interactions of $\epsilon/k = 46.5$ K and $\sigma = 3.93$ Å. The bond length is 1.54 Å for both systems. For $N = 24$ the temperature was 405 K with a density of $\rho = 0.03104 \text{ Å}^{-3}$ and for $N = 66$ the temperature was 448 K and a density of $\rho = 0.3104 \text{ Å}^{-3}$. The intramolecular chain parameters are reported in The points are from MD simulations, the dashed curves are the One-Chain SC-PRISM results and the solid lines are from the single site Two-Chain SC-PRISM theory.</td>
</tr>
<tr>
<td>Figure 3.1</td>
<td>Illustration of the definition of the total correlation between particle $p_1$ and $p_2$ in terms of chains of the direct correlation function.</td>
</tr>
<tr>
<td>Figure 3.2</td>
<td>The Picard iteration scheme used here to solve the RISM/PRISM Equations.</td>
</tr>
<tr>
<td>Figure 3.3</td>
<td>Annulus site configuration as precursor for Wall-PRISM solution.</td>
</tr>
</tbody>
</table>
Figure 3.4 Final Wall-PRISM system configuration where $R_1, R_2$ and $R_3$ from Figure 3.3 are taken to the infinite radius limit. .................................................. 36

Figure 3.5 Single Chain SC-PRISM theory solution scheme. ................................. 39

Figure 3.6 Two-Chain SC-PRISM theory solution scheme with convergence based on $C(r)$. ................................................................. 44

Figure 3.7 Direct sampling method for two chains. .............................................. 48

Figure 3.8 Window sampling method of two chains. ............................................ 49

Figure 3.9 Window overlap configuration and designation of Left and Right window halves. ................................................................. 51

Figure 3.10 Radial distribution function reconstructed from window probability distribution functions for a polyethylene chain modeled as a chain of 24 sites ($M = 24$) interacting with a repulsive LJ potential; $r_0 = r_{\text{width}} = 3\, \text{Å}$. Top: figure shows the $g(r)$ results from the joined window samples (solid line) and a direct sampling method (circles) Bottom: are the probability distributions from the even numbered windows $i = 2, 4, 6...14$ and go from left to right. ...................... 52

Figure 4.1 Example of an $N = 10$ linear tangential monomer chain where the numbers indicate the site indices for $\alpha$ and $\gamma$. Examples are shown for the definition $\tau \equiv |\alpha - \gamma|$ with respect to the site indices. Note that the actual conformation of the chain does not have to be a straight line, but is shown this way to simplify visualization of $\tau$. ................................. 57

Figure 4.2 Kratky plot of the Ideal Freely Jointed Chain and the Non-Overlapping Freely Jointed Chain for a $N = 20$ linear chain. ................................. 59

Figure 4.3 Example of an $N = 10$ cyclic tangent monomer chain where the numbers indicate the site indices $\alpha$ and $\gamma$. Examples are shown for the definition of $\tau$ with respect to the site identities. This results show the two different paths of length $\tau = 1$ and $\tau = 9$ for the case of $\alpha = 1$ and $\gamma = 10$. ................................................................. 61

Figure 4.4 Kratky plot of Ideal Freely Jointed Chain and the Non-Overlapping Freely Jointed Chain for a $N = 20$ cyclic chain. ................................. 62

Figure 4.5 Kratky plots comparing the linear and cyclic chain intramolecular structure factors for A) Ideal FJC and B) Non-overlapping FJC ................................. 63
Figure 4.6  Bulk pair distribution function from PRISM for a binary blend of 80/20 Linear/Cyclic Polystyrene with $\eta = 0.45$.  

Figure 4.7  Wall-PRISM monomer-wall pair distribution function for a binary blend of 80/20 Linear/Cyclic Polystyrene. The origin in this plot indicates the start of the wall, with $r$ indicating the distance from the wall into the bulk.

Figure 4.8  Comparison of the same 80/20 Linear/Cyclic Polystyrene at 4 different densities. The densities are given at the top of each plot.

Figure 4.9  The difference in normalized density, with the regions in dark gray where the linear chain is in excess and in light gray the cyclic.

Figure 4.10 Net integrated excess of linear chains for varying volume fractions ($\eta$).

Figure 4.11 Wall-PRISM density profiles for linear and cyclic blends at $\eta = 0.45$, A) without, and B) with smoothing using the Hann window method.

Figure 4.12 Comparison of linear chain density profiles with depth, $z$, of a linear/cyclic blend. Black: NR of the free surface of a blend, Red dotted: smoothed Wall-PRISM theory of a blend next to a hard wall. Note in the NR profiles, the density falls to the left, as expected for a free surface.

Figure 4.13 Soft wall potential compared with hard wall infinite potential. The start of the walls is at $\pm \frac{H}{2}$ where $H = 30$. See Figure 3.4.

Figure 4.14 Wall-PRISM of $C_{Ap}$ and $C_{Bp}$ where $A$ and $B$ are the cyclic and linear chains respectively for the initial Picard iteration step to show change in wall-PRISM closure for a soft wall. The black line indicates a hard wall is enforced exactly, the gray dotted line shows where the soft wall approximation is made to act like a hard wall and the red dotted shows a true soft wall.

Figure 4.15 Panels A) and B) are the cyclic and linear Wall-Polymer $g(r)$ results respectively. These plots show a range of stiffnesses $S$.

Figure 4.16 Plot of integrated excess of linear polymers at surface vs $S$ for reducing stiffness of wall.
Figure 5.1  The single-site type homodimer model, and the corresponding equivalent multi-site heterodimer model used in the scalar and matrix Two-Molecule SC-PRISM theories respectively. These correspond to the same physical molecule, when B differs only as a label but in no physical property. ................................................................. 79

Figure 5.2  Plots showing the numerical divergence appearing in $\hat{C}_{\text{New}}(k)$ for the first iteration in solving the Two-Chain SC-PRISM theory when using the heterodimer model matrix form compared with the homodimer scalar form. These results are for an initial guess of $C(r) = 0$, and thus zero solvation potential. A) is zoomed in to show that the scalar and matrix results agree for larger $k$. B) shows the scale of the divergences at low $k$. ................................................................. 80

Figure 5.3  Plot of the determinant of the intramolecular structure factor for the heterodimer. ............................. 81

Figure 5.4  Method for modifying simulation data to satisfy non-divergence constraints. Panel A) shows an example of the points selected for tail fitting and B) shows application of the tail fitting. ................................. 86

Figure 5.5  Poor tail fitting as a result of trying to satisfy $k^{-4}$ and $k^{-2}$ conditions using only the fitted tail parameters. The system is a hard sphere heterodimer outlined in section 5.1, with system parameters of $\sigma_A = \sigma_B = 3.93 \text{Å}$, and $\rho_{\text{Dimer}} = 0.006293 \text{Å}^{-3}$. ................................................................. 87

Figure 5.6  Proposed new method where each value of $h(r_i)$ in a range receives an additive “tweak” $q_i$. The tweaks can be positive or negative. The blue circles represent the raw simulation data, $h(r)_{\text{raw}}$, and the black arrows represent the addition of tweaks. ................................................................. 90

Figure 5.7  $\hat{C}_{\text{New}}(k)$ from multi-site Two-Chain theory using the “tweaking” method for the first iteration using identical conditions as those in Figure 5.2. A) shows elimination of the low-$k$ numerical divergence, and good agreement with scalar Two-Chain theory B) is zoomed in to show the match. ................................................................. 95

Figure 5.8  “Tweaking” raw simulation $h(r)$ to obtain “fixed” $h(r)$ that do not show low-$k$ divergence. A), B) and C) show the change in $h(r)$ with the addition of the tweaking vector for $AA$, $AB$ and $BB$ respectively. D) shows the change in $k$-space as it is applied to $\hat{C}_{\text{New}}(k)$. ................................................................. 96

Figure 5.9  Alternate iterative Two-Chain solution scheme using method from and including tweaking vector step. AFD is the Average Fractional Difference. ................................................................. 97
Figure 5.10 Comparison of symmetric heterodimer intermolecular correlation function, $h(r)$, using the multi-site two-molecule theory with the equivalent homodimer solved using the scalar Two-Chain theory. The x-axis corresponds to $r$ divided by the hard sphere diameter of the sites ($\sigma = 3.93 \text{ Å}$).

Figure 5.11 Comparison of multi-site and scalar Two-Molecule theories applied to attractive homodimer system: molecular nitrogen. The system uses an HNC solvation potential, with a molecule density of $\rho = 0.01851 \text{ Å}^{-3}$, $T = 65 K$, $\epsilon = 0.0739 \text{ kcal/mol}$, $\sigma = 3.315 \text{ Å}$ and a bond length of $L = 1.1 \text{ Å}$. The full Lennard Jones potential was used and is defined in Equation 5.36.

Figure 5.12 Comparison of One-Chain, Two-Chain and Monte Carlo results for Model I

Figure 5.13 Comparison of One-Chain, Two-Chain and Monte Carlo Results for Model II. The three panels A), B) and C) give the RDF for the $AA$, $AB$ and $BB$ site pairs respectively.

Figure 5.14 Comparison of One-Chain, Two-Chain and Monte Carlo Results for Model III. The three panels A), B) and C) give the RDF for the $AA$, $AB$ and $BB$ site pairs respectively.

Figure 5.15 Radial distribution functions for Model I for varying densities. A) $\rho = 0.00092 \text{ Å}^{-3}$, B) $\rho = 0.0018 \text{ Å}^{-3}$, C) $\rho = 0.0046 \text{ Å}^{-3}$ and D) $\rho = 0.0064 \text{ Å}^{-3}$.

Figure 5.16 Comparison of One-Chain, Two-Chain and Molecular Dynamics Results for Model II repulsive LJ. The three panels A), B) and C) give the RDF for the $AA$, $AB$ and $BB$ site pairs respectively.

Figure 5.17 Comparison of One-Chain, Two-Chain and Molecular Dynamics results for Model II full LJ. The three panels A), B) and C) give the RDF for the $AA$, $AB$ and $BB$ site pairs respectively.

Figure 5.18 Comparison of One-Chain, Two-Chain and Molecular Dynamics Results for Model III Repulsive LJ. The three panels A), B) and C) give the RDF for the $AA$, $AB$ and $BB$ site pairs respectively. Possible reasons for the offset of the full LJ results are mentioned previously about Model II full LJ.

Figure 5.19 Comparison of One-Chain, Two-Chain and Molecular Dynamics Results for Model III Full LJ. The three panels A), B) and C) give the RDF for the $AA$, $AB$ and $BB$ site pairs respectively.
Figure 5.20 Comparison of One-Chain, Two-Chain and Molecular Dynamics
Results for Model III, averaging $g(r)$ from the full and repulsive LJ cases. The three panels A), B) and C) give the RDF for the $AA$, $AB$ and $BB$ site pairs respectively. 

Figure 5.21 Bent and linear trimer models having two site-types. The left panel depicts a linear trimer and the right panel depicts a bent trimer.

Figure 5.22 Comparison of One-Chain, Two-Chain and Molecular Dynamics results for sulfur dioxide for $T = 350$ K and $\rho = 0.0109 \text{ Å}^{-3}$. A) SS, B) SO/OS, and C) OO site-site elements in $g(r)$. 

Figure 5.23 Comparison of One-Chain, Two-Chain and Molecular Dynamics Results for SO$_2$ at $T = 250$ K $\rho = 0.0136 \text{ Å}^{-3}$. A) SS, B) SO/OS, and C) OO site-site elements in $g(r)$. 

Figure 5.24 Linear Trimer: Comparison of One-Chain, Two-Chain and Molecular Dynamics results for $\eta = 0.0$: A) End to End, B) End to Middle, C) Middle to Middle site-type correlations. The Two-Molecule results are found using the HNC solvation potential.

Figure 5.25 Linear Trimer: Comparison of One-Chain, Two-Chain and Molecular Dynamics Results for $\eta = 0.2$: A) End to End, B) End to Middle, C) Middle to Middle site-type correlations.

Figure 5.26 Linear Trimer: Comparison of One-Chain, Two-Chain and Molecular Dynamics Results for $\eta = 0.35$: A) End to End, B) End to Middle, C) Middle to Middle site-type correlations.

Figure 5.27 Linear Trimer: Comparison of One-Chain, Two-Chain and Molecular Dynamics Results for $\eta = 0.445$ A) End to End, B) End to Middle, C) End to End zoomed in, D) End-Middle zoomed in.

Figure 5.28 Linear Trimer: Comparison of One-Chain, Two-Chain and Molecular Dynamics Results for $\eta = 0.45$: Middle to Middle.

Figure 6.1 Diagram of “tweaking” vector and tail fit for Coulombic potentials for use with Two-Molecule RISM.

Figure 6.2 Comparison of MD, One-Molecule and Two-Molecule RISM for the SPC model of water. A) OO, B) OH/HO, C) HH site-site distribution functions.
Figure 6.3 $\hat{C}(k)$ results showing the controlled divergence using the dielectric and $h(r)$ sum rules. Both A) and B) are of the same plot with B) being zoomed in. ................................................................. 138

Figure B.1 Diagram of Parallel Direct Sampling Scheme ................................. 163
LIST OF TABLES

Table 3.1 Definition of solvation potentials $(W(r))$ ........................................... 37

Table 5.1 Parameters for the 3 overlapping dimer systems to be tested with Two-Molecule theory .................................................. 100

Table 5.2 Table of LJ parameters used for Model II and Model III LJ dimers ... 107

Table 5.3 System and Lennard Jones parameters for SO$_2$ ............................. 116

Table 5.4 Linear trimer system parameters ......................................................... 121

Table 6.1 Parameters for SPC and SPC/E Models ............................................. 136
LIST OF SYMBOLS

Total Correlation Function . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . \( h(r) \)

Direct Correlation Function . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . \( C(r) \)

Radial Distribution Function . . . . . . . . . . . . . . . . . . . . . . . . . . . . . \( g(r) \)

Intramolecular Radial Distribution Function . . . . . . . . . . . . . . . . . . . . \( \omega(r) \)
# LIST OF ABBREVIATIONS

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RISM</td>
<td>Reference Interaction Site Model</td>
</tr>
<tr>
<td>PRISM</td>
<td>Polymer Reference Interaction Site Model</td>
</tr>
<tr>
<td>TCF</td>
<td>Total Correlation Function</td>
</tr>
<tr>
<td>DCF</td>
<td>Direct Correlation Function</td>
</tr>
<tr>
<td>OZ</td>
<td>Ornstein-Zernike</td>
</tr>
<tr>
<td>FFT</td>
<td>Fast Fourier Transform</td>
</tr>
</tbody>
</table>
ACKNOWLEDGMENTS

First and foremost I must thank my adviser Dr. David Wu. I am forever grateful for him taking a chance on taking me as a student and the kindness and patience he has shown me over the years. I would be remiss if I didn’t thank Dr. James Donley for his help in learning the art of computational science and for always being there to answer my questions. Also I must thank Dr. Mark Eberhart for advising me to always consult Maxwell’s demon.

I thank my research group mates Dr. Ben Zeidman, Dr. Renfeng Hu, Michael Servis, Jason Goins, Joe Maestas and Shuo Liu. You all have given me great feedback and ideas and you were instrumental in my working through all the challenges I faced in finishing my PhD. Thank you to my friends Jeff Ward for constantly answering my Linux and programming questions and Ryan Abo as a great sounding board for ideas.

I also need to thank my parents and family for their love and support and for nurturing my love of science. Thank you to my children for understanding that sometimes Dad had to work instead of play with you. Finally I would like to thank with deepest gratitude my wife for the love she has given me over these past years. She has shown such great patience and support during our marriage and especially during these past years as I completed my degree.
Dedicated to my ever curious children. Never stop asking why.
CHAPTER 1
INTRODUCTION

Fluid systems are an integral part of modern technology. Everything from seemingly simple systems such as water to complex systems such as liquid crystal displays used in modern computing can be understood using principals of statistical mechanics and data from computer simulation. Gaining insight into these systems from a theoretical and molecular perspective can allow ever better technologies to be created. One of the most important aspects for understanding liquid systems is determining how its constituents behave and arrange themselves. For fluid systems the densities and molecular interactions are high enough that they can’t be ignored as they can be in the ideal gas limit. Also, unlike solid, crystalline systems there is no long range translational order. This gives a system where local order must be accounted for but at long ranges the system is uncorrelated.

Even with these difficulties theoretical approaches have proved very useful for the study of polymer systems. One area of polymer research that continues to grow is research into polymer blends. While many blends of unlike polymers are characterized by a tendency toward demixing due to differences in monomer chemistry, blends of polymers with the same monomer constituents but with differences in monomer connectivity, i.e., in polymer architecture, exhibit interesting physical behaviors. In particular, architectural polymer blends can undergo segregation of the polymer components at in the presence of a surface or interface. Understanding surface composition can be crucially important in material design. A linear response theory by Wu and Fredrickson [1] predicted that surface enrichment of one component of an architectural blend could be interpreted in terms of surface potentials for ends and branching points. Further theoretical work by [2] using self consistent field theory (SCFT) was able to reproduce trends seen in Neutron Reflectivity experiments for differing pom-pom branched/linear blends performed by the Foster research group at the University
Continued work by both Foster and Wu research groups looked into surface segregation behavior in cyclic linear blends. The linear response theory of Wu and Fredrickson as well as the SCFT calculation by Hu concluded that entropic considerations would lead to cyclic preference at a surface for high molecular weight chains; however, experiments showed a crossover from this surface preference of cyclic chains at high MW to a surface preference for linear chains at small molecular weights. For small chains, packing effects were likely an important feature that was lost in the course grained SCFT. In the work in this thesis, I extended the non-overlapping FJC model of Curro and Schweizer [3] from linear topologies to cyclic topologies, and then applied this model to the problem of surface enrichment using the Wall Polymer Reference Interaction Site Model (Wall-PRISM) theory.

Another factor in studying fluid systems that can be challenging is that of molecular scale. Liquid systems can be composed of small molecules as well as macromolecules such as polymers and proteins and also be mixtures of the above. In the mid 1900’s computer simulations were difficult to perform on large or complex systems. As such theoretical methods were often preferable as a means for understanding liquid systems. With the increase in computer performance and efficiency more and larger systems are able to be simulated. While theory is still integral to physical understanding, use of simulations continues to grow. However, the rapid increase of computational cost is an issue when moving beyond small molecules to higher molecular weight systems such as polymers. Simulations of polymer blends that can capture miscibility effects can take months of computation time on a super-computer [4]. The difficulties of explicit simulation of systems is compounded by natural slow dynamics or rare events, such as ion transport in a polymer electrolyte. Simulating the system may not only be computationally taxing, but also be unable to capture important events such as inter-chain hopping of the ions. Theoretical approaches can help address these limitations.

Combining theoretical and simulation techniques can give the best of both by not only reducing the total computation time but also getting important physically detailed informa-
tion from the limited simulation. Self-consistent PRISM (SC-PRISM) [5] is a leading hybrid theory/simulation technique that simulates a single polymer chain, accounting for the other chains through statistical mechanical theory. It has been used to model a range of systems from blends to polymer coated nanoparticles [4, 6]. However, the accuracy of SC-PRISM can be limited by the loss of explicit molecular interactions when only simulating a single chain. Studies have shown discrepancies in the pair correlation functions calculated from SC-PRISM and from MD studies of polymer blends, especially at small distances.

To better capture these short-ranged interactions, Li and coworkers suggested that simulations could be carried out on two polymer chains explicitly, with the remaining chains handled by theory. Moreover, by simulating two chains explicitly, it was suggested that the theory may do a better job of accounting for attractive interactions, which was not handled well by PRISM or SC-PRISM. Such a hybrid “Two-Chain PRISM” theory/simulation calculation was carried out for polyethylene and shown to improve results without increasing the computational cost significantly[7]. However, there remained a problem in applying this hybrid Two-Chain SC-PRISM system to anything but single site polyethylene. To make the Two-Chain PRISM theory a generally applicable methodology requires the ability to extend the method to multiple components and site-types. To do this involves moving from scalar to matrix quantities, that results in a matrix inversion that amplifies numerical noise from the simulation and ultimately lead to numerical divergences that prevent converging on a solution. Solving this problem is a challenging but necessary task if any complex and more interesting systems, including those with attractive interactions, are to be studied.

The research of this thesis is mainly focused on extending current molecular level statistical liquid-state theory, addressing the two problems discussed above. The first portion of this thesis develops theory for the architecture controlled surface enrichment in small molecular weight linear/cyclic blends using Wall-PRISM. The development of a method to use the Two-Chain PRISM method with multiple interaction types constitutes a majority of my work and forms the larger second part of this thesis. While Two-Chain PRISM theory is
intended to be directly applicable for polymer systems, the majority of my work is on the
general development of Two-Chain PRISM theory methodology, and was primarily tested
on smaller molecule systems for proof of concept.

My thesis is organized in the following manner. Chapter 2 summarizes the state of the
art for the two background areas of my thesis, with the second area being the majority of the
work. The first area is the experimental and theoretical methods for studying architecturally
asymmetric binary polymer blends. The second area is liquid-state-theory and the theoretical
and computational approaches used for both small molecules and macromolecules. Special
attention will be focused on work dealing with SC-PRISM as well as the Two-Chain SC-
PRISM method. Chapter 3 will cover all the Reference Interaction Site Model (RISM)
theories, and their application to polymer systems, the Polymer Reference Interaction Site
Model (PRISM) theories used in this thesis. This includes introduction to the RISM theory
for small molecules and extension to ideal polymer melts with PRISM. Then Wall-PRISM is
reviewed for work of polymer blends near surfaces. Finally, the two hybrid theory/simulation
PRISM theories, namely SC-PRISM and Two-Chain SC-PRISM, are covered, including
discussion of closure relations and solvation potential approximations. This section will
be referenced heavily throughout the thesis. Chapter 4 will introduce the Wall-PRISM
method for polymers near a surface. Here results for calculations on small molecular weight
polymer blends will be presented. Also, included here will be a derivation for a ring polymer
intramolecular correlation function based on a non-overlapping freely-jointed chain model.
The Wall-PRISM results will be compared with those from the coarse-grained Self-Consistent
Field Theory (SCFT) and neutron reflectivity (NR) experiments for linear/ring polymer
blends at a surface. Chapter 5 will introduce my new method for controlling numerical issues
arising from applying Two-Chain PRISM theory to simulation data for multi-site Two-Chain
SC-PRISM. For testing purposes the RISM form will be used and small molecules such as
molecular dimers and trimers will be tested and compared with simulation methods as well
as conventional (pure theory) RISM results. Chapter 6 will present further development of
my multi-site method and application to charged systems with Coulomb type forces. This will be demonstrated using a SPC water system. Again this will be compared with MD/MC simulations and single chain SC-RISM results.
CHAPTER 2
BACKGROUND

This chapter will provide context and current work for the two main areas of my thesis. First, background on surface segregation will be presented in conjunction with the Wall-PRISM theory for polymers at a surface. Second, background on RISM/PRISM theories in general as well as the hybrid simulation/theory methods, SC-PRISM and Two-Chain SC-PRISM.

2.1 Surface segregation of linear/nonlinear polymer blends

The next few sections will briefly review both experimental and theoretical studies of surface segregation for linear/nonlinear polymer blends. I will review both experimental methods for surface excess measurements as well as explore theoretical explanations of segregation behavior. The main focus is on blends of polymers composed of the same monomer but differing polymer architecture. In dealing with blends at surfaces it is helpful to visualize how the density for each polymer type can vary compared to each other and compared to the bulk. The type of system of interest is illustrated in Figure 2.1 below.

2.2 Experimental methods for measuring surface segregation in polymer blends

While the majority of the thesis will cover work dealing with theoretical and simulation methods for studying polymers, experimental methods are fundamental in guiding the theory. For studying surface segregation there are multiple techniques that can be used. These include neutron reflectivity (NR), nuclear reaction analysis, surface enhanced Raman scattering (SERS), secondary ion mass spectrometry (SIMS), forward-recoil spectrometry and X-ray photoelectron spectroscopy (XPS). The Wall-PRISM theory results in chapter 4 are compared with NR studies so I present a brief review here of neutron reflectivity.
Figure 2.1: Polymers at substrate and air/vacuum surfaces, with monomers depicted as spheres (and bonds between monomers not shown). The enrichment of one type of polymer over another will show up in the comparisons of the density profiles.

Figure 2.2: Neutron Reflectivity experimental geometry.
In an NR experiment an incident beam of neutrons strikes the flat sample surface, but at a very acute angle (see Figure 2.2). The intensity of the reflected neutron beam with an angle equal to the corresponding incident beam is measured by the detector and the angle of both the incident and reflected (measured) beams will be varied over a range. As the angle is varied, the reflected intensity will also vary as the scattering length density changes with the depth within the sample. From this characteristic change in scattering intensity a density profile can be determined for a sample. To determine this profile, the reflectivity vs the momentum of the transfer vector $q$ of the neutron beam is recorded. The reflectivity is defined as $R = I_R/I_o$, where $I_o$ and $I_R$ are the intensities of the incident and reflected beams respectively. The $z$ component (normal to the sample surface) of the momentum transfer vector, $q$, will depend only on the neutron wavelength, $\lambda$, and the angle, $\theta$, of the incident beam (see Equation 2.1 below)

$$q_z = \frac{4\pi}{\lambda} \sin(\theta) \quad (2.1)$$

The scattering of the incident beam within the sample will depend on the depth dependent scattering length density (SLD or $\rho_{SL}$). The SLD is given by the equation

$$\rho_{SL}(z) = \sum_i n b_i V \quad (2.2)$$

where $b_i$ is the scattering length of atom $i$ and $V$ is a reference volume containing $n$ atoms of the material, for instance containing a single monomer unit of the polymer. NR can typically be used to determine $\rho_{SL}$ to a resolution of 10 Å and to a depth of 2-3 thousand Å. Data is collected by probing the surface to generate a plot similar to Figure 2.3. These data are then fit by non-linear regression to a model for the SLD depth profile. For determination of excess density in polymer blends the two polymers need to be distinguishable by NR. For blends composed of the same monomer, isotopic labeling can be used, since deuterium and hydrogen have very different neutron scattering lengths. One polymer type will be made with deuterated monomers while the other will have hydrogenated monomers. The blend is typically made in a common solvent then spin coated and allowed to anneal at
high temperature. Figure 2.3 shows an example with hydrogenated polystyrene (hPS) and deuterated polystyrene (dPS).

![Reflectivity curves for a blend of 20 volume % pom-pom branched hPS with 80 volume % linear dPS. The inset shows the fitted results for the volume fraction profile of the 6-pom, $\phi_{pom}$, at a given depth $z$ [8].](image)

**Figure 2.3: Reflectivity curves for a blend of 20 volume % pom-pom branched hPS with 80 volume % linear dPS. The inset shows the fitted results for the volume fraction profile of the 6-pom, $\phi_{pom}$, at a given depth $z$ [8].**

### 2.3 Factors that influence polymer blend surface segregation

There are multiple factors that can influence surface segregation in polymer blends. As mentioned earlier one of the simplest systems to look at is one based on size. It has been shown by both experiment and theory that surface segregation in a polymer blend can result solely from differences in chain length. Hong et al. [9] found that for linear blends of deuterated and hydrogenated polystyrene that the size of the chain had a clear effect and that the smaller chains were typically preferred at the surface, especially if the sizes were drastically different. Computer simulations also have shown that at least part of the segregation behavior can be attributed to polymer coil size [10]. For symmetric polymer blends Hariharan et al. [11] found that surface segregation was driven by the incompatibility of the blends. For blends of disparately sized chains the shorter molecules would show preference at the surface. These segregation effects were more dramatic with greater differences in sizes.
of chains [11, 12]. However, it was also found that when the long chains were made energetically preferred at the surface the entropic preference for short chains can be balanced to favor neither chain size at the surface.

It should be noted that inclusion of deuterium, commonly used in NR experiments, while helpful in providing contrast, has been shown itself to cause a weak surface segregation effect. There is a slight difference in the carbon deuterium bond length versus the carbon protium bond length, as well as in their polarizabilities. This can lead to molar volumes and surface free energy differences between the hydrogenated and deuterated polymer blend melts. These differences can result in measurable changes in the free energy of mixing and a slight surface preference for the deuterated chains [12–14]. To determine the contribution to surface segregation from isotopic differences alone, blends of deuterated and hydrogenated polymers with the same architecture (e.g. both linear chains) are studied (Lee2014).

The thickness of the film can also determine the degree to which segregation will occur. Budkowski et al [15] found that for thick polymer films segregation was minimal and more similar to the bulk, however, for very thin films the effect was more pronounced in blends of dPS and hPS. However, Hong [9] found that among very thin films the excess of dPS at the surface increased linearly with the thickness of the film. Walton [16] found that thicker films of P(MMA-r-MnG) and PMMA blends had a more pronounced surface preference for P(MMA-r-MnG) than for thinner films of the same composition.

More recently it has been shown that polymer architecture can play an important role in surface segregation. Architecture is an appealing strategy for driving surface segregation, as such a strategy can be universally applied to polymers of arbitrary monomer chemistry. Branching is one of the major ways of changing polymer architecture, and is exploited to vary polymer properties such as crystallinity, toughness, and rheology. Side chains on polymers can influence segregation. While not always enriched at the substrate surface, the interface between the air/vacuum and polymer was found to be enriched in blends and copolymers containing more short side branches [17–19]. Experiments conducted by Walton et al showed
that while linear PEO tended to be depleted at the surface when blended with linear PMMA it would be enriched when a branched architecture was introduced to the linear PMMA [16].

Likewise, Foster et al. showed that star branched polymers were preferred at surfaces relative to linear chains [20].

Cyclic polymers are of interest for blending purposes because of their interesting rheological properties, in the bulk and as additives, as well as appearing as biological macromolecules such as cyclic DNA and chromatins [21–23]. Moreover, the physics of cyclic polymers remains a challenge to theory due to the lack of effective theoretical tools to account for the topological non-crossing constraints of an ensemble of cyclic chains. While cyclic polymers have interesting properties they are difficult to synthesize; however, recent advances in anionic polymerization techniques have allowed for higher MW and better control of cyclic polymers [24]. A scalable synthesis for cyclic poly($\epsilon$-caprolactone) has also been reported using a rotaxane precursor and rotaxane protocol [25]. Cyclic polymers were also synthesized using a combination of RAFT and a UV light induced Diels-Alder click reaction. This method was able to efficiently produce ring structures at room temperature without the need for catalysts or inert atmosphere [26].

2.4 Theoretical and computational study of architecturally driven surface segregation in polymer blends

This section will review different methods for studying polymers at a surface relevant for architecturally driven surface segregation. It will focus on the coarse-grained Self-Consistent Field Theory, designed to capture large length-scale physics, and the atomistic Wall-PRISM theory, designed to capture the physics of monomer scale packing. I also briefly discuss Monte Carlo simulations for polymers.

2.4.1 Self-Consistent Field Theory

Self-Consistent Field Theory (SCFT), developed by Edwards[27, 28] and others[1, 29–33], has been used to model both bulk and surface properties of polymers. The main assumption
of SCFT is that any polymer molecule in the system can be modeled as a single polymer molecule moving in an average potential that is produced by the other polymer chains. This assumption holds for every chain, and so these mean field potentials are determined self-consistently. The single test chain in a homogeneous environment (e.g. in a pure melt) can be assumed to have Gaussian (random-walk) statistics, and so one can then compute the conformational statistics of this test chain in a mean chemical potential field. When coarse-grained, the mean field can be assumed to be local and furthermore proportional to the average density. This can be extended to apply to system not only in the bulk but also at a free air/vacuum surface or interacting with a substrate. Herein I assume these surfaces are neutral and hard; however, attractive or repulsive potentials between the chains and each other as well as the chains and the surfaces can be included as an addition to the mean field potential. SCFT has been successfully applied to a wide variety of polymer systems including brushes, copolymers and different blends [34–36].

Wu and Fredrickson [1] developed a linear response analysis of the SCFT for blends of architecturally different polymers. Molten blends of linear chains with different branched and cyclic chains were studied. Their linear response approximation solved the SCFT for these blends in the limit that the liquid is incompressible. They showed that the accounting for the complex conformational entropy differences of a chain at the surface compared to the bulk would result in end points being favored at the surface while branching points would be disfavored. Moreover, these entropic contributions could be effectively described as an effective surface potential for chain ends and branch points. A simple way to understand the surface preference for ends is to recognize that a chain with an end near the surface would only have to avoid the surface a single time, whereas a typical monomer in the middle of the chain being near the surface would result in two chain paths needing to avoid the surface. The net preference for chain ends is equivalent to that resulting from chain ends attracted to the surface via a potential. The branch points on the other hand would be more and more disfavored at the surface with more branching arms, since each of the various chain
paths emanating from the branch point would need to avoid the wall completely. This is entropically unfavored since the number of polymer configurations is much more limited and was shown to be equivalent to an effective repulsive potential between the branch point and the surface. They concluded that for linear/branched blends both polymer types will have preference for the ends. For systems where the linear chains are much smaller the entropic penalty is such that the small linear chains should be preferred.

While this linear response theory is a good starting point for understanding different factors in architecturally driven surface segregation, it is limited because only the chain ends and branch points are accounted for while the polymer topology and lengths of chains between end and branch points are not. NR studies of a family of pom-pom/linear blends with the same total molecular weight and pom-pom topology, but differing in lengths between the branch points, showed differences with the predictions of the linear response theory of Wu and Fredrickson, showing a more complete model was needed[37].
relative chains and vice-versa. A model of this pom-pom polymer is shown in Figure 2.4 along with a star polymer molecule. Linear response theory predicts the same enrichment regardless of the length of the linear portion. The SCFT, however, successfully captured the variation of integrated surface excess of 6-pom observed in the NR scattering studies. The “integrated excess” is defined to be the integral of the excess volume fraction over the surface composition profile. See section 4.2 and Equation 4.19 for more discussion and explanation of the integrated excess. See Figure 2.5 below.

![Figure 2.5](image)

**Figure 2.5**: Comparison of SCFT and NR results for the integrated excess of the various 6-pom polymers at the surface of a blend with linear polymers [2]. All chains are 36k in molecular weight. The bulk volume fractions of linear and pom-pom polymer are $\phi_{\text{linear}} = 0.8$ and $\phi_{\text{pom}} = 0.2$ respectively.

Using the same linear response SCFT framework above, Wu and Fredrickson[1] also predicted that in linear and cyclic blends, cyclic chains would be preferred at the surface at exactly twice the composition as that in the bulk. However while this is also driven by
differences in conformational entropy, the surface preference cannot be simply modeled as a surface potential, as it can be for ends and branch points. Hu [2] carried out numerical SCFT calculations on linear/cyclic blends, and found a surface excess of cyclic polymers regardless of chain size, in agreement with the linear response theory by Wu and Fredrickson. Recent NR studies by Foster et al. also found that in agreement with linear response theory for larger polymers, the cyclics had a surface composition nearly double their bulk value for large MW blends [38]. However, as the molecular weight of the polymer chains decreased, the NR results showed a crossover to a surface preference for the linear chains instead. As mentioned above, packing effects become more dominant for smaller chains and could be important in surface segregation. Chapter 4 of this thesis will cover my work using packing effects to explain the preference for linear chains in linear/cyclic blends for low molecular weights.

2.4.2 Wall-PRISM and Monte Carlo methods

I present here a brief introduction to Wall-PRISM as it is related to my work with surface segregation; however, as an extension of PRISM theory, it is explained in greater detail in section 2.6 below.

Wall-PRISM is an extension of PRISM theory developed by Yethiraj and Hall to model polymers near a surface [39]. Because it is based on PRISM, an atomistic theory, Wall-PRISM can incorporate molecular interactions such as packing of polymer molecules at a surface. The packing of polymers, a feature that is not accounted for in coarse-grained theories such as SCFT, has been shown to be important for smaller polymers. In keeping with the theme of polymer blends based on architectural differences, Yethiraj looked at systems involving linear and star polymers[40]. Yethiraj found that linear polymers are preferred directly at the surface due to packing effects, however, the integrated excess of the linear vs star polymers over a molecular length scale near the surface typically showed preference for star polymers in most cases. The star polymer enrichment increases with the length of the side chain arms. Moreover the segregation behavior was dependent on the
density of the system. For low density systems more approximate to a solution than a melt, the star polymers were typically found at the surface regardless of the length of the branch chains. As the density was increased to correspond to that of a melt, linear chains were again preferred at the surface because of their packing efficiency for the blends with short branched star chains. Only the long branch length blends would show star preference at high density over linear chains. This variability in surface preference contrasts with the prediction of the coarse-grained linear response theory by Wu and Fredrickson that for similar MW polymer blends, the more branched polymer should be preferred because of the attractive potential of the ends with the surface.

Monte-Carlo simulations were also performed in conjunction with Wall-PRISM calculations to look at the effect of chain stiffness on binary segregation [41, 42]. These studies looked at blends of identical chains with different stiffness levels. While it was always found that the less flexible chains were preferred at the surface, the excess of stiff molecules in simulation was less than predicted by the theories. While packing of the stiff molecules was important, it was also argued that the entropy penalty for the stiffer chains explained the lower surface excess in the simulations. At lower densities it was then the more flexible chains that would be favored at the surface due to the reduced packing and more important configurational entropy effects. Further simulation of blends of stars and linear polymers at surfaces was studied by Yethiraj [43]. The polymers were modeled as hard spheres with attractive potentials between the polymers and between the polymers and the walls. These systems also showed a preference for the branched system over the linear system even with the additional attractive potentials.

2.5 Reference Interaction Site Model (RISM)

Integral equation theories of liquids have been shown to be a useful tool for understanding the liquid state [44]. Initially developed for atomic liquids and liquids of small molecules, such liquid-state theories incorporate atomic level detail and capture both packing effects as well as longer-ranged pair distribution information. The basic premise is to use atomic
and molecular interaction potentials along with a diagrammatic analysis of the equilibrium
distribution functions to generate relations, typically integral equations, for the correlation
functions between atoms and molecules in a solution. The correlation functions can be
directly compared with scattering experiments, e.g. whose resulting structure factors are the
Fourier transforms of the correlation functions. These integral equations generally involve
approximations, or “closures”, that allow solution of the equations. In the remainder of this
chapter, I present an overview of these methods, and give a more detailed exposition in the
next chapter.

One of the most successful liquid state theories of the past 50 years has been the Reference Interaction Site Model (RISM) theory. First developed in the 1970’s by Chandler and Andersen [45] as an extension of the previous integral equation theories of atoms. It was based on the Ornstein-Zernike integral equation theory for atomic liquids, but incorporated the intramolecular correlation in addition to the intermolecular correlations propagated within a molecular liquid. The molecules are represented as collections of spherical sites connected together, and so the intramolecular correlation function is an input into the calculation of intermolecular correlation functions. The spherical sites can represent single atoms in molecules or larger multi-atom sites. Initial work on RISM was carried out with small molecule diatomics such as molecular nitrogen and bromine [46], then also polyatomic rigid molecules such as benzene, carbon tetrachloride and carbon disulfide [47], showing agreement with both simulation and scattering experiments. RISM also showed the ability to capture the behavior of non-rigid molecules and was applied to a variety of flexible molecules including alkane chains [48, 49].

2.6 Polymer Reference Interaction Site Model (PRISM)

The RISM theory was extended from small molecules to macromolecules by Curro and Schweizer [50] in 1987, and their resulting theory is named Polymer RISM or PRISM. As with RISM, the intramolecular correlation function is needed as an input to the theory, allowing calculation of intermolecular correlations. The development of PRISM thus relied on the
assumption that polymer chains will exhibit ideal conformations in the polymer melt. This assumption of ideal conformations is justified by the screening of the intramolecular excluded volume interactions by the intermolecular interactions between chains. This represents a significant simplification, since the intramolecular correlation function needed by RISM type theories can be precalculated for a given chain model. There are a variety of polymer chain models with varying degrees of realism that have been used in PRISM for generating the intramolecular correlation function, such as the freely-jointed [51–53], Gaussian [3], and semiflexible chains [5, 54].

2.7 Self-Consistent PRISM (SC-PRISM or One-Chain SC-PRISM)

In a pure melt chain conformational non-ideality is less pronounced especially in dense systems, however for polymer alloys and blends it was shown by both experiment and simulation [55, 56] that the non-idealities are non-negligible. To address this, the self-consistent PRISM (SC-PRISM) [5] theory was developed by Schweizer, Honnell and Curro. In this theory the intramolecular correlations would be determined self-consistently. To do this a single polymer molecule would be simulated with molecular dynamics or Monte Carlo methods. The benefit of simulating the polymer is that monomers will be repelled rather than overlap, more accurately representing the steric interactions within the polymer chain. Care needed to be take when simulating the single chain however, because incorporating only the bare (e.g. Lennard-Jones) potential between sites leads to a self-avoiding chain, more representative of a polymer in a solvent rather than in a dense melt. Also, the single chain simulation would in this case not account for polymer blend composition. To account for these effects, the polymer chain is simulated in a mean-field solvation potential in addition to the bare potential for the intrachain interactions. The mean-field potential is calculated self-consistently from the PRISM equations. As such, SC-PRISM is a prototypical hybrid theory/simulation approach for modeling polymers.

SC-PRISM theory has been applied to many systems including polyolefins blends. Heine et. al [4] looked at miscibility of blends of isotactic polypropylene/polyethylene blends and
compared results with both molecular dynamics (MD) studies as well as scattering experiments. They were able to evaluate the changes in individual chain energies contributing to the heat of mixing, from both the MD studies as well as the SC-PRISM theory. The major intramolecular energy contributions to the heat of mixing were found to be the torsional and van der Waals energies. More current work involving SC-PRISM centers around understanding surface modification of nanoparticles with polymer chains. Martin et. al [6] were concerned with understanding how polymer size distributions on the surface of the coated nano-particles changed the aggregation behavior in solution. They found that with increasing polydispersity the attractive well between particles was reduced and an increase in particle dispersity should be seen. Also, they found that an increase in nano-particle size would cause an even greater dispersion because monomer crowding on the flatter surface of the larger nano-particle would enhance the polydispersity effect of the grafted polymer chains. Banerjee et al. [57] looked at systems of dilute nano-particles in a solution with non-microphase separating AB multiblock copolymers and the effect of the multiblock copolymer chemistry on nano-particle dispersion and misibility in polymer melts. The diblock systems were such that one monomer type was more strongly adsorbed to the nano-particle than the other. They found that the effect of wettablilty of the polymer to the nano-particle surface on interparticle dispersion was different than that for the homopolymer systems. They also found that the dispersion of the nano-particles was strongly dependent on the relative sizes of the nano-particles and block chain lengths. Miscibility of larger particles was found to correlate with longer block sizes while small particles were insensitive to the block size.

2.8 Two-Chain Self-Consistent PRISM (Two-Chain SC-PRISM)

While SC-PRISM was able to improve on pure PRISM there were still some notable deficiencies. As mentioned above, simulations of a single chain can more realistically capture correlations within a single chain, but can still not capture the correlations between chains. A solution to this issue was proposed by Li and coworkers [7]. Instead of simulating a single chain in SC-PRISM, which we thus also call One-Chain SC-PRISM, one can simulate
two chains, which we call Two-Chain SC-PRISM. Results from (One-Chain) SC-PRISM are compared against those from the two chain method, Two-Chain SC-PRISM, in Figure 2.6 below. By using two chains in the simulation, the short range correlation functions can be seen to be improved. However with existing methods this new two-chain hybrid theory method could not be applied to systems with more than a single site type. The Two-Chain SC-PRISM calculations were thus limited to only polyethylene. Improvements to the radial distribution function, \( g(r) \), at small \( r \) values can be seen in Figure 2.6 for parameters corresponding to linear polyethylene (PE). While these differences may appear subtle, because interaction energies depend strongly on how close two monomers can approach each other, these differences can result in relatively large differences in mixing energies that can determine bulk or surface segregation.

2.8.1 Summary

To summarize, neutron reflectivity is an experimental technique able to capture surface segregation in polymer blends. Isotopic differences used to enhance contrast must also be accounted for, as differences in bond distances and polarizability can lead to preference of the deuterated polymers at the surface. Segregation due to architectural differences for linear and branched systems can be understood qualitatively, at the level of linear response theory, by only considering the number of branch and end points and these interactions can be thought of as surface potentials. However, for more accurate results the chain lengths between ends and branch points must also be taken into account. Coarse-grained SCFT studies can produce good results for explaining surface segregation that accounts for such variations in distance between ends and branch points, as well as for both enthalpic and entropic driving forces. However simulation and Wall-PRISM theories have shown that local interactions and packing may dominate for the limit of low molecular weight chains. The synthesis and study of cyclic chains is an interesting and growing field, with improved synthetic routes stimulating more studies on cyclic polymers. Cyclic chains have interesting properties because of their more compact size, non-crossing topology, and lack of chain ends.
Figure 2.6: The radial distribution functions, $g(r)$, between monomers for $N = 24$ and $N = 66$ polymer melts, where $N$ is the number of monomer units in the polymer chain. The monomers interact with the repulsive Lennard-Jones potential with parameters for the non-bonded interactions of $\epsilon/k = 46.5 \, K$ and $\sigma = 3.93 \, \text{Å}$. The bond length is $1.54 \, \text{Å}$ for both systems. For $N = 24$ the temperature was $405 \, K$ with a density of $\rho = 0.03104 \, \text{Å}^{-3}$ and for $N = 66$ the temperature was $448 \, K$ and a density of $\rho = 0.3104 \, \text{Å}^{-3}$. The intramolecular chain parameters are reported in [58]. The points are from MD simulations [59], the dashed curves are the One-Chain SC-PRISM results and the solid lines are from the single site Two-Chain SC-PRISM theory [7].
High molecular weight cyclics are preferred at the surface when blended with linear chains, as predicted by both linear response and SCFT theories and observed by NR experiments. However, low molecular weight cyclic chains are observed to be disfavored at a surface in favor of linear chains due to packing effects. I will test this hypothesis in this thesis using Wall-PRISM theory.

RISM theory is an integral equation theory and the generalization of the OZ relation for molecules. RISM has been successfully applied to many different molecular systems. PRISM is the polymer extension of the RISM theory. Both RISM and PRISM require input of the intramolecular correlation function as well as a closure relation to solve the RISM or PRISM equations. The SC-PRISM method generates the intramolecular correlation function by using a Monte Carlo simulations and the pivot algorithm of a single polymer chain. The single chain is simulated in a solvation potential that mimics the presence of the other chains in the melt. SC-PRISM has been successfully applied to a range of systems including polymer melts. A new method involving the simulation of two molecules together has shown to give improved radial distribution functions over one-chain SC-PRISM when compared with MD results. Two-chain results also allow the closure relation to be found directly from the simulation results rather than having to be approximated.
CHAPTER 3
RISM/PRISM THEORIES

This section is dedicated to covering the RISM/PRISM theories that will be used in this thesis, as such, this chapter will be referenced extensively. The chapter will begin with the RISM theory for small molecules followed by the PRISM theory for macromolecules. The PRISM theory will then be expanded in the sections that follow, first, with Wall-PRISM for polymers at a surface while last two sections will discuss the one-chain self consistent PRISM (SC-PRISM) and the Two-Chain self consistent PRISM (Two-Chain) both of which utilize Monte Carlo simulations of the polymer chains. The Two-Chain SC-PRISM theory is the basis for the remainder of the work in the thesis, however, because a great deal of it is method development, I will focus on small molecules for testing in Chapters 5 and 6.

3.1 Integral equation theory

As an introduction, consider an equilibrated system of $N$ uniform atoms modeled as spherical particles in a volume, $V$, at a constant temperature $T$. The structure of the system can be measured with the radial distribution function, $g(r)$, and is defined with Equation 3.1, where the number density, $\rho$, is defined by $\rho = N/V$ and the angled brackets denote the ensemble average.

$$\rho^2g(r) = \left\langle \sum_{i \neq j, i=1}^{N} \delta(r_i)\delta(r - r_j) \right\rangle$$  (3.1)

The function $g(r)$ is related to the probability that two particles are at a distance $r$ from one another in the uniform liquid. A more intuitive way to think of this is to consider the function $\rho g(r)$ as giving the density of particles at a distance $r$ from an arbitrarily chosen particle. In the limit of very large separation, the density of the particles will be equal to $\rho$, therefore, $g(r)$ is expected to approach unity. In this way I can use $g(r)$ to talk about both the short range local structure and long range bulk behavior of a liquid. Also, the
thermodynamics of the system can then be calculated using the information from \( g(r) \) as well as the interaction pair potential energies.

Within liquid state theory, a foundational equation is the Ornstein-Zernike (OZ) relation (see Equation 3.2). The integral on the right hand side of this equation is the three-dimensional convolution (see Equation 3.5) of \( C(r) \) and \( h(r) \), where \( C(r) \) is the direct correlation function and \( h(r) \) is the total correlation function and is related to the radial distribution function by the equation \( h(r) = g(r) - 1 \). While the total correlation function is fairly straightforward, the direct correlation function definition is more subtle.

\[
h(r) = C(r) + \rho \int C(|\mathbf{r} - \mathbf{r}'|)h(r) d\mathbf{r}' \quad (3.2)
\]

To understand the role of \( C(r) \), I start by looking at Equation 3.2 and notice I can rewrite it as an iterative expansion between arbitrary labeled atoms in the system. For instance, if I want to look at the total correlation between particles labeled p1 and p2 I can define the expansion from Equation 3.2 as:

\[
h(p1, p2) = C(p1, p2) + \rho \int C(p1, p3)C(p3, p2) dp3
\]
\[
+ \rho^2 \int \int C(p1, p3)C(p3, p4)C(p4, p2) dp3 dp4 + \ldots \quad (3.3)
\]

The above equation can be rewritten in terms of the convolution using Equation 3.5.

\[
g(r) * f(r) = \int f(\mathbf{r}')g(\mathbf{r}' - \mathbf{r}) d\mathbf{r}' \quad (3.5)
\]

For a visual representation see Figure 3.1. Because the site labeling is arbitrary, the system is assumed to be isotropic and each \( C(r) \) is short ranged. The iterative expansion in Equation 3.3 simplifies to the following series of convolutions.

\[
h(r) = C(r) + \rho C(r) * C(r) + \rho^2 C(r) * C(r) * C(r) + \rho^3 C(r) * C(r) * C(r) * C(r) + \ldots \quad (3.6)
\]

The RHS of the above equation can be interpreted as stating that two particles at a given distance \( r \) will be correlated with one another first by their own direct correlation, which is
the first \( C(r) \) on the RHS, and then subsequently through the higher-order density terms that are the chains of correlation constructed with direct correlation functions of intermediate atoms. For instance, see in Figure 3.1, how there is a path from \( p_1 \) to \( p_2 \) made by direct correlation functions going from \( p_1 \rightarrow p_3 \rightarrow p_4 \rightarrow p_2 \). This leads to the model that the direct correlation function captures some multi-body effects.

![Diagram](image)

Figure 3.1: Illustration of the definition of the total correlation between particle \( p_1 \) and \( p_2 \) in terms of chains of the direct correlation function.

### 3.1.1 Reference Interaction Site Model (RISM)

The RISM theory of Chandler and Andersen [60] is a generalized form of the Ornstein-Zernike relation for isotropic molecular liquids. In RISM, the molecules are composed of spherical sites that interact. These sites can represent atoms or polyatomic subunits within a larger molecule. The main difference between the OZ relation and the RISM theory is that within RISM, correlations can now be propagated within a molecule. \( C(r) \) is still the direct correlation function and \( h(r) \) is still the total correlation function. These two functions describe correlations between sites on separate molecules and so are called the intermolecular correlation functions while the correlations within the molecules are handled by the intramolecular radial distribution function, \( \omega(r) \), and whose Fourier Transform is defined in Equation 3.12. The RISM equation of Chandler and Andersen is defined below in
Equation 3.7. If the RISM equation is applied to an atomic case, $\omega(r)$ will be the identity matrix, and the OZ relation is recovered.

$$h(r) = \int dr' \int dr'' \omega(|r-r'|) C(|r'-r''|) [\omega(r') + \rho h(r'')] \tag{3.7}$$

$$h(r) = \omega(r) * C(r) * \omega(r) + \rho \omega(r) * C(r) * h(r) \tag{3.8}$$

$$\tilde{\gamma}(v(r)) = \hat{v}(k) = \int v(r)e^{-2\pi ikr} dr \tag{3.9}$$

$$\hat{v}(k) = \int_0^\infty v(r) \frac{\sin(kr)}{kr} \frac{4\pi r^2}{k} dr \tag{3.10}$$

Taking the Fourier Transform (FT), I can express the RISM equation in $k$-space. The definition for the 3d FT is given in Equation 3.9. The benefit of doing the FT is that convolutions of functions in $r$-space become simple multiplication of functions in $k$-space as shown in Equation 3.11. This leads to the simplification of the $r$-space convolution definition of RISM to the $k$-space definition of RISM in 3.11.

$$\hat{\hat{h}}(k) = \hat{\omega}(k) \hat{C}(k) \hat{\omega}(k) + \rho \hat{\omega}(k) \hat{C}(k) \hat{h}(k) \tag{3.11}$$

where $\hat{\omega}(k)$ is the intramolecular structure and is defined below for rigid molecules. Here $\alpha$ and $\gamma$ are the site types and each molecule will contain $M_\alpha$ and $M_\gamma$ total sites.

$$\hat{\omega}_{\alpha\gamma}(k) = \frac{1}{M_\alpha} \sum_i^{M_\alpha} \sum_j^{M_\gamma} \hat{\omega}_{ij}(k) \tag{3.12}$$

and where $\hat{\omega}_{ij}(k)$ is defined below and $L_{ij}$ is the distance between site $i$ and site $j$.

$$\hat{\omega}_{ij}(k) = \frac{\sin(kL_{ij})}{kL_{ij}}$$

### 3.1.2 Polymer Reference Interaction Site Model (PRISM)

Curro and Schweizer developed PRISM theory, which an is the extension of RISM for polymers [61]. In PRISM, the polymers are assumed to exhibit ideal behavior in the melt. Also, end groups are, for simplicity, not treated as distinct sites, as the size of the molecule is so large that the end effects of the polymer chains is reduced and assumed to be negligible.
Consider a case now similar to the first of spherical polymers but now a polymer melt at equilibrium with a volume of \( V \), that is comprised of \( N \) identical polymer chains at temperature \( T \). Each chain will be composed of \( M \) spherical monomer sites. The monomer sites could be of the same type or multiple types. If the polymers are composed of identical site types, the PRISM equations will be scalar in nature, however, if multiple sites are required for the polymer chains (e.g. in isotactic polypropylene, iPP) the PRISM equations will be of a matrix form. Care should be taken when expressing the terms by the respective molecular and site densities. Here I express the full RISM/PRISM equations:

\[
\hat{H}_{\alpha\gamma}(k) = \rho_{\alpha} \rho_{\gamma} \hat{h}_{\alpha\gamma}(k) \quad (3.13)
\]

\[
\hat{\Omega}_{\alpha\gamma} = \rho_{\alpha} \hat{\omega}_{\alpha\gamma}(k) \quad (3.14)
\]

\[
\hat{H}(k) = \hat{\Omega}(k)\hat{C}(k)\hat{\Omega}(k) + \hat{\Omega}(k)\hat{C}(k)\hat{H}(k) \quad (3.15)
\]

The structure factor, defined in Equation 3.16, is also useful as it can be compared to experimental scattering data. The intensity of the scattering data, \( \hat{I}(k) \), is related directly to the structure factor by way of the relation \( \hat{I}(k) = \sum_{\alpha\gamma} b_{\alpha} b_{\gamma} \hat{S}_{\alpha\gamma}(k) \) where \( b_{\alpha} \) is the scattering cross section of sites of type \( \alpha \) and \( \hat{S}_{\alpha\gamma}(k) \) is defined below.

\[
\hat{S}_{\alpha\gamma}(k) = \hat{\Omega}_{\alpha\gamma}(k) + \hat{H}_{\alpha\gamma}(k) \quad (3.16)
\]

To solve the RISM/PRISM Equation (3.15) the chain structure, \( \hat{\omega}(k) \), is required as input. This can be determined by using a polymer chain model mentioned in 2.6 or by using simulation information from Monte Carlo or Molecular Dynamics. Even after the intramolecular radial distribution function information is determined the PRISM Equation 3.15 is still under defined. To fully define the system of equations, closure relations are introduced that give approximations for how two intermolecular correlation functions, \( h(r) \) and \( C(r) \), are related. The next section will cover these closure equations more fully.
3.1.3 Closure equations

To solve the RISM/PRISM equations, as mentioned previously, information of not only the chain structure is required but also an approximation of the relationship of the two intermolecular correlation functions of $h(r)$ and $C(r)$ is needed. Theory and simulations have shown that in many circumstances $C(r)$ is effectively a short-ranged function [44]. There are several approximations for how to relate $h(r)$ and $C(r)$, however, there is no single best closure relation and so the choice of closure will be system specific (site interaction potentials, pressure, temperature, etc.) [62].

The Percus-Yevick (PY) closure is derived using Percus’ method of a perturbation expansion as well as by Stell’s graph summation method [62]. The PY closure is typically used and most successful in approximating the local structure of simple hard-core liquids [44]. The PY closure is given by the following equations.

\[
C_{\alpha\gamma}(r) = (1 - e^{\beta u_{\alpha\gamma}(r)}) \quad r < \sigma_{\alpha\gamma} \\
g_{\alpha\gamma}(r) = 0 \quad r > \sigma_{\alpha\gamma}
\]  

(3.17)

where $\sigma_{\alpha\gamma}$ are the minimum contact distances between two sites. For hard spheres, I can approximate the site pair potential as, $u_{\alpha\gamma}(r) = \begin{cases} \infty & r < \sigma_{\alpha\gamma} \\ 0 & r > \sigma_{\alpha\gamma} \end{cases}$, which simplifies Equation 3.17 to the following relationship

\[
PY \ Closure = \begin{cases} C_{\alpha\gamma}(r) = 0 & r > \sigma_{\alpha\gamma} \\ g_{\alpha\gamma}(r) = 0 & r < \sigma_{\alpha\gamma} \end{cases}
\]  

(3.18)

where $\sigma_{\alpha\gamma} = \frac{1}{2}(\sigma_\alpha + \sigma_\gamma)$ with $\sigma_\alpha$ and $\sigma_\gamma$ being the diameters of the respective sites of type $\alpha$ and $\gamma$.

Another common closure equation is the hypernetted chain (HNC) [44]. An exact cluster expansion expression allows the direct correlation function to be defined as

\[
C_{\alpha\gamma}(r) = -\beta V_{\alpha\gamma}(r) + h_{\alpha\gamma}(r) + d_{\alpha\gamma}(r) - \ln(h_{\alpha\gamma}(r) + 1)
\]  

(3.19)
where $V_{\alpha\gamma}(r)$ is the potential between sites $\alpha$ and $\gamma$ and $d_{\alpha\gamma}(r)$ is the bridge function. For the HNC approximation, the bridge function is assumed to be zero. This results in the following equation as the definition of the HNC closure.

$$C_{\alpha\gamma}(r) = -\beta V_{\alpha\gamma}(r) + h_{\alpha\gamma}(r) - \ln(h_{\alpha\gamma}(r) + 1) \quad (3.20)$$

While the HNC closure contains more classes of diagrams from the cluster expansion than PY and is able to handle attractive potentials where the PY closure tends to work best with only repulsive potentials, HNC will also tend to predict much larger fluctuations in the bulk density especially at low density with high MW polymer chains leading to long length scale inaccuracies [63].

Finally, another common closure is the Martynov-Sarkisov (MS) closure [64], given by

$$C_{\alpha\gamma}(r) = -\beta V_{\alpha\gamma}(r) + h_{\alpha\gamma}(r) - (1 + \ln(h_{\alpha\gamma}(r) + 1))^2 - \frac{1}{2} \quad (3.21)$$

This closure is a generalization of the HNC closure and has shown better accuracy than PY for hard core spherical molecules. The MS closure, as for the HNC closure, is non-zero outside of the hard core diameter [64, 65].

### 3.1.4 Picard iteration solution for PRISM equations

For much of this thesis the RISM/PRISM equations will be solved using the Picard iteration technique [44]. A diagram of this method is shown in Figure 3.2. The method I used employs a smooth, continuous function, $\gamma(r)$, that relates the two intermolecular correlation functions of $h(r)$ and $C(r)$ through the following relation

$$\gamma_{\text{Start}}(r) = h(r) - C(r) \quad (3.22)$$

For the first iteration there is a guess made for $\gamma(r)$. Often the first guess is $\gamma(r) = 0$. Otherwise, it will be the $\gamma_{\text{Mix}}(r)$ (defined later) from the previous iteration. Using the desired closure equation from section 3.1.3 and Equation 3.22, a new value for $C(r)$ is calculated. An FFT is then performed to give $\hat{C}(k)$. Using the RISM/PRISM Equation 3.11 or 3.15 and Equation 3.22 a new estimate for $\hat{\gamma}(k)$ can be found using the following equation:
\[ \hat{\gamma}_{\text{New}}(k) = [1 - \rho \hat{\omega}(k) \hat{C}(k)]^{-1} \hat{\omega}(k) \hat{C}(k) \hat{\omega}(k) - \hat{C}(k) \] 

(3.23)

The inverse FFT is used to calculate \( \gamma_{\text{New}}(r) \). The average fractional difference (AFD)

\[ AFD = \sqrt{\frac{1}{N_{\text{grid}}} \sum_{i=1}^{N_{\text{grid}}} \sum_{\eta,\kappa} \left( \frac{\gamma_{\text{New}}{\eta\kappa} - \gamma_{\text{Old}}{\eta\kappa}}{\gamma_{\text{New}}{\eta\kappa} + \gamma_{\text{Old}}{\eta\kappa}} \right)^2} \] 

(3.24)

is then calculated to determine if the solution is converged. For the work on bulk and Wall-PRISM a tolerance of \( AFD < 10^{-7} \) was used as the convergence criterion. If the AFD indicates that it is not converged, the original \( \gamma_{\text{Old}}(r) \) is then mixed with the new \( \gamma_{\text{New}}(r) \) by the relation \( \gamma_{\text{Mix}}(r) = (1 - \alpha) \gamma_{\text{Old}}(r) - \alpha \gamma_{\text{New}}(r) \) where \( 0.0 < \alpha < 1.0 \). \( \alpha \) for work in this thesis is typically in the range of \( 0.05 - 0.01 \). Better convergence is found by mixing the two \( \gamma(r) \) functions.

Figure 3.2: The Picard iteration scheme used here to solve the RISM/PRISM Equations.
3.2 Wall-PRISM theory

Wall-PRISM theory [39], of Yethiraj and Hall, is an extension of PRISM theory and was developed for studying polymer melts near a hard surface. It has been applied to a variety of molecular and polymer systems such as alkane chains [66], complex polymer architectures [67] and polyolefin blends of linear and branched polymers [40].

The system of interest in Chapter 4 is a binary blend of two different polystyrene polymer chain architectures. The two topologies will be the linear chain and a cyclic chain. These will be designated as $A$ for monomer site units on the linear chain and $B$ for monomer site units on the cyclic chain. There will be a third component which will be the wall site, to be designated $w$. The $3 \times 3$ (sites $A$, $B$ and $w$) matrix PRISM Equation 3.15 will be first solved for the $\hat{h}(k)$ matrix elements in the limit of the wall site, $w$, being infinitely dilute. Second, the PRISM equation will be solved with the wall site in the limit of infinite radius resulting in an approximation of a flat surface. The result is the following equations (3.25, 3.26 and 3.27) for the total correlation functions of the polymer melt in the bulk.

\[
\rho_A \rho_A \hat{h}_{AA}(k) = \hat{S}_{AA}(k) - \rho_A \hat{\omega}_{AA}(k) \tag{3.25}
\]
\[
\rho_A \rho_B \hat{h}_{AB}(k) = \hat{S}_{AB}(k) \tag{3.26}
\]
\[
\rho_B \rho_B \hat{h}_{BB}(k) = \hat{S}_{BB}(k) - \rho_B \hat{\omega}_{BB}(k) \tag{3.27}
\]

The total correlation between the polymers and the wall site are then given by the Wall-PRISM Equations,

\[
\rho_A \hat{h}_{Aw}(k) = \hat{S}_{AA}(k) \hat{C}_{Aw}(k) + \hat{S}_{AB}(k) \hat{C}_{Bw}(k) \tag{3.28}
\]
\[
\rho_B \hat{h}_{Bw}(k) = \hat{S}_{BB}(k) \hat{C}_{Bw}(k) + \hat{S}_{AB}(k) \hat{C}_{Aw}(k) \tag{3.29}
\]

These can be written more compactly as

\[
\rho_a \hat{h}_{aw}(k) = \sum_{\gamma} \hat{S}_{a\gamma}(k) \hat{C}_{\gamma w}(k) \tag{3.30}
\]

where $\hat{S}_{a\gamma}(k)$ is the partial structure factor and all other terms are as defined above.
As mentioned previously, the spherical wall site diameter must be grown to approximate a flat substrate or surface. To model two surfaces an annulus structure is used (see Figure 3.3). The polymer blend will be in the gray area while the blue is defined to be the impenetrable, hard surface. I then define $R_2 - R_1 = H$ where $H$ is the width of the system containing the polymer. For some cases this could be the width of the slit pore for confined polymers but in this thesis it will represent the approximate width of the polymer blend film, with the solid substrate being one wall and the air/vacuum being represented by the second. The polymer blend will be allowed to reside between the two surfaces such that $R_1 < r < R_2$. The solvent site will now be grown in the limit of $R_1, R_2, R_3 \to \infty$, where $R_3$ is the size of the outer diameter of the wall “particle”.

To facilitate taking this limit, the compact form of the Wall-PRISM equation (3.30) will be used. It is rewritten in real space as a three-dimensional convolution but the coordinate system will be changed to a bipolar one with the origin at the center of the solvent site that will be acting as the surface.

$$\rho_a h_{\alpha w}(r) = \frac{2\pi}{r} \sum_\gamma \int_0^\infty t S_{\alpha \gamma}(t) dt \int_{|r-t|}^{r+t} y C_{\gamma w}(y) dy \tag{3.35}$$

This means $r$ will be the position of a site on a polymer chain with respect to the new origin, as depicted in Figure 3.4. The following variables are also defined
\[ z = r - (R_1 - \frac{H}{2}) \]  
\[ s = y - (R_1 - \frac{H}{2}) \]  

(3.36)  
(3.37)

The recast Wall-PRISM Equation 3.35, along with Equations 3.36 and 3.37, are solved in the limit as \( R_1 \to \infty \) to give the following result.

\[ \rho_{\alpha} h_{\alpha w}(z) = 2\pi \sum_{\gamma} \int_{0}^{\infty} t \sigma_{\alpha \gamma}(t) dt \int_{|z-t|}^{z+t} C_{\gamma w}(s) ds \]  

(3.38)

The polymer-wall interaction will be assumed to be a hard sphere like interaction, and so the PY closure (see eqn. 3.18) will be used. For the modified PRISM Equations the PY closure is:

\[ \text{PY Closure}_{Wall-PRISM} = \begin{cases} 
  h_{\alpha w}(z) = -1 & |z| > \frac{H}{2} \\
  C_{\alpha w}(z) = 0 & -\frac{H}{2} < z < \frac{H}{2}
\end{cases} \]  

(3.39)

However, using this form of the PY closure produces some mathematical difficulties. When calculating the Fourier Transform, \( C_{\alpha w}(z) \) will not approach zero as \( z \to \pm \infty \). To make a more mathematically friendly set of equations on which to perform the Fourier Transform, a modified version of the direct correlation function will be used which will have the asymptotic portion designated \( C^{z=\pm \infty}_{\alpha w}(z) \) (see Equation 3.42) subtracted off so it will approach zero for large \( z \). The total correlation function, \( h_{\alpha w}(z) \), also does not decay to zero as \( z \to \pm \infty \) so \( g_{\alpha w}(z) \) will be used instead (see Equation 3.43).

\[ C_{\alpha p}(z) = C_{\alpha w}(z) - C^{z=\pm \infty}_{\alpha w}(z) \]  

(3.40)

\[ g_{\alpha w}(z) = 1 + h_{\alpha w} \]  

(3.41)

\[ C^{z=\pm \infty}_{\alpha w}(z) = \frac{\rho_A \hat{S}_{BB}(0) - \rho_B \hat{S}_{AB}(0)}{\hat{S}_{AB}(0) - \hat{S}_{AA}(0)\hat{S}_{BB}(0)} \]  

(3.42)
\[ \rho_{\alpha}g_{\alpha w}(z) = 2\pi \sum_{\gamma} \int_{0}^{\infty} t S_{\alpha \gamma}(t) dt \int_{|z-t|}^{z+t} C_{\alpha}(s) ds \]  \quad (3.43)

By taking advantage of the convolution equation theorem I can write 3.43 in a more compact form below.

\[ \rho_{\alpha}\hat{g}_{\alpha w}(k) = \sum_{\gamma} \hat{S}_{\alpha \gamma}(k)\hat{C}_{\alpha w}(k) \]  \quad (3.44)

Looking at Equation 3.42, the partial structure factors of the polymer blend in the bulk must first be obtained by solving the bulk PRISM Equations, 3.15, for this system specified with Equation 3.25 using the PY closure. This system of equations is solved using the Picard iteration [44] where the computational details are outlined in [68] (also see Section 3.1.4 as well as A.2). The solution scheme works as follows. First, the functions are discretized onto a grid with \( N_{\text{grid}} \) discrete points. The distance between the grid points will be constant and is defined as \( \Delta r \). For the Fourier Transforms, the grid will be defined in \( k \)-space with \( \Delta k = \pi / (N_{\text{grid}}\Delta r) \). A new function

\[ \gamma(r) = h(r) - C(r) \]  \quad (3.45)

is introduced that relates the two intermolecular functions together and is smooth.

Using the PY closure condition found in Equation 3.18 and the definition of \( \gamma(r) \) in Equation 3.45, \( C(r) \) is then calculated using an initial guess for \( \gamma(r) \). This initial guess is designated as \( \gamma^{\text{Old}}(r) \). \( \hat{C}(k) \) is then calculated from \( C(r) \) using the discrete sine FFT. Using the \( k \)-space definition of \( \gamma(r) \) found in Equation 3.23, gives a new estimation of \( \gamma(r) \) I will call \( \gamma^{\text{New}}(r) \). The new and old versions of \( \gamma(r) \) are now mixed with a mixing ratio of \( \alpha \) so that \( \gamma^{\text{mix}}(r) = (1 - \alpha)\gamma^{\text{Old}}(r) - \alpha\gamma^{\text{New}}(r) \) where \( 0.0 < \alpha < 1.0 \). For stability, values for \( \alpha \) are typically less than 0.10. The new estimates are then compared against the old estimates for convergence. When the average fractional difference (AFD, see equation 3.24) is smaller than some tolerance (in this work \( AFD < 10^{-7} \)) value the system is considered converged. If the system is not considered converged then \( \gamma^{\text{mix}}(r) \) is the new guess and the calculation starts again.
After the bulk correlation functions are calculated, the functions $C_{z=±∞}^{z}(z)$ and $C^{z}_{z=±∞}(z)$ can be calculated using the partial structure factor and equation 3.42. Once these values are found, a similar Picard iteration method can be used to solve the Wall-PRISM Equations. Using $\gamma_{aw}(z) = g_{aw}(z) + C_{op}(z)$, the wall PY closure relation found in Equation the new PY closure for the Wall-PRISM equations is found in Equation 3.47. Again an initial guess is made for $\gamma_{aw}(z)$. Using the new Wall-PRISM PY closure relation $C_{op}(z)$ is calculated. A new $\gamma_{aw}(r)$ can then be calculated using the following equation:

$$\hat{\gamma}^\text{New}(k) = (I - \rho \hat{\omega}(k) \hat{C}(k))^{-1} \hat{\omega}(k) \hat{C}_{op}(k) - \hat{C}_{op}(k)$$  (3.46)

The Picard iteration for Wall-PRISM can be found implemented in computer code in Appendix A.2.

$$PY \ Closure_{Wall-PRISM} = \begin{cases} C_{op}(z) = C^{z=±∞}_{aw}(z) & |z| < \frac{H}{2} \\ C_{op}(z) = -\gamma_{aw}(z) & |z| > \frac{H}{2} \end{cases}$$  (3.47)

Figure 3.3: Annulus site configuration as precursor for Wall-PRISM solution.

### 3.3 Self-Consistent PRISM

One of the most critical factors in the calculation of RISM/PRISM results is the intramolecular correlation function. As discussed previously, there are several analytical mod-
els that can be employed; however these can suffer from site overlap and incorrect polymer volumes. Also, in studies of polymer blends it has been shown that blending will alter the intramolecular correlations as polymer configurations change due to the presence of different polymer chain types [69].

3.3.1 Solvation potential

As mentioned in the SC-PRISM background section, a single molecule/polymer chain simulation is used to give realistic intramolecular correlations. For the single chain simulation a solvation potential is used to account for the intermolecular interactions with the other chains in the melt. However, there have been several proposed solvation potentials. These potentials can be derived from the closures in section 3.1.3. The first is an HNC-style potential,

$$\beta W_{HNC}(r) = -C(r) * S(r) * C(r)$$  \hspace{1cm} (3.48)$$

and is based on a Gaussian fluctuation potential [70–72]. This potential is typically more accurate for long range potentials with attractive and repulsive potentials such as Coulombic and Lennard-Jones potentials. In keeping with previous sections, $C(r)$ is the direct correla-
tion function and $S(r)$ is the inverse Fourier transform of the total structure factor defined in Equation 3.16.

The other two solvation potentials typically used are also based on the $C(r) * S(r) * C(r)$ approximation and are solved using the PY and MS closures. For short range repulsive behavior consistent with dense hard sphere systems, PY is typically used and is given by 3.49

$$\beta W_{PY}(r) = -\ln(1 + C(r) * S(r) * C(r))$$ \hspace{1cm} (3.49)

Finally the MS closure will give the MS-type solvation potential defined as

$$\beta W_{MS} = -\sqrt{1 + 2C(r) * S(r) * C(r)} + 1$$ \hspace{1cm} (3.50)

The MS solvation potential has been shown to provide more accurate approximations for hard-sphere diatomic liquids [63] and is of an intermediate strength between the HNC and PY potentials. The 3 solvation potentials are summarized in Table 3.1.

<table>
<thead>
<tr>
<th>Solvation Potential Model</th>
<th>Equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hypernetted Chain(HNC)</td>
<td>$\beta W_{HNC}(r) = -C(r) * S(r) * C(r)$</td>
</tr>
<tr>
<td>Percus-Yevick(PY)</td>
<td>$\beta W_{PY}(r) = -\ln(1 + C(r) * S(r) * C(r))$</td>
</tr>
<tr>
<td>Martynov-Sarkisov(MS)</td>
<td>$\beta W_{MS} = -\sqrt{1 + 2C(r) * S(r) * C(r)} + 1$</td>
</tr>
</tbody>
</table>

### 3.3.2 Self-Consistent PRISM method

The solvation potential definitions outlined in the previous section show that the correlation functions $\omega(r)$, $C(r)$ and $h(r)$ are required to calculate them. $\omega(r)$ will be determined from simulation and $C(r)$ and $h(r)$ will be determined using the RISM/PRISM Equations (3.15). However, to determine $C(r)$ and $h(r)$ from the RISM/PRISM Equations requires knowledge of $\omega(r)$. This interdependence of equations, sets up the equations to be solved self-consistently. The SC-PRISM scheme is illustrated in Figure 3.5 below. To solve this
problem, an initial approximation for the solvation potential, \( W(r) \), is made. For a new system, this guess is commonly taken to be zero. A single chain MC simulation will then be performed and this will give a resulting intramolecular correlation function, \( \omega(r) \). Using the RISM/PRISM equation as well as the PY closure, new values of \( C(r) \) and \( h(r) \) are found using a Picard iterative scheme (see Section 3.1.4). Using the resulting \( C(r) \), \( h(r) \) and \( \omega(r) \), a new solvation potential, \( W_{\alpha\gamma}^{\text{new}} \), of the desired type can be found. However, this new guess for the solvation potential is mixed with the prior guess for the solvation potential to improve convergence. Here, I define

\[
W_{\alpha\gamma}^{i+1}(r) = (1 - a)W_{\alpha\gamma}^{i}(r) + aW_{\alpha\gamma}^{\text{new}}(r),
\]

where \( i \) is the iteration index of the previous solution (or the input solution for the first step), and \( i + 1 \) is the iteration index for the next (mixed) solution. The \( a \) variable will tell how strongly or weakly to mix in the new solution and will be a value of \( 0 < a < 1 \). Typically this value will be fairly small, on the order of 0.05 or less.

Computation time can be reduced by incorporation of the Monte Carlo reweighting scheme [4, 69, 74]. The method allows previously generated chain configurations to be reused using the new solvation potential, \( W_{\alpha\gamma}^{i+1} \), in calculation of the new \( \omega(r) \). If there are \( J \) conformations generated to determine \( \omega_{\text{old}}(r) \) using the old solvation potential (defined as \( W_{\alpha\gamma}^{i}(r) \) above) then \( \omega_{\text{new}}(r) \) can be calculated by using the new solvation potential \( W(r)_{\text{New}} \) (defined as \( W_{\alpha\gamma}^{i+1}(r) \) above) with the old polymer confirmations using the equation

\[
\langle \omega_{\text{New}} \rangle \{ W_{\text{New}} \} = \frac{1}{Z} \sum_{j=1}^{J} \omega_{\text{Old}}^{j} \{ W_{\text{Old}} \} e^{-\beta|W_{\text{new}}(j) - W_{\text{Old}}(j)|} \tag{3.51}
\]

where

\[
Z = \sum_{j=1}^{J} e^{-\beta|W_{\text{new}}(j) - W_{\text{Old}}(j)|} \tag{3.52}
\]

The reweighting can be performed until the difference in the respective energy for the two solvation potentials is large enough to cause the following equation

\[
\min\{Z, 1/Z\} < \frac{J}{4} \tag{3.53}
\]

to fail. If it does fail the criterion, then a new set of polymer configurations is generated.
3.4 Two-Chain SC-PRISM

The method for the Two-Chain PRISM theory was developed by Wu and Li [7]. While One-Chain SC-PRISM has shown improved results due to the intramolecular correlation originating from a Monte Carlo simulation in a solvation potential, it has also been shown that One-Chain results at small distances did not agree with Molecular Dynamics studies, as shown for example in Figure 2.6. To improve on this, two chains will be sampled interacting together.

3.4.1 Two-Chain theory

I will still be working with the RISM/PRISM theories that I have outlined before, however, it will be beneficial to rewrite some of the equations from the perspective of simulated chain configurations. By deriving it from this perspective, I can then show how the simulation with a solvation potential is incorporated with the RISM/PRISM theory for both the One-Chain SC-PRISM as well as the Two-Chain theory. The work here is based on the two-chain DFT theory of Donley, Curro and McCoy that is abbreviated DCM [75]. This theory is a more computationally demanding theory, however, it is exact in the limit of two isolated chains. The DCM theory is based on the density functional theory of Chandler, McCoy and Singer (CMS) [76–78], which accounts for the medium-induced solvation potential. Let us again consider a case of a molecular liquid at equilibrium that consists of
$N$ identical polymer chains in a volume $V$ and at a temperature $T$. Each molecule/chain is again composed of $M$ spherical sites. I allow the sites to be of different types so there are $M_\alpha$ sites of type $\alpha$ and $M_\gamma$ sites of type $\gamma$ and so on. I define the $\alpha$ site density as $\rho_\alpha = M_\alpha N/V$ with the other site densities defined in the same manner. The chain density is $\rho_{\text{chain}} = N/V$. I specify each site’s spatial position on an arbitrary chain as follows. The 3-dimensional position is defined as $r_{ij\alpha}$, where $j$ is the number of the site of type $\alpha$ on the molecule/chain labeled $i$. For a system of many molecules, all such molecules/chains will have multiple configurations. I designate the set of all the configurations of molecules or polymers with the symbol $\mathcal{R}$, and so the set of configurations for the $i$th chain is $\mathcal{R}_i$. I define the total potential energy of the system of $N$ chains to be $V_N$. This potential energy can be broken down into the intramolecular energy contribution, $U(\mathcal{R}_i)$, from each individual molecule/chain and the intermolecular contribution, $u(\mathcal{R}_i, \mathcal{R}_q)$, between chains/molecules $i$ and $q$. To get the total potential energy I sum over all the molecules/chains and all the pairs of molecules/chains defined below as

$$V_N(\mathcal{R}_1, \mathcal{R}_2, ..., \mathcal{R}_N) = \sum_{i=1}^{N} U(\mathcal{R}_i) + \sum_{i=1}^{N-1} \sum_{q=i+1}^{N} u(\mathcal{R}_i, \mathcal{R}_q)$$

(3.54)

Note that all energy values here are given in units of $k_B T$.

The intrachain energy, $U(\mathcal{R}_i)$, consists of all the bending, stretching, and torsional energies of the molecule/chain. The interchain potential energy depends on the distances $r$ between sites. The two sums work such that every site type defined by $\alpha$ for molecule/chain $i$ and $\gamma$ for molecule/chain $q$ will be considered in pairs and the total interaction energy will be summed as

$$u(\mathcal{R}_i, \mathcal{R}_q) = \sum_{\alpha,\gamma} \sum_{j,f} M_\alpha M_\gamma u_{\alpha j\gamma f}(r)$$

(3.55)

The intramolecular structure factor is then defined below, in Equation 3.56, and it should be noted that $\delta(r)$ is the Dirac delta function in this context. In this case the chain labeled 1 is selected, but it could have been any of the molecules/chains.
The radial distribution function, $g_{\alpha\gamma}(r)$, can be written as

$$ g_{\alpha\gamma}(r) = \frac{V}{M_{\alpha}M_{\gamma}} \sum_{j,f} \langle \delta(r - r_{1\alpha j} + r_{1\gamma f}) \rangle $$

(Note $V$ here is the volume, not the potential energy $V_N$.)

These equations give the correlation functions in terms of the system potential energy. As I have shown, a portion of the total potential energy is the energy of interaction of the many molecules/chains with one another. However, for a two-molecule/chain simulation, I would like to average over all but two of the molecules/chains interactions. I therefore address the loss of the other molecules/chains by replacing them with an effective potential that will approximate the influence of the other molecules/chains. First, I discuss the more familiar one-chain SC-PRISM, then the new Two-Chain SC-PRISM as they are very similar.

The intramolecular correlation function defined in Equation 3.56 can be expressed in terms of a single molecule/chain:

$$ \Omega_{\alpha\gamma}(r) \equiv \rho_\alpha \omega_{\alpha\gamma}(r) \equiv \rho_\alpha \sum_{j,f} \left\langle \frac{1}{M_\alpha} \delta(r - r_{1\alpha j} + r_{1\gamma f}) \right\rangle $$

Now the brackets denote the average of a the probability distribution for a single molecule/chain conformation

$$ \langle \cdots \rangle_{R_{1}} \equiv \int dR_{1} P(R_{1}) \cdots $$

and the probability distribution for the single chain is

$$ P(R) \equiv \frac{e^{-V_{1}(R)}}{\int dR e^{-V_{1}(R)}} $$

However, now the total potential energy is replaced by a free energy potential.
\[ V_1(\mathcal{R}) = U(\mathcal{R}) + W(\mathcal{R}) \]  

(3.61)

where I define the total solvation potential component of the single chain interactions as

\[ W(\mathcal{R}) = \sum_{\alpha, \gamma}^{M_\alpha, M_\gamma} \sum_{j, f} W_{1\alpha j 1\gamma f}(|r_{1\alpha j} - r_{1\gamma f}|) \]  

(3.62)

(See Table 3.1 for the forms \( W(\mathcal{R}) \) used in this thesis.)

In this way, Equation 3.61 is similar to 3.54 with the second term in Equation 3.54 replaced by the solvation potential and the first term of 3.54 now a sum only over a single chain rather than the whole system of molecules/chains. The remainder of the One-Chain SC-PRISM calculation involves solving the RISM/PRISM equations as outlined before using the Picard iteration to obtain the functions needed to get the next solvation potential guess.

One benefit of doing the Two-Chain simulation is to also get the radial distribution function information as well as the intramolecular correlation information. I have already shown how to get the intramolecular correlation function by using a solvation potential rather than a many molecule/chain simulation. For finding the radial distribution function I will again invoke the DCM theory that makes use of the CMS density functional theory and allows the many molecule/chain average to be reduced to two molecules/chains. Here I define the radial distribution function, \( g_{\alpha \gamma}(r) \) below

\[ g_{\alpha \gamma}(r) = \left\langle \left\langle \frac{V}{M_\alpha M_\gamma} \sum_{j, f} \delta(r - r_{1\alpha j} + r_{2\gamma f}) e^{-V_2(\mathcal{R}_1, \mathcal{R}_2)} \right\rangle \right\rangle_{\mathcal{R}_1, \mathcal{R}_2} \]  

(3.63)

where now the effective interaction potential is between the two chains and defined in Equation 3.64 below.

\[ V_2(\mathcal{R}_1, \mathcal{R}_2) = \sum_{\alpha, \gamma}^{M_\alpha, M_\gamma} \sum_{j, f} u_{1\alpha j 2\gamma f}(|r_{1\alpha j} - r_{2\gamma f}|) + W_{1\alpha j 2\gamma f}(|r_{1\alpha j} - r_{2\gamma f}|) \]  

(3.64)

One thing to notice from the definition of \( g_{\alpha \gamma}(r) \) is the use of the double brackets. The double brackets refer to now doing an average over the sets of conformations for both the non-interacting chains, however, done with respect to their independent intramolecular prob-
ability distributions.

\[ \langle \cdots \rangle_{\mathbf{R}_1, \mathbf{R}_2} \equiv \int \int d\mathbf{R}_1 d\mathbf{R}_2 P(\mathbf{R}_1) P(\mathbf{R}_2) \cdots \]  

(3.65)

For convenience when doing the evaluation for \(g_{\alpha \gamma}(r)\) with simulation, I can invoke translational invariance and rewrite Equation 3.63 in the more helpful form below.

\[ g_{\alpha \gamma}(r) = \int d\mathbf{R}_1 d\mathbf{R}_2 P(\mathbf{R}_1) P(\mathbf{R}_2) \delta(\mathbf{r}_{111}) e^{-\frac{V^2}{M_\alpha M_\gamma} \sum_{j,f} \delta(\mathbf{r} - \mathbf{r}_{1\alpha j} + \mathbf{r}_{2\gamma f})} \]  

(3.66)

The reason for the delta function is to limit the integral to only configurations that have the first site on molecule/chain number 1, \(\mathbf{r}_{111}\), at the origin.

An additional benefit of doing the two-molecule/chain simulation is that the closure approximation can be replaced by Equation 3.63 and the respective solvation potential approximation from Table 3.1. Having the simulation itself give the closure relation potentially gives a more accurate representation of the behavior of the intermolecular correlation function. The equations needed for solving the two-molecule/chain approach are 3.58, 3.63 and the PRISM Equation 3.15. The solution scheme will be covered in the next section. Also, the Monte Carlo sampling methods for the two-chains will be covered.

### 3.4.2 General Two-Chain numerical scheme

In the single molecule/chain SC-PRISM method the MC simulation only produced the intramolecular correlation function. However, now with two chains the MC will also be used to generate the radial distribution function, \(g(r)\), as described in the previous section. Because the Two-Chain simulation is focused on improving results at smaller interaction distances, the two-molecule/chain sampling will only be done out to a limited range. This means that the long tail region for the \(g(r)\) must be accounted for. Because the one-chain theory gives good long range results, a single chain SC-PRISM calculation is done first. This will supply the tail portion to be pieced on to the simulation data. The general scheme for solving the Two-Molecule/Chain theory is again a Picard type iteration. To help guide the
reader, see Figure 3.6 for a visual representation of the numerical scheme. First, I will define a few new terms using the PRISM Equation:

\[ \gamma(r) \equiv H(r) - \Theta(r) \]  \hspace{1cm} (3.67)

\[ \Theta \equiv \Omega(r) * C(r) * \Omega(r) \]  \hspace{1cm} (3.68)

Figure 3.6: Two-Chain SC-PRISM theory solution scheme with convergence based on \( C(r) \).

An initial guess for \( \omega(r) \) and \( C(r) \) is made. The starting \( C(r) \) is labeled as \( C_{Initial}(r) \). The FFT is applied to these functions to give their k-space representations of \( \hat{\omega}(k) \) and \( \hat{C}(k) \). The PRISM equation can be rewritten to solve for \( \hat{H}(k) \) and that value will be designated as \( \hat{H}_{PRISM}(k) \). Using the inverse FFT to get \( H_{PRISM}(r) \), I can find the \( \Theta_{New}(r) \) using Equation 3.68 and \( \gamma_{New}(r) \) using Equation 3.67. After these values are found, the \( \hat{C}(k) \hat{S}(k) \hat{C}(k) \) term can be found. Reviewing table Table 3.1 shows all 3 solvation potential types rely on this term. After the appropriate solvation potential is found, it is put into the two-chain Monte Carlo simulation. There are two different sampling methods that can be used. For specifics on these two methods, see the next two sections. After the simulation is completed, the intramolecular correlation function, \( \Omega(r) \), and the total correlation function, \( H(r) \), can be found from the radial distribution function, \( g(r) \), generated by the simulation. The tail from the One-Molecule/Chain SC-PRISM is then patched on to account for the long range values of \( r \) not sampled. This allows the two-chain simulation to be more efficient and focus on the shorter range \( r \) values. Once \( \Omega(r) \) and \( H_{Sim}(r) \) are found, a new direct correlation function,
$C(r)$, can be found using the relations below

$$\hat{\Theta}_{\text{New}}(k) = \hat{\gamma}[H_{\text{Sim}}(r) - \gamma_{\text{New}}(r)] \quad (3.69)$$

$$\hat{C}_{\text{New}}(k) = \hat{\Omega}_{\text{New}}^{-1}(k)\hat{\Theta}_{\text{New}}(k)\hat{\Omega}_{\text{New}}^{-1}(k) \quad (3.70)$$

To improve convergence the $C_{\text{New}}(r)$ is mixed with $C_{\text{Initial}}(r)$ in a manner shown below in Equation 3.71 to give $C_{\text{Mix}}(r)$

$$C_{\text{Mix}}(r) = aC_{\text{New}}(r) + (1 - a)C_{\text{Initial}}(r) \quad (3.71)$$

where $a$ is known as a “mixing ratio”. It is defined to be $0 < a < 1$ and is typically small and on the order of $a = 0.3$ or smaller. This new $C_{\text{Mix}}(r)$ will then become $C_{\text{Initial}}(r)$ for the next iteration. Prior to this step, however, there is a convergence test to determine when the solution is converged. When the $C_{\text{New}}(r)$ is very close to $C_{\text{Initial}}(r)$ based on the error being defined as

$$\text{Error} = \left(\frac{1}{N_{\text{Grid}}} \sum_{j=1}^{N_{\text{Grid}}} (C_{\text{Initial}}(r_j) - C_{\text{New}}(r_j))^2\right)^{\frac{1}{2}} \quad (3.72)$$

the calculation is considered a converged solution. Here $N_{\text{Grid}}$ is the number of discrete points (in most of this work, it is 2048 points) and $r_j$ is the value of $r$ at the discrete values of index $j$. The criteria typically used is that $\text{Error} < 10^{-5}$ for a converged result.

### 3.5 Two-Chain Monte Carlo simulation

The Monte Carlo simulation for two chains can be used for small molecules or large polymer chains. For polymer chains and non-rigid molecules the standard pivot algorithm is used [79]. The standard pivot algorithm consists of a pivot step where by an angle, defined by 3 consecutive monomers in the chain, is picked at random and the two resulting arms will be bent around the central angle site by a set angle. The chain is also twisted in a torsional move around a random dihedral made from 4 consecutive monomer units. For branched systems pivot steps can also take place on improper dihedral groups. The arms from the bond angle are bent by $\pm 20^\circ$ around an axis normal to the plane made by the 3 monomer
sites in the angle. The torsional move consists of doing a twisting move of ±180° around the axis made by the two middle dihedral sites. After each pivot move is made, the bonded and non-bonded energies are calculated and the acceptance of the move will be based on a Metropolis criteria. Previous work looked at the auto-correlation function of the pivot moves and found a correlation time of 400 pivot steps[4], therefore, every 400 pivot steps the configuration of the chains will be saved. From these saved configurations the intramolecular correlation function \( \Omega(r) \) can be found. The evaluation of the radial distribution function \( g(r) \) will require non-Boltzmann sampling methods to keep the chains from drifting apart. The two separate sampling methods that have been utilized are the direct and the window sampling methods. Both of these methods of sampling used in the two-chain MC simulation are explained in the next two sections.

### 3.5.1 Direct sampling

For the direct sampling method, the radial distribution function will be calculated by using the definition for \( g_{\alpha\gamma}(r) \) found in quation 3.66. The direct sampling set up of the two chains is shown in Figure 3.7. Here, I average over configurations of the two chains from the given probability distributions that are defined to be for non-interacting chains that are held at a fixed distance apart. Each of these chains will be from a single-chain distribution generated using the pivot algorithm method described above. The direct sampling is based on the method developed by Yethiraj et al. [80].

The method of direct sampling works as follows. A set of single chain configurations is generated using the solvation potential and the standard pivot algorithm. From this set of configurations, the intramolecular correlation function can be calculated via Equation 3.58. The two-molecule/chain configurations from this set are chosen at random. Subsequently two random sites on the two-molecules/chains are also chosen at random as well. Molecule/Chain number one will have its random site placed at the origin. Molecule/Chain number two will have its random site also placed at the origin. Molecule/Chain two will then be translated by a discrete distance \( r \) that is pointing in a random direction and orientation from the origin.
This will be repeated for all discrete values of $r$ of interest (recall I only actually sample to a certain distance and then piece in the One-Chain SC-PRISM results for large $r$.)

I can improve the statistics by collecting information on all the pairs that are interacting in the two-chain simulation. Recall, that I defined $g_{\alpha\gamma}(r)$ specifically assuming that the chains are non-interacting and so samples with a uniform probability to all scalar distances $r = |r_{1\alpha j} - r_{2\gamma f}|$, which will introduce a sampling bias that is proportional to $1/r^2$. In order to sample the other pairs, I need to include a normalization correction for the bias. For the two-chain configuration where $r = |r_{1\alpha j} - r_{2\gamma f}|$ (this is defined above by the two random sites on the chains), I can then calculate the other distances and label them as $r' = |r_{1kd} - r_{2sy}|$ where $r'$ is the distance between the $d$th and $y$th sites of type $k$ and $s$ on molecule/chain 1 and 2 respectively. See Figure 3.7 for examples of $r'$. Therefore, I can calculate the radial distribution function for $g_{ks}(r')$ by multiplying every configuration that will be binned for when accumulating the average Boltzmann factor ($e^{-V_2(R_1R_2)}$) by a value of $r^2/r'^2$. This allows for the two-molecule/chain configuration to contribute data for all the interacting pairs rather than just those defined by $r = |r_{1\alpha j} - r_{2\gamma f}|$. This does two things. First, it allows for more efficient simulation as new two-molecule/chain pairs don’t need to be generated for every pair sampling. Second, it reduces the noise of the simulation as there will be $M^2$ data points rather than a single one and so the noise for the radial distribution function will be reduced by a factor of $1/\sqrt{M_\alpha M_\gamma}$. For doing the actual sampling here are the equations for calculating $g_{\alpha\gamma}(r)$:

$$g_{ks}(r') = \frac{r^2}{r'^2} e^{-V_2(R_1R_2)}$$

(3.73)

using the definitions of $r$ and $r'$ above. The radial distribution function is then

$$g_{\alpha\gamma}(r) = \left\langle \frac{1}{M_\alpha M_\gamma} \sum_{k,s} g_{ks}(r) \right\rangle$$

where the $\langle \ldots \rangle$ indicate the average over all two-molecule/chain configurations generated for respective $r$ distances.
3.5.2 Window sampling

While direct sampling has the advantage of giving $g(r)$ from an average over the sets of two-chain configurations it doesn’t incorporate importance sampling in generation of the configurations. This can result in problems in the sampling statistics as molecules/chains can spend time in unlikely configurations. This inefficiency can be especially true for large polymer chains. This can be addressed by employing a window sampling method [81].

For generation of the two-molecule/chain configurations a Metropolis sampling technique is employed where the configurations are proportional to their Boltzmann weight. One chain will be put at the origin to start, however, it is not confined to stay there and the second chain will be free to move. However, the middle site of the molecule/chain is restricted from leaving its windowed boundaries. As such, I will define the distance between the middle site of chain 1 at the origin and the middle site of chain 2 to be $r_m$. The size of the window is chosen such that the free energy change inside the window will not be larger than $k_BT$.

This value for the size of the windows must be determined by trial and error. Consecutive
windows will overlap by half their widths. See Figure 3.8 for an example of the use of overlapping windows.

![Figure 3.8: Window sampling method of two chains.](image)

The overlap of the windows is used to improve stitching the separate window probability functions together to make a smooth $g(r)$. The windows are set up such that there is a starting minimum distance for window 1 from the origin. This is the distance labeled $r_0$. There will be $n$ windows all of $r_{\text{width}}$. Recall that $r_{\text{width}}$ is determined by ensuring that the free energy change inside of the windows is no larger than $k_B T$. Each simulation of two chains will consist of one standard pivot move for each molecule/chain. See Section 3.5 for pivot move details. Finally, a displacement move and a rotation move is performed on chain number two. Any move, including pivot moves, that causes $r_m$ to step out of its respective window will be rejected even if it is energetically favorable. Every 400 MC steps the two-chain configurations will be saved and the pairs will be binned and then an average is taken.
over the total $N$ saved configurations.

$$P_{\alpha\gamma}(r) = \frac{1}{N} \sum_{i}^{N} \sum_{j,f} \frac{M_{\alpha} M_{\gamma}}{4\pi r^2} \delta(r - |r_{1\alpha j} + r_{2\gamma f}|) \quad (3.74)$$

Each window will generate its own probability distribution using the Equation 3.74 above. These now need to be stitched together to make a continuous function over all the windows. This is done by using a multiplicative factor in a least squares method to fit the overlapping portions of adjacent windows together. The windows are distributed such that I can find window interval boundaries for the $i$th window as $[r_0 + \frac{1}{2}(i - 1)r_{\text{width}}, r_0 + \frac{1}{2}(i + 1)r_{\text{width}}]$. Since adjacent windows overlap I will then split our probability distributions into 2 different functions depending on which half I am in. So the probability distribution for the left half of the $i$th window is $P_{\alpha\gamma}^{(i,l)}(r)$ for the configurations where $r_m \in [r_0 + \frac{1}{2}i r_{\text{width}}, r_0 + \frac{1}{2}(i - 1)r_{\text{width}}]$ and for the right side $P_{\alpha\gamma}^{(i,r)}(r)$ is defined where $r_m \in [r_0 + \frac{1}{2}i r_{\text{width}}, r_0 + \frac{1}{2}(i + 1)r_{\text{width}}]$. The first 3 windows are shown in Figure 3.9 for clarity. This means that I can define where the two probability distributions overlap as $P_{\alpha\gamma}^{(i,l)}(r)$ and $P_{\alpha\gamma}^{(i,r)}(r)$. These two portions will differ by a weighting factor, $A_{\alpha\gamma}^{(i-1)}$, such that $p_{\alpha\gamma}^{(i-1),r}(r) \approx A_{\alpha\gamma}^{(i-1)} P_{\alpha\gamma}^{(i,r)}(r)$. As said before this weighting factor is determined via a least square fit defined below in Equation 3.75.

$$A_{\alpha\gamma}^{(i-1)} = \frac{\int_{r_0 + \frac{1}{2}(i-1)r_{\text{width}}}^{r_0 + \frac{1}{2}i r_{\text{width}}} dr P_{\alpha\gamma}^{(i,l)}(r) P_{\alpha\gamma}^{(i-1,r)}(r)}{\int_{r_0 + \frac{1}{2}(i-1)r_{\text{width}}}^{r_0 + \frac{1}{2}i r_{\text{width}}} dr P_{\alpha\gamma}^{(i,l)}(r) P_{\alpha\gamma}^{(i,i)}(r)} \quad (3.75)$$

Since each window is fit successively there will be a net weighting factor from all the weighting terms such as $A_{\alpha\gamma}^{(1)} A_{\alpha\gamma}^{(2)} \cdots A_{\alpha\gamma}^{(n-1)}$. The probability distribution for the $i$th interval, $P_{\alpha\gamma}^{(i)}$, can therefore be taken as an average of the two contributions from each of the overlapping windows.

$$p_{\alpha\gamma}^{(i)} = (\prod_{j=1}^{i-2} A_{\alpha\gamma}^{(j)}[B_{\alpha\gamma}^{(j)}(r) P_{\alpha\gamma}^{(i-1,r)}(r) + (1 - B_{\alpha\gamma}^{(j)}(r)) A_{\alpha\gamma}^{(i-1)} P_{\alpha\gamma}^{(i,l)}(r)])$$

where the factor $B_{\alpha\gamma}^{(i)}$ is defined as

$$B_{\alpha\gamma}^{(i)} = \frac{(A_{\alpha\gamma}^{(i-1)})^2 P_{\alpha\gamma}^{(i,l)}(r)}{P_{\alpha\gamma}^{(i-1,r)}(r) + (A_{\alpha\gamma}^{(i-1)})^2 P_{\alpha\gamma}^{(i,l)}(r)}$$
Figure 3.9: Window overlap configuration and designation of Left and Right window halves.

An example of PE with M=24 (24 monomer units) is shown in Figure 3.10 The distributions for each window display long tails beyond the boundary of the respective windows on the order of the chain dimensions. Therefore, the distribution function will still have a contribution in the range of interest beyond what is being simulated explicitly. This means the joined distribution results, $p_{o\alpha\gamma}$, will have a continuous cutoff at large $r$ values. For large molecule/chain separation, instead of piecing on the tail from the one-chain simulation I instead use the last window results to approximate the probability distribution further out. The distributions in windows further out will just be the final simulated window shifted by $\frac{1}{2}r_{width}$ and so I can use the final window to approximate the distribution functions and make up the missing data:

$$P_{o\alpha\gamma}(r) = P_{o\alpha\gamma}^{\prime}(r) + \sum_{i=1}^{n'} P_{o\alpha\gamma}^{(n)}(r - i \times \frac{1}{2}r_{width}) \quad (3.76)$$

where $P_{o\alpha\gamma}^{(n)}(r)$ is the final window distribution function (for Figure 3.10 $n = 14$ so $n' = 15$ in this case). Then the entire probability function, $P_{o\alpha\gamma}(r)$, over the range of [0, $R$], can be related to the radial distribution function by a final multiplicative factor $A$ such that

$$g_{o\alpha\gamma}(r) = AP_{o\alpha\gamma}(r)$$

Here $A$ can be determined again using the least squares fit but in a range where the tail of both $P_{o\alpha\gamma}(r)$ and $g_{o\alpha\gamma}(r)$ are flat. $g_{o\alpha\gamma}(r)$ is supplied again from the one-chain SC-
PRISM results. The intramolecular correlation function is also extracted from the saved two-molecule/chain configurations. However, because I now have interacting chains there is a bias introduced. The single chain structure is then found by averaging the two-chain configuration and reweighting factor for the respective two-chain configuration, according to

\[
\omega_{\alpha\gamma}(r) = \frac{1}{4\pi r^2} \sum_{i=1,2}^{M_{\alpha},M_{\gamma}} \sum_{j,f} e^{V_2(R_1,R_2)} \delta(r - |r_{ij\alpha} - r_{j\gamma f}|)
\]

Figure 3.10: Radial distribution function reconstructed from window probability distribution functions for a polyethylene chain modeled as a chain of 24 sites \((M = 24)\) interacting with a repulsive LJ potential; \(r_0 = r_{\text{width}} = 3\text{Å}\). Top: figure shows the \(g(r)\) results from the joined window samples (solid line) and a direct sampling method (circles) Bottom: are the probability distributions from the even numbered windows \(i = 2, 4, 6...14\) and go from left to right[7].
CHAPTER 4
SURFACE SEGREGATION OF LINEAR/CYCLIC BLENDS

This chapter will focus on surface segregation of lower molecular weight linear/cyclic polymer blends. As discussed in Chapter 2, surface segregation of polymer blends can be driven by differences in polymer molecular architecture. Previous work had dealt with blends of linear and branched polymers with various architectures. Linear response theory predicts that segregation behavior can be understood in terms of surface potentials that include both enthalpic as well as entropic factors [1]; however, NR measurements on pom-pom molecules showed that these potentials alone weren’t sufficient to explain the observed segregation behavior [37]. SCFT calculations by Hu [2] were performed to explain the results of the NR studies, focusing on the influence of the chain length between branch points. The SCFT results showed agreement with the experiments (see Figure 2.5). The success of this work motivated further work with linear/cyclic blends.

While the surface enrichment in blends of linear and branched polymers could be largely, though not completely, explained in terms of effective potentials for end and branch points, the driving force for segregation of cyclic polymers is interesting because it lacks end groups. Instead, linear response theory showed that because linear chains have a larger configurational entropy penalty as compared with cyclic chains, the cyclic chains are enriched at the surface by a factor of 2 compared to the bulk composition. SCFT calculations on linear/cyclic blends by Hu were in agreement in finding cyclics enhanced by a factor of 1.8 for all molecular weights [2]. However, while NR results confirmed a surface preference for cyclics at high MW, at low MW linear chains were actually preferred at the surface [38]. A likely reason that SCFT fails to capture the preference for linear chains at low MW is the larger relative importance of packing effects for small chains.
Wall-PRISM is an atomistic theory and can capture the packing effects of the chains at the surface that SCFT cannot. However, models for the polymer chain types and the corresponding intramolecular correlation functions \( \omega_{\alpha\gamma}(r) \) are required. While the intramolecular correlation function for a non-overlapping freely jointed chain model had been developed for linear chains \([3]\), the corresponding model for cyclic chains had not been developed. The following is my work developing the cyclic model and evaluating the needed intramolecular correlation function. Using this model, I present below my Wall-PRISM calculation results for comparison with NR experiments, focusing on a blend of 2 kD linear and cyclic polystyrene chains that shows a surface preference for the linear component. The first few sections describe the polymer models used for the blend.

4.1 Non-overlapping freely jointed chain polymer model

The model for the polymer chain used for the linear and cyclic blend is the non-overlapping freely jointed chain (NFJC) \([3]\). This model treats the polymer chain as an ideal freely jointed chain (IFJC) in the sense that successive bonds, each with constant and equal bond length, have orientations that are statistically uncorrelated, i.e. “freely jointed”. However, unlike the ideal freely jointed chain, overlap of the two beads under consideration for the intramolecular correlation function are explicitly forbidden. For long chains, the NFJC has been shown to be inaccurate when compared with MD simulations \([82]\), however, for shorter chains the results are more reasonable because the intramolecular correlations are short ranged. The NFJC model also has the virtue that no prior chain conformation information is required. The NJFC model was successfully used in the original Wall-PRISM paper \([39]\) for small 4-mer and 8-mer linear polymers.

4.1.1 Linear chain - new derivation of NFJC intramolecular correlation functions

The 2 kD polystyrene (PS) chains consist of approximately 20 styrene monomer units, for both linear and cyclic chains. The PS monomer sites are modeled as coarse-grained
tangential spheres.

I first attempted to use directly the model given for the linear Non-overlapping Freely Jointed Chain by Curro and Schweizer Schweizer and Curro [3], Yethiraj and Hall [39], Schweizer and Curro [83]. The model assigns the intramolecular distribution functions $\omega_{\alpha\gamma}(r)$ to have their ideal uncorrelated values for distances beyond the hard sphere diameter for all non-bonded sites $\alpha$ and $\gamma$. Within the hard sphere diameter the probability of overlapping configurations will be zero. Thus

$$\omega_{\alpha\gamma}(r) = \begin{cases} 0 & r < \sigma \\ \frac{1}{(2\pi)^3} B_{\alpha\gamma} \int d\mathbf{k} e^{i\mathbf{k}\cdot\mathbf{r}} \left( \frac{\sin(k\sigma)}{k\sigma} \right)^{|\alpha-\gamma|} & r \geq \sigma \end{cases}$$

for $|\alpha-\gamma| \geq 2$, where

$$B_{\alpha\gamma} \equiv (1 - \hat{J}_{\alpha\gamma}(0))^{-1}$$

is a normalization factor such that $\hat{\omega}_{0\gamma}(0) \equiv 1$, and where

$$\hat{J}_{\alpha\gamma}(K) = \frac{2}{\pi K} \int_0^1 dR \sin(KR) \int_0^\infty dK' \sin(K'R) \left( \frac{\sin(K')}{K'} \right)^{|\alpha-\gamma|}.$$ 

Here I use the dimensionless variables of $R = r/\sigma$ and $K = k\sigma$. The dependence on the indices $\alpha$ and $\gamma$ reduces to one just on the number of links $\tau \equiv |\alpha-\gamma|$ between the two sites, and so we use the notation $\hat{\omega}_{\tau}(K)$ below to refer to $\hat{\omega}_{\alpha\gamma}(K)$ for $\tau \equiv |\alpha-\gamma|$. See Figure 4.1 for an illustration of the definition of $\tau$. Note that the $\tau = 0$ and $\tau = 1$ cases can be evaluated analytically in $k$ space to give

$$\hat{\omega}_0(k) = 1 \quad (4.1)$$

and

$$\hat{\omega}_1(k) = \frac{\sin(kL)}{kL} \quad (4.2)$$

where $L$ is the bond length between successive monomers and $L = \sigma$ for tangential spheres. The intramolecular structure factor is then
\[ \hat{\omega}(K) = N^{-1} \sum_{\alpha,\gamma=1}^{N} \hat{\omega}_{\alpha\gamma}(K) \]

\[ = 1 + 2N^{-1} \sum_{\tau=1}^{N-1} (N - \tau)\hat{\omega}_{\tau}(K) \] (4.3)

which can be rewritten as

\[ \hat{\omega}(K) = \hat{\omega}^{IFJC}(K) + \frac{2}{N} \sum_{\tau=2}^{N-1} (N - \tau)(\hat{\omega}_{\tau}(K) - \left( \frac{\sin(K)}{K} \right)^{\tau}) \] (4.4)

where \( \hat{\omega}^{IFJC}(K) \) is the Fourier transform of the intramolecular structure factor for the ideal freely jointed chain model (IFJC), given in Equations 4.5 and 4.6 below. \( N \) is the number of monomer sites on the chain. Thus we have

\[ \hat{\omega}^{IFJC}(k) = (1 - f)^{-2}(1 - f^2 - 2N^{-1}f + 2N^{-1}f^{N+1}) \] (4.5)

\[ f = \frac{\sin(kL)}{kL} \] (4.6)

with \( L \) being the bond length between monomers and \( N \) being the number of monomers in the chain. Recall that the only difference between the NFJC and the IFJC is that intramolecular overlaps are forbidden in the NFJC, but all other aspects of the model are the same. Therefore the second term on the right-hand side (RHS) of Equation 4.4 involves enforcement of the overlap restriction in \( r \) space for \( r < \sigma \). This second term in Equation 4.4 can be evaluated using

\[ \hat{\omega}_{\tau}(K) \equiv B_{\tau}(\left( \frac{\sin(K)}{K} \right)^{\tau} - \hat{J}_{\tau}(K)) \] (4.7)

where

\[ B_{\tau} \equiv (1 - \hat{J}_{\tau}(0))^{-1} \] (4.8)

and

\[ \hat{J}_{\tau}(K) = \frac{1}{\pi K} \int_{0}^{\tau} dy y \left( \frac{\sin(y)}{y} \right)^{\tau} \left( \frac{\sin(K - y)}{K - y} - \frac{\sin(K + y)}{K + y} \right) \] (4.9)

with the zero wave vector limit of the latter term given by
\[ \hat{J}_\tau(0) = \frac{2}{\pi} \int_0^\infty dy \left( \frac{\sin(y)}{y} \right)^\tau \left( \frac{\sin(y)}{y} - \cos(y) \right) \] (4.10)

Figure 4.1: Example of an \( N = 10 \) linear tangential monomer chain where the numbers indicate the site indices for \( \alpha \) and \( \gamma \). Examples are shown for the definition \( \tau \equiv |\alpha - \gamma| \) with respect to the site indices. Note that the actual conformation of the chain does not have to be a straight line, but is shown this way to simplify visualization of \( \tau \).

Equations 4.9 and 4.10 are evaluated numerically. Calculations for the NFJC \( \hat{\omega}(k) \) for short chains of 20 and 2000 monomers were obtained using this method and compared with published results from [84]. However, when the above method was altered for cyclic topology calculation of the NFJC \( \hat{\omega}(k) \) the resulting Wall-PRISM calculations could not be converged due to numerical instability.

Accordingly, I tried a new method for calculating the intramolecular correlation functions, described below, solving the freely jointed chain intramolecular correlation function in \( r \) space rather than \( k \) space. Then the non-overlap will be enforced directly in \( r \) space. The intramolecular correlation functions obey the recursive relationship \( \omega_{\tau+1}(r) = \int dr' \omega_{\tau}(r')\omega_1(r - r') \) since a chain of \( \tau + 1 \) sites is formed of a chain of \( \tau \) sites connected by a single bond to another single site. This equation can be solved by induction to yield the following analytical real-space expression for the intramolecular correlation function for \( \tau \geq 2 \). (For \( \tau = 0 \) and \( \tau = 1 \), I use the analytic k-space expressions given above.)
\[ \omega_{\tau}^{IFJC}(r) = \frac{1}{2^{r+1} \pi (\tau - 2)!} \sum_{s=0}^{\tau-2} (-1)^s \binom{\tau - 2}{s} r^{s-1} \]
\[ \times \sum_{m=0}^{F((\tau - 1)/2)} (-1)^m \binom{\tau}{m} H(\tau - 2m - r) (\tau - 2m)^{\tau-2-s} \]
\[ \times \sum_{m=0}^{F((\tau - 1)/2)} (-1)^m \binom{\tau}{m} H(\tau - 2m - r) (\tau - 2m)^{\tau-2-s} \]

where the function \( F(x) \) is the (integer) floor function and \( H(x) \) is the Heaviside (unit step) function. The explicit expressions for \( \omega_{\tau}^{IFJC}(r) \) can be calculated once and saved as input for further calculation. The non-overlapping (NO) condition is now strictly enforced in r-space by defining the \( \omega_{\tau}^{NFJC}(r) \) to be zero inside the site diameter in Equation 4.12 using the ideal function. (For the ideal freely jointed chain, this step is skipped.)

\[ \omega_{\tau}^{NFJC}(r) = \begin{cases} 0 & r < \sigma \\ \omega_{\tau}^{IFJC}(r) & r > \sigma \end{cases} \] (4.12)

The 3d Fourier sine transform is then performed to give the \( k \)-space representation for values of \( \tau \) from 2 to \( N-1 \). The intramolecular correlation functions \( \hat{\omega}(k) \) for either the ideal freely jointed chain or non-overlapping freely jointed chain can then finally be calculated using

\[ \hat{\omega}_{Linear}(k) = \frac{1}{N} \sum_{\tau=2}^{N-1} 2(N - \tau) \hat{\omega}_\tau(k) \frac{\hat{\omega}_\tau(0)}{\hat{\omega}_\tau(0)} + N\hat{\omega}_0(k) + 2(N - 1)\hat{\omega}_1(k) \] (4.13)

where the \( \hat{\omega}_\tau(k) \) is either \( \hat{\omega}_\tau^{IFJC}(k) \) or \( \hat{\omega}_\tau^{NFJC}(k) \) depending on the desired model.

Here I compare \( \hat{\omega}(k) \) for both the ideal freely jointed chain and the non-overlapping freely jointed chain for \( N = 20 \) using the standard Kratky plot in Figure 4.2. The plots are identical at very small \( k \) but then differentiate much more for \( 1 < k\sigma < 5 \). It is in that range that the ideal case plateaus as is expected for a random coil while the non-overlapping case does not. For \( k\sigma > 5 \), the curves for the ideal and non-overlapping cases become parallel and any difference is caused by the enforcement of the non-overlapping sites.

The mean squared radius of gyration for each of the two chains can be calculated to be
Figure 4.2: Kratky plot of the Ideal Freely Jointed Chain and the Non-Overlapping Freely Jointed Chain for a $N = 20$ linear chain.
\[ \langle R^2_g \rangle_{\text{ideal}} = \frac{(N + 1)(N - 1)\sigma^2}{6N} \]  
(4.14)

for the ideal case, and

\[ \langle R^2_g \rangle_{\text{NO}} = \frac{2\pi}{2N} \int_0^\infty drr^4\omega(r) \]  
(4.15)

for the non-overlapping case Yamakawa [85], which can also be written in k space as Schweizer and Curro [3]

\[ \langle R^2_g \rangle_{\text{NO}} = -\frac{3\sigma^3}{2N} \lim_{K \to 0} \frac{d^2\hat{\omega}(K)}{dK^2} \]  
(4.16)

Using these expressions, the radius of gyration for the \( N = 20 \) linear polystyrene chains studied here are \( R^2_g^{\text{ideal}} = 1.82\sigma \) assuming an ideal freely jointed chain and \( R^2_g^{\text{NO}} = 1.89\sigma \) assuming a non-overlapping freely chain. While they are close, the non-overlapping form is slightly larger, which is to be expected from the chain expansion due to enforcement of the non-overlapping condition.

### 4.1.2 Cyclic chain - derivation of new model

As mentioned above, to apply Wall-PRISM theory to understand surface segregation in blends of small linear and cyclic chains, the intramolecular correlation function is needed. Here, I develop the cyclic model as an extension of the previous work on the linear non-overlapping freely jointed chain discussed in the previous section. For cyclic chains, there are no end groups, and so the intramolecular correlation between any two sites is determined by the statistics of the two chain paths connecting the two sites, one of length \( \tau \), and the other of length \( N-\tau \). Figure 4.3 illustrates an example of the two paths for the links between sites \( \alpha = 1 \) and \( \gamma = 10 \).

To account for both the long and short paths described above, the statistical weight of the two sites \( \alpha \) and \( \gamma \) being at particular positions is given by the product of the statistical weights of the long and short paths alone. The statistical weight of one of the paths is given as that of a linear chain connecting the two sites. Thus we have
Figure 4.3: Example of an $N = 10$ cyclic tangent monomer chain where the numbers indicate the site indices $\alpha$ and $\gamma$. Examples are shown for the definition of $\tau$ with respect to the site identities. This results show the two different paths of length $\tau = 1$ and $\tau = 9$ for the case of $\alpha = 1$ and $\gamma = 10$.

\[ \omega^\text{Ring}_\tau(r) = \omega_\tau(r)\omega^-N-\tau(r) \]  
where $\omega_\tau(r)$ is the function defined for the corresponding model for linear chains given above.

For instance, for a cyclic ideal freely jointed chain, the expression for $\omega^\text{Ideal}_\tau(r)$ from Equation 4.11 above can be used. This is again calculated for all $\tau$ except for $\tau = 0$ and $\tau = 1$, which are trivial and exactly equal to the linear case. Likewise, for a cyclic non-overlapping freely jointed chain, the non-overlapping of sites is enforced for $\omega^\text{Ring}_\tau(r)$ for $2 < \tau \leq N$ as shown in Equation 4.12. As for linears, this step is omitted for the ideal FJC case. The 3D sine Fourier transform is then evaluated for each function $\omega^\text{Ring}_\tau(r)$, and the resulting $k$ space versions are combined in the equation below to give the final result for the intramolecular correlation function for a cyclic chain

\[ \hat{\omega}_\text{Ring}(k) = \frac{1}{N} \sum_{\tau=2}^{N-2} \hat{\omega}_\tau(k) + N\hat{\omega}_0(k) + 2N\hat{\omega}_1(k) \]  

Notice that in the ring/cyclic formulation the summation goes from $\tau = 2$ to $\tau = N - 2$ rather than $\tau = N - 1$. This is because in the ring/cyclic topology, the last site numbered as $N$ is necessarily attached to the first site. This will therefore be equivalent to it being a second $\delta$ function as in the $\tau = 1$, resulting in a second contribution of $N\hat{\omega}_1(k)$ to Equation
4.18. This approach generated intramolecular correlation functions for cyclic chains that resulted in stable and convergent Wall-PRISM calculations for blends. Again, I plot a comparison between the two intramolecular functions below in Figure 4.4. In the case of cyclic chains, as for linear chains, at small $k\sigma$ there is only small differences between the non-overlapping and ideal cases. However at $1 < k\sigma < 5$ there is an even more dramatic deviation than was seen for linear chains. For $k\sigma > 5$, again the two models give curves which become parallel to each other.

![Figure 4.4: Kratky plot of Ideal Freely Jointed Chain and the Non-Overlapping Freely Jointed Chain for a $N = 20$ cyclic chain.](image)

The intramolecular structure factors for the linear and cyclic chains are compared in Figure 4.5, with panel A depicting the ideal chains and panel B depicting the non-overlapping chains. For the ideal case, the linear and ring chains agree for $k\sigma < 0.5$ as well as for $k\sigma > 3.5$. Between 0.5 and 3.5, there is a deviation where the ring has more of a peaked portion near $k\sigma = 1.2$ while the linear chain exhibits a plateau there. For the non-overlapping chains
shown in panel B, the linear and cyclic chains are similar for $k\sigma < 0.5$. The tail regions for $k > 5.5$ have similar slopes but do not match up completely. There is a greater difference in the middle $k$ value region for the non-overlapping versions of the chains compared with the ideal case.

Figure 4.5: Kratky plots comparing the linear and cyclic chain intramolecular structure factors for A) Ideal FJC and B) Non-overlapping FJC

The non-overlapping freely jointed chain models, which capture steric packing effects in the intramolecular correlation function, will be used as the model for the following sections for the surface segregation calculations using Wall-PRISM. As mentioned previously, the non-overlapping model does not do well for large polymers, but does show good results for the smaller molecular weight system being tested.

4.2 Wall-PRISM results

With the chain models and intramolecular correlation functions in hand, I carried out Wall-PRISM calculations for a blend of linear and cyclic polymers with parameters designed to match those studied by NR. The blend is composed of 80% linear and 20% cyclic PS chains of 20 monomer units each. As discussed in chapter 3, prior to doing the Wall-PRISM surface calculations, the bulk PRISM Equations (3.25-3.27 or 3.15) must first be solved. The PRISM equations are solved using a general Picard iteration scheme described
in section 3.1.4, with the PY closure (see Equation 3.18). The packing fraction for the polymer chains is \( \eta = 0.45 \), which gives a site density of 0.0344 \( \sigma^{-3} \) for the linear chains and 0.0086 \( \sigma^{-3} \) for the cyclic chains. Unless otherwise specified, lengths will be reported in units of \( \sigma \) for the PS monomer site. Thus the two chains monomer diameters are defined as \( \sigma_{\text{linear}} = \sigma_{\text{cyclic}} = 1 \). The numerical solution was evaluated on a grid of \( N_{\text{grid}} = 2048 \) evenly spaced points with a spacing of \( \Delta r = 0.02 \) and a spacing of \( \Delta k = \pi/\Delta r/N_{\text{grid}} \) for the k-space FFT representation. The Picard iteration was run with with a 5% mixing ratio, and run until a convergence tolerance of \( AFD < 10^{-7} \) was achieved (see Equation 3.24 for AFD definition). The results for the bulk intermolecular pair distribution function \( g(r) \) are shown below in Figure 4.6. The packing of cyclic molecules next to other cyclics shows the strongest structure, exhibiting sharper peaks in \( g(r) \), perhaps reflecting their more compact and rigid shape.

![Figure 4.6: Bulk pair distribution function from PRISM for a binary blend of 80/20 Linear/Cyclic Polystyrene with \( \eta = 0.45 \).](image-url)

Figure 4.6: Bulk pair distribution function from PRISM for a binary blend of 80/20 Linear/Cyclic Polystyrene with \( \eta = 0.45 \).
To solve the Wall-PRISM equations, first the respective asymptotic direct correlation functions are calculated using Equation 3.42. This is coupled with Equation 3.44 and the PY closure for Wall-PRISM Equation, 3.39. This is again solved using Picard iteration. All other parameters are the same as those used for the bulk PRISM solution above, including the mixing ratio and the AFD tolerance. The $g(r)$ results for the Wall-PRISM calculations are shown in Figure 4.7.

![Graph showing $g(r)$ results for Wall-PRISM calculations.](image-url)

**Figure 4.7:** Wall-PRISM monomer-wall pair distribution function for a binary blend of 80/20 Linear/Cyclic Polystyrene. The origin in this plot indicates the start of the wall, with $r$ indicating the distance from the wall into the bulk.

The Wall-PRISM results show both linear and cyclic chains enriched directly at the surface when compared with the bulk. Most significantly, the linear polymers are preferentially enriched over cyclic polymers at the surface, in agreement with experiments done by the Foster group [38]. There is also indication of the packing of the polymers at the surface by the pronounced multiple peaks in the $g(r)$ plot. From the radial distribution plots for a range of densities, Figure 4.8 shows the emergence of increasing packing as the system moves from a dilute solution to a dense melt.
Figure 4.8: Comparison of the same 80/20 Linear/Cyclic Polystyrene at 4 different densities. The densities are given at the top of each plot.
While linear chains are always preferentially enriched directly at the surface \((r = 0)\) moving away from the surface, there are regions where cyclic chains are enriched. For example in Figure 4.8 B, cyclic chains are preferred near \(r = 1.5\sigma\).

Given that the spatial resolution of NR is limited by the inverse of the maximum scattering wavevector, the experimentally measured surface excess represents an integrated excess over the spatial resolution length. To aid comparison between theory and experiment, I calculate the integrated excess of linear chains from the Wall-PRISM results. The normalized densities profiles from the surface of both components are calculated by first calculating the normalized density of the two blend component sites by using the bulk density and \(g(r)\) calculated from Wall-PRISM. The excess of the linear component is calculated as the difference between the linear and ring normalized densities, i.e., \(\Delta \rho(r) = \frac{\rho_{\text{Linear}}(r)}{\rho_{\text{Linear}}} - \frac{\rho_{\text{Cyclic}}(r)}{\rho_{\text{Cyclic}}}\). A plot of this difference is shown in Figure 4.9. A positive value of \(\Delta \rho(r)\) indicates that the normalized density of the linear density profiles is in excess of the cyclic chain. When the value is negative it indicates that the density profile for the cyclic chain will be in excess. The value of the net integrated excess, IE, is then determined by taking the integral of the density profile differences over the entire range.

\[
IE = \int_{0}^{\infty} dr \Delta \rho(r)
\]  

Figure 4.10 plots the net integrated excess IE versus the bulk density. For the lowest densities in the range of \(0.0 - 0.006\) monomers/\(\sigma^3\) the cyclic component is actually preferred. This is likely due to the fact that the cyclic chains are smaller because of their more restricted configurations. At higher densities, the linear chains pack better at a flat surface, and are then found in increasing excess at the surface until the integrated excess hits a maximum at around a density of \(0.035\) monomers/\(\sigma^3\) after which the integrated excess decreases, though still shows preference for the linear chains.
Figure 4.9: The difference in normalized density, with the regions in dark gray where the linear chain is in excess and in light gray the cyclic.

Figure 4.10: Net integrated excess of linear chains for varying volume fractions ($\eta$).
4.2.1 Resolution smoothing to compare with experimental density profiles

To compare with experimental density profiles derived from NR, the theoretical Wall-PRISM results must be smoothed to account for the resolution of the NR experiments. The subangstrom resolution of the simulation can be smoothed to account for this to give a density profile result at a resolution similar to that of the NR scattering results. The Hann window method [86, 87] is used to smooth the function using a resolution of \( k_{\text{max}} = \frac{1}{2} \sigma \). For comparing with experiment, I estimate the diameter of the PS monomer to be \( \sigma = 5.0 \text{Å} \). Figure 4.11 illustrates how the effect of smoothing the density profiles for both linear and cyclic chains. The packing structure is washed away by the smoothing, leaving behind only the averaged slight excess of linear over a longer distance.

![Figure 4.11](image)

**Figure 4.11**: Wall-PRISM density profiles for linear and cyclic blends at \( \eta = 0.45 \), A) without, and B) with smoothing using the Hann window method.

The smoothed Wall-PRISM density profile for linear chains are compared with the experimental NR results in Figure 4.12. The black lines indicate the density profile for the linear chains. While they do not match exactly, there is qualitative agreement on width and magnitude of the excess density of linear chains.
Figure 4.12: Comparison of linear chain density profiles with depth, \( z \), of a linear/cyclic blend. Black: NR of the free surface of a blend, Red dotted: smoothed Wall-PRISM theory of a blend next to a hard wall. Note in the NR profiles, the density falls to the left, as expected for a free surface.
4.2.2 Soft wall derivation

The previous calculations assumed the polymer blend was in contact with a perfectly hard wall. This is an approximation meant to capture the free energy penalty of monomers being found beyond the interface with a substrate (due to steric/repulsive exclusion) or beyond a free surface (due to a loss of cohesive energy). However, in experiment both the polymer/substrate interface and polymer/air surface will likely allow for some penetration of the polymer beyond the nominal boundary. To account for this I can modify the Wall-PRISM closure approximation with a “soft” quadratic potential, characterized by a stiffness $S$, rather than a hard wall potential, as depicted in Figure 4.13.

$$V(z)_{Wall/Chain} = \begin{cases} 
S \left( \frac{z + H}{\sigma} \right)^2 & z < -\left( \frac{H}{2} \right) \\
0 & -\left( \frac{H}{2} \right) \leq z \leq \left( \frac{H}{2} \right) \\
S \left( \frac{z - H}{\sigma} \right)^2 & z > \left( \frac{H}{2} \right) 
\end{cases} \quad (4.20)$$

Figure 4.13: Soft wall potential compared with hard wall infinite potential. The start of the walls is at $\pm \frac{H}{2}$ where $H = 30$. See Figure 3.4.

Here I solve the Wall-PRISM equations for a soft wall potential using the PY closure. Recall that for Wall-PRISM the closure is defined in Equation 3.39. However, this closure is defined for an infinitely hard sphere. Using the full definition of the PY closure in Equation 3.17, I can apply the PY closure to the quadratic “soft” wall potential. In the Picard solution
scheme, $C_{ap}(z)$ and $g_{aw}(z)$ are related through the function $\gamma_{aw}(z)$ by

$$g_{aw}(z) = \gamma_{aw}(z) + C_{ap}(z) \tag{4.21}$$

I can then use the full PY definition (see Equation 3.17) for the direct correlation function for the chain/wall site

$$C_{aw} = (1 - e^{-\phi(z)})g_{aw}(z) \tag{4.22}$$

where $\phi(z) = \beta V(z)_{Wall/Chain}$. I can then use the definition of $C_{ap}(z)$ in Equation 3.40 and Equations 4.22 and 4.21 from above to get the following relationship

$$C_{ap}(z) + C_{aw}^{z=\pm\infty}(z) = (1 - e^{-\phi(z)})(\gamma_{aw}(z) + C_{ap}(z)) \tag{4.23}$$

That can be simplified to give

$$C_{ap}(z) = \gamma_{aw}(z)(e^{-\phi(z)} - 1) - C_{aw}^{z=\pm\infty}(z)e^{-\phi(z)} \tag{4.24}$$

I can double check that using this relationship will give us the PY closure used in the Picard iteration and defined in equation 3.47. Using the hard wall potential

$$V(z)_{StiffWall} = \begin{cases} \infty & z < -\left(\frac{H}{2}\right) \\ 0 & -\left(\frac{H}{2}\right) \leq z \leq \left(\frac{H}{2}\right) \\ \infty & z > \left(\frac{H}{2}\right) \end{cases} \tag{4.25}$$

I find that in between the two walls ($|z| < \frac{H}{2}$) Equation 4.24 becomes

$$C_{ap}(z) = \gamma_{aw}(z)(e^0 - 1) - C_{aw}^{z=\pm\infty}(z)e^0 \tag{4.26}$$

which simplifies to

$$C_{ap}(z) = \gamma_{aw}(z)(1 - 1) - C_{aw}^{z=\pm\infty}(z)1$$

$$C_{ap}(z) = -C_{aw}^{z=\pm\infty}(z) \tag{4.27}$$

Outside the two walls ($|z| > \frac{H}{2}$), Equation 4.24 becomes

$$C_{ap}(z) = \gamma_{aw}(z)(e^{-\infty} - 1) - C_{aw}^{z=\pm\infty}(z)e^{-\infty}$$

which simplifies to

$$C_{ap}(z) = \gamma_{aw}(z)(0 - 1) - C_{aw}^{z=\pm\infty}(z)0$$
\[ C_{ap}(z) = -\gamma_{aw}(z) \] (4.28)

The two conditions in Equations 4.27 and 4.28 match our Picard iteration closure definition for hard walls from Equation 3.47.

A base stiffness value, \( S \), of the wall in Equation 4.20 needs to be determined for calculation of the wall-PRISM closure. I will find the minimum value of \( S \) for the soft potential in Equation 4.20 and Equation 4.24 that will reproduce the results for this first iteration of finding \( C_{ap}(z) \) for the fully stiff wall, calculated using the analytic PY closure in Equation 3.47 and the HS potential in Equation 4.25, to within a tolerance of \( 10^{-12} \). The “base” level of stiffness is then reduced to, in effect, soften the wall. I found the minimum value to replicate the true HS potential to be \( S \approx 65000 \). To show how the magnitude of the stiffness value, \( S \), changes the Wall-PRISM closure I have shown a comparison in Figure 4.14. The larger \( S \) value indeed reproduces the stiff wall results while the smaller \( S \) value deviates near \( -\frac{H}{2} \). This also will happen at \( -\frac{H}{2} \) however for clarity only part of the \( C_{ap}(z) \) plot is shown.

The quadratic potential is defined below where \( \sigma \) is still the monomer diameter and \( S \) sets the stiffness of the wall. As mentioned previously, \( S = 65000 \), is used for a completely stiff wall.

The radial distribution functions for both cyclic and linear chains are plotted in Figure 4.15 for a range of stiffnesses. Even for a reduction of 90\% for \( S = 65000 \) and \( S = 6500 \), the change in the RDF is not substantial as the lines for both are very similar. However, the influence of the softer wall for \( S = 6500 \) is visible as the \( g(r) \) is non-zero at just less than \( r = 0 \). Looking more broadly there is evidence for chain penetration into the wall especially when comparing the “softer” potentials. The peak height especially for lower values of \( S \) are smaller and the peaks themselves are broadened with increasing softness. The peaks also shift left as effectively the wall is shifting left with the softer potential. with the softer
Figure 4.14: Wall-PRISM of $C_{Ap}$ and $C_{Bp}$ where $A$ and $B$ are the cyclic and linear chains respectively for the initial Picard iteration step to show change in wall-PRISM closure for a soft wall. The black line indicates a hard wall is enforced exactly, the gray dotted line shows where the soft wall approximation is made to act like a hard wall and the red dotted shows a true soft wall.

Figure 4.16 plots the net integrated excess of linear chains as a function of stiffness showing that lower MW linear chains are still preferred at the surface even of soft walls. Over all the range of stiffness values there is only about a 10% reduction in linear excess. There is however a sharp drop-off at around $S = 5000$ down to $S = 325$. Values of $S < 325$ were tried, however I was unable to get convergent Wall-PRISM calculations.

4.3 Summary

Wall-PRISM theory is applied to a study of small molecular weight linear/cyclic polystyrene blends. This work was motivated by both NR experiments and SCFT studies of architectural effects in surface segregation of binary polymer blends.
Figure 4.15: Panels A) and B) are the cyclic and linear Wall-Polymer $g(r)$ results respectively. These plots show a range of stiffnesses $S$. 
Figure 4.16: Plot of integrated excess of linear polymers at surface vs $S$ for reducing stiffness of wall.
A new cyclic model was developed based on the NFJC model of linear chains. Also, a new computational method was developed for calculation of the intramolecular radial distribution function for both linear and cyclic chains that allowed for convergent Wall-PRISM calculations. Wall-PRISM calculations were performed for blends of linear and cyclic polymer chains both of 20 monomer units. In agreement with experiment, and in contrast to SCFT, Wall-PRISM calculations showed preference for linear over cyclic chains due to packing effects. Also, Wall-PRISM calculations showed preference for the linear chains directly at the surface as well as further from the surface over a range of densities. Only for dilute systems did cyclic chains show a preference when comparing the integrated excess. For the dilute systems the size of the polymer chains likely had a greater influence than packing leading to preference for the cyclic topology. Linear chains were also shown to still be preferred at the surface even for systems with soft walls.
CHAPTER 5
MULTI-SITE TWO-CHAIN

The following section presents my work to develop a method to allow Two-Chain SC-PRISM to work with molecules and polymers with more than one site type. While the motivation for this work is to eventually apply it to polymer systems, the development done here is only on small molecules. In the strict sense, the work in this chapter is Two-Chain SC-PRISM theory for small molecules but the method will be applicable to polymer systems as well. As a note I will usually refer to Two-Chain SC-PRISM as “Two-Molecule RISM” or just “Two-Molecule” in most of the work done in this chapter as the term “chain” typically refers to polymers. “One-Chain” or “single chain/molecule” refers to the prior SC-PRISM theory that only considers one molecule or polymer chain. This chapter will illustrate the divergence problem of moving to multi-site Two-Chain SC-PRISM and my method for overcoming the divergence. I will then validate and assess the performance of the Two-Molecule method against large molecule simulations (MD or MC) and One-Molecule SC-PRISM for several small molecule systems. These include athermal dimers and trimers as well as testing attractive potentials of Lennard Jones dimers and trimers.

5.1 Two-Chain SC-PRISM numerical hurdles

As mentioned previously, the preliminary Two-Chain SC-PRISM work of Li and Wu [7] had shown improvement over the single chain SC-PRISM results especially for short range correlation results (see Figure 2.6). However, the Two-Chain method was only applied to small homonuclear dimers and polyethylene polymer chains in which all the site types are identical. By having only a single type of site interaction, the RISM/PRISM Equation 3.15, reduces from a matrix to a scalar equation.

As an example of a multi-site system, I consider a modification of a simple homodimer system used by Li in developing the scalar Two-Chain theory. I illustrate here the nature of
the numerical hurdles, namely strong divergences at low wavevector, that occur with naive direct application of the scalar Two-Chain SC-PRISM methodology to a multi-site system. The dimers have a hard sphere diameter of $\sigma = 3.93\,\text{Å}$, with a bond length $L = 3.93\,\text{Å}$ and packing fraction of $\eta = 0.40$. These sizes were chosen as they correspond to the size of monomers and bond lengths used for the polyethylene chains. This results in a molecular density of $\rho_{\text{Dimer}} = 0.006293\,\text{Å}^{-3}$. This dimer can be made into a multi-site model and compared with the scalar results by leaving all other attributes the same and only altering the site label (see Figure 5.1). For the homodimer, the density of sites is $\rho_A = 2\rho_{\text{Dimer}}$. For the heterodimer, the site densities used in the matrix form of the RISM equation are $\rho_{\text{Dimer}} = \rho_A = \rho_B$. The number of discrete points used is $N_{\text{Grid}} = 2048$ that are spaced $\Delta r = 0.1\,\text{Å}$ apart, giving a value of $\Delta k = \pi/(\Delta r N_{\text{Grid}})$ for the grid spacing in Fourier space.

Because the bond is of fixed length, $\hat{\omega}(k)$ is known exactly and given by the following relation for the homodimer scalar version:

$$\hat{\omega}(k) = 1 + \frac{\sin(kL)}{kL}$$  \hspace{1cm} (5.1)

and

$$\hat{\omega}(k) = \begin{bmatrix} 1 & \sin(kL) \\ \sin(kL) & 1 \end{bmatrix}$$  \hspace{1cm} (5.2)

for the heterodimer version.

<table>
<thead>
<tr>
<th>Homodimer</th>
<th>Heterodimer</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Homodimer" /></td>
<td><img src="image2.png" alt="Heterodimer" /></td>
</tr>
</tbody>
</table>

Figure 5.1: The single-site type homodimer model, and the corresponding equivalent multi-site heterodimer model used in the scalar and matrix Two-Molecule SC-PRISM theories respectively. These correspond to the same physical molecule, when B differs only as a label but in no physical property.
For a review of the Two-Molecule scheme see section 3.4.2. Beginning with an initial
guess of $C(r) = 0$, the resulting solvation potential is $W(r) = 0$. This is used as input into
the two-molecule simulation to give $H_{Sim}(r)$. The resulting $\hat{C}_{New}(k)$ is calculated using 3.70.
Below I show the results in Figure 5.2 for this calculation performed using the scalar form of
the equations where only one site type is used, and the matrix form where the A-B site type
interactions are allowed. When comparing panels A) and B) in Figure 5.2 the divergence
in $\hat{C}(k)$ in the matrix (multi-site) form of the Two-Molecule theory is clearly evident for
$k < 0.1 \text{ Å}$. 

Figure 5.2: Plots showing the numerical divergence appearing in $\hat{C}_{New}(k)$ for the first itera-
tion in solving the Two-Chain SC-PRISM theory when using the heterodimer model matrix
form compared with the homodimer scalar form. These results are for an initial guess of
$C(r) = 0$, and thus zero solvation potential. A) is zoomed in to show that the scalar and
matrix results agree for larger $k$. B) shows the scale of the divergences at low $k$.

To understand why the divergence occurs, it is important to note that in the process
of solving for $\hat{C}_{New}(k)$ using the RISM/PRISM equations, a matrix inversion is utilized as
mentioned previously. Here is the explicit expression for a 2x2 matrix inversion below.

$$M \equiv \begin{bmatrix} a & b \\ c & d \end{bmatrix} \quad (5.3)$$

$$M^{-1} \equiv \frac{1}{|M|} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix} \quad (5.4)$$

where
\[ |M| \equiv ad - bc \quad (5.5) \]

is the determinant of the 2x2 matrix. Because of the initial guess of \( \mathbf{C}(r) = 0 \), the initial calculation of \( \hat{\mathbf{C}}_{\text{New}}(k) \) is simply given by the expression

\[ \hat{\mathbf{C}}_{\text{New}}(k) = \hat{\Omega}^{-1}(k) \hat{\mathbf{H}}(k) \hat{\Omega}^{-1}(k) \quad (5.6) \]

In Figure 5.3 I plot the determinant \( |\hat{\omega}(k)| \) and show that the resulting function approaches zero as \( k \to 0 \) leading to a divergence in \( \hat{\Omega}^{-1}(k) \) at small \( k \) because the inverse leads to a prefactor of \( 1/|\hat{\omega}(k)| \). Analytically, this divergence is not a problem because this division by zero is canceled by the low wavevector behavior of \( \hat{\mathbf{H}}(k) \sim O(k^2) \) in the calculation of \( \hat{\mathbf{C}}_{\text{New}}(k) \). However, numerical simulations will have noise present in \( \hat{\mathbf{H}}(k) \) as a result of imperfect noisy sampling. This noise is \( \sim O(1) \) and not \( \sim O(k^2) \), and is therefore amplified by the divergence in \( \hat{\Omega}^{-1}(k) \) when put into the PRISM equations. To understand the nature of the effect by the simulation noise I looked at the series expansion of the \( k \)-space matrix inversion with the expansion of the simulation results used in \( \hat{\mathbf{H}}(k) \).

![Figure 5.3: Plot of the determinant of the intramolecular structure factor for the heterodimer.](image)

The following results were calculated in Mathematica and utilized the Series[] function. One of the benefits of doing the analysis in Mathematica is that there are internal algorithms

81
which will collect the power terms from the multiple matrix multiplications in the calculation. This problem would be significantly more time consuming to solve completely by hand without the use of a symbolic mathematics program. First, I evaluate the series expansion of the intramolecular structure factor for the dimer. Recall \( L \) is the bond length between the sites.

\[
\hat{\omega}(k) = \left[ 1 - \frac{k^2L^2}{6} + \frac{k^4L^4}{120} - \frac{k^6L^6}{5040} \right] (5.7)
\]

The Taylor series expansion for the total correlation function is

\[
\hat{h}(k) = \left[ \hat{h}_{AA}(0) + \frac{k^2}{2} \frac{\hat{h}^{(2)}_{AA}(0)}{k^2} + \frac{k^4}{24} \frac{\hat{h}^{(4)}_{AA}(0)}{k^4} + \frac{k^6}{720} \frac{\hat{h}^{(6)}_{AA}(0)}{k^6} \right] + \left[ \hat{h}_{AB}(0) + \frac{k^2}{2} \frac{\hat{h}^{(2)}_{AB}(0)}{k^2} + \frac{k^4}{24} \frac{\hat{h}^{(4)}_{AB}(0)}{k^4} + \frac{k^6}{720} \frac{\hat{h}^{(6)}_{AB}(0)}{k^6} \right] + \left[ \hat{h}_{BA}(0) + \frac{k^2}{2} \frac{\hat{h}^{(2)}_{BA}(0)}{k^2} + \frac{k^4}{24} \frac{\hat{h}^{(4)}_{BA}(0)}{k^4} + \frac{k^6}{720} \frac{\hat{h}^{(6)}_{BA}(0)}{k^6} \right] + \left[ \hat{h}_{BB}(0) + \frac{k^2}{2} \frac{\hat{h}^{(2)}_{BB}(0)}{k^2} + \frac{k^4}{24} \frac{\hat{h}^{(4)}_{BB}(0)}{k^4} + \frac{k^6}{720} \frac{\hat{h}^{(6)}_{BB}(0)}{k^6} \right] (5.8)
\]

In this context the superscript in parentheses, for instance \((2)\) in \( \hat{h}^{(2)}_{AA}(0) \), indicates the second derivative with respect to \( k \). Using the series expansion definitions in Equations 5.7 and 5.8 as well as Equation 5.6, \( \hat{C}_{\text{New}}(k) \) is calculated. Each of the 4 matrix components \( \hat{C}_{\text{New}} \) \((k), \hat{C}_{AA}(k), \hat{C}_{AB}(k) \) and \( \hat{C}_{BB}(k) \) will be displayed separately for readability.

\[
\hat{C}_{AA}(k) = \frac{9(\hat{h}_{AA}(0) - \hat{h}_{AB}(0) - \hat{h}_{BA}(0) + \hat{h}_{BB}(0))}{10k^4} + \ldots (5.9)
\]

\[
\hat{C}_{AB}(k) = \frac{-9(\hat{h}_{AA}(0) - \hat{h}_{AB}(0) - \hat{h}_{BA}(0) + \hat{h}_{BB}(0))}{10k^4} + \ldots (5.10)
\]

\[
\hat{C}_{BA}(k) = \frac{-9(\hat{h}_{AA}(0) - \hat{h}_{AB}(0) - \hat{h}_{BA}(0) + \hat{h}_{BB}(0))}{10k^4} + \ldots (5.11)
\]

\[
\hat{C}_{BB}(k) = \frac{9(\hat{h}_{AA}(0) - \hat{h}_{AB}(0) - \hat{h}_{BA}(0) + \hat{h}_{BB}(0))}{10k^4}
\]

\[
(5.12)
\]
\[
\hat{C}_{BB}(k) = \frac{9(\hat{h}_{AA}(0) - \hat{h}_{AB}(0) - \hat{h}_{BA}(0) + \hat{h}_{BB}(0))}{k^4 l^4} + \frac{3(-2^2 \hat{h}_{AA}(0) - 3^2 \hat{h}_{AB}(0) - 3^2 \hat{h}_{BA}(0) + 8l^2 \hat{h}_{BB}(0) + 15 \hat{h}_{AA}^{(2)}(0) - 15 \hat{h}_{AB}^{(2)}(0) - 15 \hat{h}_{BA}^{(2)}(0) + 15 \hat{h}_{BB}^{(2)}(0))}{10k^2 l^4} + \ldots \tag{5.12}
\]

The resulting matrix elements contain terms with \(k^{-4}\) and \(k^{-2}\) as well as higher order terms present. In the case of a perfect simulation without noise these terms should have zero contribution. The zero wave vector values for the total correlation functions, e.g. \(\hat{h}_{AA}(0)\), in the \(k^{-4}\) and \(k^{-2}\) are such that they cancel leaving the \(k^{-4}\) and \(k^{-2}\) terms at small \(k\) with no contribution. I then analyze on the discretized grid how the small \(k\) values affect the divergence in terms of their magnitude. The smallest \(k\) value used in my discrete system is \(k_1 = 0.0153 \, \text{Å}^{-1}\), which means that \(k_1^{-4} = 1.806 \times 10^7 \, \text{Å}^4\). Figure 5.2 B) shows a similar order of magnitude in terms of the low \(k\) divergence. There is likewise a similar, though smaller, contribution to the divergence from the \(k^{-2}\) terms as well that are on order of \(10^3\). Inspecting the \(k^{-4}\) term for all the matrix elements in Equations 5.9 through 5.12, a requirement is that \(\hat{h}_{AA}(0) - \hat{h}_{AB}(0) - \hat{h}_{BA}(0) + \hat{h}_{BB}(0) = 0\) in order to eliminate the contribution of the \(k^{-4}\) portion. This requirement is enforcing the stoichiometry of the system. In the limit of a perfect simulation (which would give the exact values of the correlation functions), this condition would be automatically satisfied. A slight deviation from satisfying this rule will result in a divergence in \(\hat{C}_{New}(k)\). The same line of thinking is then applied to the \(k^{-2}\) terms, resulting in further, larger equations containing higher order terms (\(\hat{h}_{AA}^{(2)}(0)\) etc.) that must all be satisfied simultaneously.

The resulting equations, for the heterodimer case, that must be satisfied to remove the spurious divergence are the following.

\[
(\hat{h}_{AA}(0) - \hat{h}_{AB}(0) - \hat{h}_{BA}(0) + \hat{h}_{BB}(0)) = 0 \tag{5.13}
\]

\[
8l^2 \hat{h}_{AA}(0) - 3l^2 \hat{h}_{AB}(0) - 3l^2 \hat{h}_{BA}(0) + 2l^2 \hat{h}_{BB}(0) + 15 \hat{h}_{AA}^{(2)} - 15 \hat{h}_{AB}^{(2)} - 15 \hat{h}_{BA}^{(2)} + 15 \hat{h}_{BB}^{(2)} = 0 \tag{5.14}
\]
\[3l^2 \hat{h}_{AA}(0) - 8l^2 \hat{h}_{AB}(0) + 2l^2 \hat{h}_{BA}(0) + 3l^2 \hat{h}_{BB}(0) + 15\hat{h}_{AB}^{(2)}(0) - 15\hat{h}_{BA}^{(2)}(0) + 15\hat{h}_{BB}^{(2)}(0) = 0 \tag{5.15}\]

\[3l^2 \hat{h}_{AA}(0) + 2l^2 \hat{h}_{AB}(0) - 8l^2 \hat{h}_{BA}(0) + 3l^2 \hat{h}_{BB}(0) + 15\hat{h}_{AB}^{(2)}(0) - 15\hat{h}_{BA}^{(2)}(0) + 15\hat{h}_{BB}^{(2)}(0) = 0 \tag{5.16}\]

\[-2l^2 \hat{h}_{AA}(0) - 3l^2 \hat{h}_{AB}(0) + 8l^2 \hat{h}_{BA}(0) + 15\hat{h}_{AB}^{(2)}(0) - 15\hat{h}_{BA}^{(2)}(0) + 15\hat{h}_{BB}^{(2)}(0) = 0 \tag{5.17}\]

To summarize, I have identified a set of equations, based on the series expansion of \(\hat{C}_{\text{New}}(k)\) and using 5.9 through 5.12, that must be satisfied to eliminate the divergence.

These equations are automatically satisfied for the exact solutions, but finite simulations introduce noise in the numerical intermolecular correlation functions, \(\hat{h}_{\alpha\gamma}(k)\), that produce a spurious low wave vector divergence in the theory. For instance, for a symmetric heterodimer, 5.13 implies that all matrix elements \(\hat{h}_{\alpha\gamma}(k)\) are equal. However, for Figure 5.2, the simulation produced numerical values \(\hat{h}_{AA}(0) = -694.922, \hat{h}_{AB}(0) = \hat{h}_{BA}(0) = -694.743, \) and \(\hat{h}_{BB}(0) = -692.260\) that are not all equal. This difference results in a non-zero coefficient for the \(k^{-4}\) term since \(\hat{h}_{AA}(0) - \hat{h}_{AB}(0) - \hat{h}_{BA}(0) + \hat{h}_{BB}(0) = 2.30477\). Using these values, the first term in the equation for \(\hat{C}_{AA}(k)\) is

\[\frac{9(\hat{h}_{AA}(0) - \hat{h}_{AB}(0) - \hat{h}_{BA}(0) + \hat{h}_{BB}(0))}{k^4 l^4} = 1.57 \times 10^6\]

consistent with the divergence magnitude seen in Figure 5.2 B). The other matrix elements have divergences that are similar in magnitude, though sometimes negative.

The expansion explains why the divergence is present and also shows a way to correct the problem. The origin of the problem comes from using noisy \(r\) space simulation data, however, all the equations are written in terms of \(k\) space total correlation functions. The task then is to remove the noise appropriately from the simulation data. One can take advantage of the mathematical relationships between the zero wave vector of the Fourier transform of a function and the function in \(r\) space function, specifically
This means that I can control the values of $\hat{h}_{\alpha \gamma}(0)$, $\hat{h}_{\alpha \gamma}^{(2)}(0)$, ... etc to correct them and have them cancel by addressing the simulation directly in $r$ space. One thing to note here is that the $k^{-4}$ and $k^{-2}$ conditions are not all linearly independent. Also, for the case of the symmetric heterodimer, $h_{AB}(r) = h_{BA}(r)$, and the 5 $k^{-4}$ and $k^{-2}$ conditions can be reduced to 3 linearly independent equations.

My initial attempts at controlling this divergence by controlling the $h_{\alpha \gamma}(r)$ integral value were to piece on a tail that would decay to zero for large $r$. Figure 5.4 depicts the addition of a tail to the simulation data. The working hypothesis motivating the addition of a tail is that the simulation is doing a good job capturing the correlation functions at shorter distances, and it is rather the missing correlation functions at longer distances that result in the zero wavevector values of $\hat{h}_{\alpha \gamma}(k)$ to be unequal. Thus the correlation function $h_{\alpha \gamma}(r)$ consists of the raw simulation data up to a maximum distance, which can be numerically integrated using a polynomial interpolation, followed by an analytic tail, which can be analytically integrated. I assumed an exponential decay form for the fitted tails. This results in 3 exponential tails $\alpha e^{-\beta r}$ with preexponential coefficients $\alpha_{AA}$, $\alpha_{AB}$, and $\alpha_{BB}$ and exponential coefficients of $\beta_1, \beta_2$ and $\beta_3$. These 6 parameters are determined by a least squares fit of the 3 tails to the data over a limited range, subject to the non-divergence constraints of the 5 Equations (5.13, 5.14, 5.17, 5.15 and 5.16) that can be reduced to 3 linearly independent equations. Figure 5.4 A) shows the range of data, plotted in green, used to fit the tails. Figure 5.4B) shows the use of the fit tail in place of the noisy data at the highest values of simulated $r$ and beyond.
Figure 5.4: Method for modifying simulation data to satisfy non-divergence constraints. Panel A) shows an example of the points selected for tail fitting and B) shows application of the tail fitting.

This method proved to only work in a number of limited cases and often resulting in plots that looked like Figure 5.5. There would be a small to moderate reduction in divergence at low $k$ values for the newly calculated $\tilde{C}(k)$, however, not sufficient to completely remove the divergence. After looking at how the method attempted to work it becomes obvious why this method is a flawed approach. The basic idea is to take the numerically problematic and uncontrollable simulation results at smaller $r$ values and fix it with a fully tunable function for larger $r$ values with the tail fit. As mentioned previously the terms $\hat{h}_{\alpha\gamma}(0), \hat{h}^{(2)}_{\alpha\gamma}(0)$ and similar terms are divided into two portions for the equivalent $r$-space integral. In terms of their contribution to the integral $4\pi \int_0^\infty dr h_{\alpha\gamma}(r)r^2$, the tail portions were found to only contribute about 0.09% of the total integral value. Because so little of the total integral value can be adjusted by the tail it just isn’t always possible to make up the differences caused by the simulation with the tail portion alone. Using this method resulted in ill fitting tails that lead to unusable, unphysical results. Different approaches were tried including extending the tail section to encompass more of the total integral, however, most attempts resulted in either continued divergences in $\tilde{C}(k)$, or results that fundamentally changed the new $\tilde{C}(k)$ too much when compared with the scalar results.
Figure 5.5: Poor tail fitting as a result of trying to satisfy $k^{-4}$ and $k^{-2}$ conditions using only the fitted tail parameters. The system is a hard sphere heterodimer outlined in section 5.1, with system parameters of $\sigma_A = \sigma_B = 3.93 \text{ Å}$, and $\rho_{\text{Dimer}} = 0.006293 \text{ Å}^{-3}$.

The conclusion from these studies is that the dominant contribution to the simulation noise leading to the unphysical divergences in $\hat{C}(k)$ are not from the correlation at large distances, as originally hypothesized, but rather at small and intermediate distances. The next section describes a method to appropriately project out the noise from the simulation data at small and intermediate distances to overcome the numerical hurdle associate with divergences in $\hat{C}(k)$.

5.2 The “tweaking” method to project out unphysical noise from simulation correlation functions

The previous section involved my initial attempt to use a fitted tail function to correct the two-molecule multi-site numerical problems and showed that correcting the tail was not sufficient to correct the $\hat{h}_{\alpha\gamma}(0), \hat{h}_{\alpha\gamma}^{(2)}(0) \ldots$ etc terms to satisfy the $k^{-4}$ and $k^{-2}$ conditions sufficiently and consistently. I have developed a new method that will modify the simulation data itself but in such a way that it remains relatively unaffected, in keeping with the concept of only filtering out simulation noise. Because so much of the calculation of the integral relies on the magnitude of the smaller $r$ region of $h_{\alpha\gamma}(r)$ from the simulation, this new method
will take each discrete point from the simulation and modify it very slightly. These slight modifications, it is hoped, will leave the results true to the original simulations while having a pronounced effect on the calculation of the zero wave vector components. Since this section involves matrices with many entries and lengthy mathematical expressions that are difficult to write in the limited space of the page, the reader is referred to C.2 for more detail, as it contains the actual software for the system tested in section 5.5.2.

Because I will be modifying many of the simulation points, it is more useful to used the trapezoid method of numerical integration rather than polynomial interpolation. The trapezoid method is shown in the following equation.

$$\int_{A}^{B} f(r) \approx \frac{h}{2} \left( f(r_1) + 2 \sum_{i=2}^{n-1} f(r_i) + f(r_n) \right)$$

(5.19)

where $h = \Delta r$ is the grid spacing and $n = N_{Grid}$ is the total number of discrete points used. Each of the terms $\hat{h}_{\alpha\gamma}(0), \hat{h}_{\alpha\gamma}^{(2)}(0)$...etc can be expressed numerically in terms of this approximation as defined below.

$$\hat{h}_{\alpha\gamma}(0) = 4\pi \frac{\Delta r}{2} (h_{\alpha\gamma}(r_1) r_1^2 + 2 \sum_{i=2}^{n-1} h_{\alpha\gamma}(r_i) r_i^2 + h_{\alpha\gamma}(r_n) r_n^2)$$

$$\hat{h}_{\alpha\gamma}^{(2)}(0) = -\frac{4\pi}{3} \frac{\Delta r}{2} (h_{\alpha\gamma}(r_1) r_1^4 + 2 \sum_{i=2}^{n-1} h_{\alpha\gamma}(r_i) r_i^4 + h_{\alpha\gamma}(r_n) r_n^4)$$

... 

$$\hat{h}_{\alpha\gamma}^{(8)}(0) = \frac{4\pi}{9} \frac{\Delta r}{2} (h_{\alpha\gamma}(r_1) r_1^{10} + 2 \sum_{i=2}^{n-1} h_{\alpha\gamma}(r_i) r_i^{10} + h_{\alpha\gamma}(r_n) r_n^{10})$$

(5.20)

The remaining terms are not shown in the interest of clarity, but can be found by the same method using Equation 5.18.

The simulation data are now modified by the addition of small changes, or what I call a vector of “tweaks”, over a range of $r$. There are no tweaks made inside the hard core overlap distance, since for perfectly hard spheres $h_{\alpha\gamma}(r) = 0$ for $r < 1/2(\sigma_\alpha + \sigma_\gamma)$. Thus the first point to be tweaked is at a distance $r_i$ that is just outside this region. Tweaks are then made on the simulation data up to where the actual simulation is cut-off and the single chain.
SC-PRISM PY solution is pieced on. \( i = S_{\alpha \gamma} \) will indicate the index where the “tweaking” segment starts and \( i = P_{\alpha \gamma} \) will indicate the index for when the tweaking range stops for each respective \( \alpha \gamma \) pair. These values will not necessarily be the same for all \( h_{\alpha \gamma}(r) \). I then rewrite my integrals from Equation 5.20 for the \( \hat{h}_{\alpha \gamma}(0) \) and other terms as

\[
\hat{h}_{\alpha \gamma}(0) = 4\pi \frac{\Delta r}{2} (h_{\alpha \gamma}(r_1) r_1^2 + 2 \sum_{i=2}^{S_{\alpha \gamma} - 1} h_{\alpha \gamma}(r_i) r_i^2 + 2 \sum_{i=S_{\alpha \gamma}}^{P_{\alpha \gamma}} (h_{\alpha \gamma}(r_i) + q_i^{\alpha \gamma}) r_i^2 + 2 \sum_{i=P+1}^{n-1} h_{\alpha \gamma}(r_i) r_i^2 + h_{\alpha \gamma}(r_i) r_i^2) \\
\hat{h}_{\alpha \gamma}^{(2)}(0) = -\frac{4\pi}{3} \frac{\Delta r}{2} (h_{\alpha \gamma}(r_1) r_1^4 + 2 \sum_{i=2}^{S_{\alpha \gamma} - 1} h_{\alpha \gamma}(r_i) r_i^4 + 2 \sum_{i=S_{\alpha \gamma}}^{P_{\alpha \gamma}} (h_{\alpha \gamma}(r_i) + q_i^{\alpha \gamma}) r_i^4 + 2 \sum_{i=P+1}^{n-1} h_{\alpha \gamma}(r_i) r_i^4 + h_{\alpha \gamma}(r_i) r_i^4) \\
\vdots \\
\hat{h}_{\alpha \gamma}^{(8)}(0) = \frac{4\pi}{9} \frac{\Delta r}{2} (h_{\alpha \gamma}(r_1) r_1^{10} + 2 \sum_{i=2}^{S_{\alpha \gamma} - 1} h_{\alpha \gamma}(r_i) r_i^{10} + 2 \sum_{i=S_{\alpha \gamma}}^{P_{\alpha \gamma}} (h_{\alpha \gamma}(r_i) + q_i^{\alpha \gamma}) r_i^{10} + 2 \sum_{i=P+1}^{n-1} h_{\alpha \gamma}(r_i) r_i^{10} + h_{\alpha \gamma}(r_i) r_i^{10}) + (5.21)
\]

where the underscore in \( \hat{h}_{\alpha \gamma}(0) \) indicates that a tweak of magnitude \( q_i^{\alpha \gamma} \) (sometimes abbreviated \( q_i \) below) has been added to the original simulation values \( h_{\alpha \gamma}(r_i) \). See Figure 5.6 for an illustration of the addition of tweaks. As above, the omitted terms above can be found using the same method.

I can now rewrite the equations setting the \( k^{-4} \) and \( k^{-2} \) coefficients to zero for dimers (Equations 5.13 - 5.17) using the respective trapezoid summation definitions with the additional tweaking constants \( q_i^{\alpha \gamma} \) found in 5.17. Because of the shared \( q_i \) values between different \( k^{-4} \) and \( k^{-2} \) non-divergence conditions, the entire system represents a linear set of equations and can be written in matrix form. Note that \( \hat{h}_{\alpha \gamma}^{(m)}(0) \) denotes the \( m \)-th derivative with respect to \( k \) at \( k = 0 \). (The \( m = 0 \) case is just \( \hat{h}_{\alpha \gamma}(0) \).) Equations 5.13 - 5.17 are then rewritten as such below. For the resulting matrix the rows correspond to the \( k^{-4} \) and \( k^{-2} \) conditional equations and the columns indicate the respective \( h_{\alpha \gamma}(r) \) used in the integral. The system of equations to solve to find all the \( q_i \) values is shown below.
Figure 5.6: Proposed new method where each value of $h(r_i)$ in a range receives an additive “tweak” $q_i$. The tweaks can be positive or negative. The blue circles represent the raw simulation data, $h(r)_{raw}$, and the black arrows represent the addition of tweaks.

\[
\hat{h}_{AA}(0) - \hat{h}_{AB}(0) - \hat{h}_{BA}(0) + \hat{h}_{BB}(0) = 0
\]

\[
8l^2\hat{h}_{AA}(0) + 15\hat{h}_{(2)}^{(AA)}(0) - 3l^2\hat{h}_{AB}(0) - 15\hat{h}_{(2)}^{(AB)}(0) - 3l^2\hat{h}_{BA}(0) - 15\hat{h}_{(2)}^{(BA)}(0) - 2l^2\hat{h}_{BB}(0) + 15\hat{h}_{(2)}^{(BB)}(0) = 0
\]

\[
3l^2\hat{h}_{AA}(0) + 15\hat{h}_{(2)}^{(AA)}(0) - 8l^2\hat{h}_{AB}(0) - 15\hat{h}_{(2)}^{(AB)}(0) - 2l^2\hat{h}_{BA}(0) - 15\hat{h}_{(2)}^{(BA)}(0) + 3l^2\hat{h}_{BB}(0) + 15\hat{h}_{(2)}^{(BB)}(0) = 0
\]

\[
-2l^2\hat{h}_{AA}(0) + 15\hat{h}_{(2)}^{(AA)}(0) - 3l^2\hat{h}_{AB}(0) - 15\hat{h}_{(2)}^{(AB)}(0) - 3l^2\hat{h}_{BA}(0) - 15\hat{h}_{(2)}^{(BA)}(0) + 8l^2\hat{h}_{BB}(0) + 15\hat{h}_{(2)}^{(BB)}(0) = 0
\]

These equations can be simplify by noticing that in Equations 5.21, the original raw $h_{a\gamma}(r_i)$ terms can be separated from the $q_i$ tweaking terms and taken over to the RHS. For example, the first equation in 5.21 can be rewritten to have separate sums for the $h_{a\gamma}(r)$ and $q_i$ terms:
\[
\hat{h}_{\alpha\gamma}(0) = 4\pi \frac{\Delta r}{2} (h_{\alpha\gamma}(r_1)r_1^2 + 2 \sum_{i=2}^{S_{\alpha\gamma}^-} h_{\alpha\gamma}(r_i)r_i^2 + 2 \sum_{i=S_{\alpha\gamma}^-}^{P_{\alpha\gamma}} (h_{\alpha\gamma}(r_i))r_i^2 + \\
2 \sum_{i=S_{\alpha\gamma}^-}^{P_{\alpha\gamma}} (q_i)r_i^2 + 2 \sum_{i=P+1}^{n-1} h_{\alpha\gamma}(r_i)r_i^2 + h_{\alpha\gamma}(r_i)r_i^2)
\]

Notice that this separation leaves the original trapezoidal integral definition of \(\hat{h}_{\alpha\gamma}(0)\) with the addition of the sum term of unknown \(q_i\) values, i.e.

\[
\hat{h}_{\alpha\gamma}(0) = \hat{h}_{\alpha\gamma}(0) + \Delta r \sum_{i=S_{\alpha\gamma}^-}^{P_{\alpha\gamma}} (4\pi)(q_i)r_i^2 \tag{5.23}
\]

Likewise, the other \(\hat{h}_{\alpha\gamma}(m)\) can be found using the definitions of \(\hat{h}_{\alpha\gamma}(m)\) the \(\hat{h}_{\alpha\gamma}(0)\), and moving the analogous terms of Equation 5.22 to the right hand side (RHS). I define the results for each RHS below:

\[
RHS_1 = -(\hat{h}_{AA}(0) - \hat{h}_{AB}(0) - \hat{h}_{BA}(0) + \hat{h}_{BB}(0))/\Delta r \tag{5.24}
\]

\[
RHS_2 = -(8l^2\hat{h}_{AA}(0) + 15\hat{h}_{AA}^{(2)}(0) - 3l^2\hat{h}_{AB}(0) - 15\hat{h}_{AB}^{(2)}(0) - 2l^2\hat{h}_{BB}(0) + 15\hat{h}_{BB}^{(2)}(0))/\Delta r \tag{5.25}
\]

\[
RHS_3 = -(3l^2\hat{h}_{AA}(0) + 15\hat{h}_{AA}^{(2)}(0) - 8l^2\hat{h}_{AB}(0) - 15\hat{h}_{AB}^{(2)}(0) + 3l^2\hat{h}_{BB}(0) + 15\hat{h}_{BB}^{(2)}(0))/\Delta r \tag{5.26}
\]

\[
RHS_4 = -(2l^2\hat{h}_{AA}(0) + 15\hat{h}_{AA}^{(2)}(0) - 3l^2\hat{h}_{AB}(0) - 15\hat{h}_{AB}^{(2)}(0) + 8l^2\hat{h}_{BB}(0) + 15\hat{h}_{BB}^{(2)}(0))/\Delta r \tag{5.27}
\]

The left hand side (LHS) then becomes only the terms with the unknown \(q_i\) constants. Thus the equations above can be cast into matrix form. Because of the large number of elements, the matrix is condensed to fit reasonably on the page. The columns with \(\ldots\) indicate the rest of the \(r\) values in the series from the equations and are defined explicitly later. The \(\ldots\) in the fourth column indicates are the \(r_{BA}\) columns that were removed to save space but will follow the same pattern as the others.
Each entry in this matrix is actually a compact form that will be expanded into more columns. For example, the first entry in the matrix
\[
\left[ \begin{array}{cccc}
4\pi r_{AA}^2 & \ldots & -4\pi r_{AB}^2 & \ldots \\
8l^2 r_{AA}^2 + \frac{15}{3}(-4\pi)r_{AA}^4 & \ldots & -3l^2 r_{AB}^2 - \frac{15}{3}(-4\pi)r_{AB}^4 & \ldots \\
3l^2 r_{AA}^2 + \frac{15}{3}(-4\pi)r_{AA}^4 & \ldots & -8l^2 r_{AB}^2 - \frac{15}{3}(-4\pi)r_{AB}^4 & \ldots \\
-2l^2 r_{AA}^2 + \frac{15}{3}(-4\pi)r_{AA}^4 & \ldots & -3l^2 r_{AB}^2 - \frac{15}{3}(-4\pi)r_{AB}^4 & \ldots \\
\end{array} \right]
\]
(5.30)

Each entry in this matrix is actually a compact form that will be expanded into more columns. For example, the first entry in the matrix \(\left\{ 4\pi r_{AA}^2 \ldots \right\}\) expands to
\[\left\{ \ldots 4\pi r_{AA}^2, 4\pi r_{AA}^2 + 1 \ldots 4\pi r_{AA}^2 \ldots \right\}\]
and the lower right corner of the matrix for the entry \(\left\{ l^2 r_{BB}^2 + \frac{15}{3}(-4\pi)r_{BB}^4 \ldots \right\}\) expands to
\[\left\{ \ldots l^2 r_{BB}^2 + \frac{15}{3}(-4\pi)r_{BB}^4, l^2 r_{BB}^2 + 1 \ldots l^2 r_{BB}^2 \ldots \right\}\]
with the \(r_{BB}^p\) terms being the final column.

Each collection of columns for a given \(\alpha\gamma\) pair will have \(P_{\alpha\gamma} - S_{\alpha\gamma} + 1\) terms. So for the \(A - B\) dimer system being considered, the resulting matrix will have \(Q\) total columns (see Equation 5.31 below) and in this case four rows, however, the number of rows is determined by the number of equations needed to solve the equations eliminating the \(k^{-n}\) type divergences. \(Q\) also defines the number of elements in the “tweaking” vector. The matrix defined in Equation 5.30 can be represented in a more compact form by substituting \(R_{\alpha\gamma}\) for all the \(r_{\alpha\gamma}\) terms above.

\[
Q = (P_{AA} - S_{AA}) + (P_{AB} - S_{AB}) + (P_{BA} - S_{BA}) + (P_{BB} - S_{BB})
\]  
(5.31)
This set of linear equations is underdetermined since there are more variables than equations. To solve these linear equations with minimum tweak magnitudes, I use the Moore-Penrose pseudo inverse: A linear system

\[ A \cdot x = b \]

can be solved to give a solution vector

\[ z = A^+ b \]

where \( A^+ \) is the pseudo inverse of \( A \) such that

\[ \|z\|_2 \leq \|x\|_2 \]

and \( \|\cdot\|_2 \) is the Euclidean norm. This gives us a solution that is the minimum solution to the under-defined problem. Therefore the \( q \) vector components solved in this way to satisfy the \( k^{-n} \) conditions will be the smallest of all the possible solutions.

I tested this tweaking method on the same system in Figure 5.5 for which the tail fitting scheme failed. Using this tweaking method successfully eliminated the divergence in \( \hat{C}(k) \) divergence (see Figure 5.7), and moreover passed the test of reproducing the scalar version of
the Two-Chain theory as a benchmark, as it should for symmetric heterodimers. However, it is necessary to determine how much the “tweaking” vector is changing the results from the raw simulation data. Figure 5.8 A) - C) shows the change of the $h_{\alpha\gamma}(r)$ from the raw simulation data to that of the simulation data with the tweaking vector added, which I call the “fixed” data. Rather than the points being shifted in any direction the points all seem to shift one direction or another. This is something that is observed upon repeated testing. Typically the $AA$ and the $BB$ will shift in the opposite direction of the $AB/BA$ data. The individual tweaks moving in the same direction is consistent with how the pseudo-inverse works. If the points were shifted in arbitrary random directions, the individual tweaking values would have to be larger overall since shifting 2 adjacent points in opposite directions would have a smaller effect on the overall integral than two adjacent points moving a smaller amount in the same direction. Doing very small shifts in the same direction can satisfy the optimization constraints of the $k^{-n}$ conditions while changing the overall plot very little in keeping with minimizing the Euclidean norm of the resulting tweaking vector $q$. A more helpful look at the application of the tweaking vector to the resulting fixed $h_{\alpha\gamma}(r)$ data is too look at the differences in terms of the respective $\hat{h}_{\alpha\gamma}(k)$ since this is what is being applied to the PRISM equations to give $\hat{C}_{\text{New}}(k)$. Figure 5.8 D) shows the difference between the raw and “fixed” $\hat{h}_{\alpha\gamma}(k)$. For all the respective $\alpha\gamma$ pairs the largest differences come at low wave vectors. This is consistent with where the $\hat{C}_{\text{New}}(k)$ was previously showing divergence. At longer wave vectors the difference approaches zero indicating that the tweaking vector technique is targeting the correct part of the simulation data for correction.

While the initial iteration showed promise in correcting the divergence, as seen above, further iterations showed numerical instability. Using small mixing fractions only delayed computational problems. Testing indicated that the tweaking vector method was working correctly upon each iteration, and so likely the problem was with the Picard solution scheme itself. To address these numerical problems associated with the convergence scheme, an alternate method for solving the Two-Molecule theory was used and is outlined in the next
Figure 5.7: \( \hat{C}_{New}(k) \) from multi-site Two-Chain theory using the “tweaking” method for the first iteration using identical conditions as those in Figure 5.2. A) shows elimination of the low-\( k \) numerical divergence, and good agreement with scalar Two-Chain theory B) is zoomed in to show the match.

section.

5.2.1 Alternate self-consistent scheme

An alternate to the Picard iteration method based on convergence in \( \hat{C}(k) \), used for solving the Two-Chain PRISM theory, was developed by Donley et al. [75, 88]. First a new function, \( \hat{\Delta}(k) \), is defined to be

\[
\hat{\Delta}(k) = -(1 - \hat{\Omega}(k)\hat{C}_{Start}(k))^{-1}\hat{C}_{Start}(k)
\]  

(5.33)

where \( \hat{C}_{Start}(k) \) is either the initial guess or the result of the previous iteration. Using the RISM/PRISM Equation 3.15, \( \hat{H}_{PRISM}(k) \) and \( \hat{C}_{Start}(k)\hat{S}(k)\hat{C}_{Start}(k) \) can be written as:

\[
\hat{H}_{PRISM}(k) = \hat{\Omega}(k)\hat{\Delta}(k)\hat{\Omega}(k)
\]  

(5.34)

\[
\hat{C}_{Start}(k)\hat{S}(k)\hat{C}_{Start}(k) = -\hat{C}_{Start}(k) - \hat{\Delta}(k)
\]  

(5.35)

The results from equation 5.35 can then be used with the desired solvation potential definition from Table 3.1. The results from the simulation with this solvation potential can then be used to find \( \hat{H}_{Simulation}(k) \). The change in the direct correlation function is then found
Figure 5.8: “Tweaking” raw simulation $h(r)$ to obtain “fixed” $h(r)$ that do not show low-$k$ divergence. A), B) and C) show the change in $h(r)$ with the addition of the tweaking vector for $AA$, $AB$ and $BB$ respectively. D) shows the change in $k$-space as it is applied to $\tilde{C}_{New}(k)$. 
with the following relation

$$\delta \hat{C}(k) = \hat{\Omega}(k)^{-1}\hat{H}_{Sim}(k)\hat{\Omega}(k)^{-1} - \hat{\Delta}(k)$$

The new guess for the direct correlation function is then given by

$$\hat{C}_{New}(k) = \hat{C}_{Start}(k) + a(\delta \hat{C}(k))$$

The parameter $a$ is used to improve convergence by reducing the effect of $\delta \hat{C}(k)$. A typical range for the values used in this thesis is $0.005 < a < 0.05$. The total computational method is outlined in Figure 5.9.

![Figure 5.9: Alternate iterative Two-Chain solution scheme using method from [75, 88] and including tweaking vector step. AFD is the Average Fractional Difference.](image)

After utilizing the new convergence scheme, the numerical issues for multiple iterations went away and the calculations were much better behaved. The solution scheme was run to convergence and compared with scalar results for an identical system from the original Two-Chain work on homonuclear dimers [7]. A comparison is shown in Figure 5.10.

A small issue was discovered with my code where normalization of my simulation data was off by a single index. This resulted in small non-zero portion of the $h(r)$ for $r/\sigma > 1.0$. 

97
Figure 5.10: Comparison of symmetric heterodimer intermolecular correlation function, $h(r)$, using the multi-site two-molecule theory with the equivalent homodimer solved using the scalar Two-Chain theory. The x-axis corresponds to $r$ divided by the hard sphere diameter of the sites ($\sigma = 3.93$ Å).

Some of the simulations in the thesis have this small error. An effort has been made to correct these and has shown only minor changes in the results. I will make the reader aware of any results that contain this error. Additionally, work done on the original two-molecule scalar theory [7] incorporated LJ in addition to the hard sphere potentials. A natural attractive homonuclear dimer system to study is molecular nitrogen. Here I use the molecular nitrogen system as a proof of concept for this multi-site Two-Chain theory and method of solution with a LJ potential. The results can be seen in Figure 5.11. The molecular nitrogen system is modeled as a multi-site heterodimer, but in name only with no physical properties differences between the two sites, just as my initial hard sphere system was. All the parameters are kept such that the system can be compared with results for the equivalent homodimer system using the scalar equations. My approach will next be tested on a series of systems of asymmetric dimers where the size of the two sites will vary. This is a system that the original Two-Molecule theory was incapable of solving, but can be treated by my multi-site Two-Molecule theory and solution method.
Figure 5.11: Comparison of multi-site and scalar Two-Molecule theories applied to attractive homodimer system: molecular nitrogen. The system uses an HNC solvation potential, with a molecule density of $\rho = 0.01851 \, \text{Å}^{-3}$, $T = 65 \, \text{K}$, $\epsilon = 0.0739 \, \text{kcal/mol}$, $\sigma = 3.315 \, \text{Å}$ and a bond length of $L = 1.1 \, \text{Å}$. The full Lennard Jones potential was used and is defined in Equation 5.36.

5.3 Overlapping heterodimers

Chandler and coworkers [89] applied RISM theory for a system of 3 different models of overlapping hard spheres. The 3 different models, outlined in Table 5.1, correspond to specific molecules: Model I is liquid $Cl_2$ or $Br_2$, Model II is $H_3C-I$ and Model III $H_3C-F$. Chandler et al. found that the RISM results for these systems did not entirely match the hard sphere Monte Carlo simulation of many dimers, and this difference was especially prevalent at smaller $r$ values. This is an ideal set of models on which to test the Two-Molecule theory to see if there is an improvement over the traditional single molecule (“One-Molecule”) method. Results for all three models will be compared with hard sphere Monte Carlo simulations of 512 molecules. The size of the box is adjusted to give the correct density in each case. While (chandler1977a) does give MC results these were redone to give more easily comparable results. All the Monte Carlo simulations were first run and compared to the published results to verify consistency. All densities presented in the Chandler paper were based on a reduced density of 0.9. The reduced density of $\rho d^3 = 0.9$ is defined where $d$
is the diameter of a sphere of the same total volume as the molecule. They based all dimer parameter values on the site -$A$ diameter, $\sigma_A$, where site -$A$ is the larger of the two spheres. I have reported all results here for $\sigma_A = 3.93\,\text{Å}$, and so will report all quantities including bond lengths as well as molecule densities based on this size. Likewise, the Monte Carlo simulations were performed using these sizes as well.

Table 5.1: Parameters for the 3 overlapping dimer systems to be tested with Two-Molecule theory [89]

<table>
<thead>
<tr>
<th>Model</th>
<th>$\sigma_A(A)$</th>
<th>$\sigma_B(A)$</th>
<th>Bond Length L (Å)</th>
<th>$\rho_{\text{dimer}}(A^{-3})$</th>
<th>Diagram</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>3.93</td>
<td>3.93</td>
<td>2.358</td>
<td>0.00827</td>
<td></td>
</tr>
<tr>
<td>II</td>
<td>3.93</td>
<td>3.105</td>
<td>1.9257</td>
<td>0.01130</td>
<td></td>
</tr>
<tr>
<td>III</td>
<td>3.93</td>
<td>2.653</td>
<td>1.36</td>
<td>0.01356</td>
<td></td>
</tr>
</tbody>
</table>

Below are the results reported for the 3 different models. For comparison, the One-Molecule SC-RISM PY results are given as well as the hard sphere Monte Carlo simulation that consisted of 512 molecules. The two-molecule SC-RISM were run with all three solvation potentials (PY, HNC and MS).

5.3.1 Model I results

Because Model I consists of the same site, differing in label only but not in any properties, the results for all the $\alpha\gamma$ interaction pairs are identical. Looking at Figure 5.12 it is difficult to tell if any of the systems more closely approximates the simulation results. All of the theoretical methods underestimate the magnitude of the cusp at $r \approx 1.6\sigma$. The HNC solvation potential seems to be somewhat better than the PY.

Each two-molecule Monte Carlo simulation was run using a parallel Direct Sampling method (see Appendix B). A population of 20,000 different configurations were pregenerated
and $2^{18}$ direct sampling steps were taken. Recall that 1 sampling set constitutes going from \( r_i = \text{initial} \) to \( r_i = \text{final} \), which in this case is \( r = 1.7 \) to 30.7 Å by steps of \( \Delta r = 0.1 \) Å.

All three models appear to agree in terms of the \( r \) location of the cusps. The One-Molecule PY and Two-Molecule HNC seem to more correctly approximate the low \( r \) slope up from \( r = 1.5 \) to 1.0 where the HS interaction starts. Of particular note is how all the PRISM type theories, regardless whether One or Two-Molecule, underestimate the first cusp at \( r = 1.5 \). In this region it appears to be more favorable than average to have increased site density. Likewise, all the theories seem to underestimate the depth of the well between \( r = 1.5 \) and 2.0. Interestingly, above \( r = 2.0 \) the One-Molecule PY overestimates the radial distribution function while both two-molecule theories seem to agree more or less with the Monte Carlo simulation. One thing to note is how the Two-Chain bins the simulation data makes it appear to be non-zero very slightly near \( r/\sigma = 1.0 \). This is a result of the fact that for the index of \( r_i = 39 \), \( g(r) \) can be non-zero. For instance, if the distances being measured is \( r = 3.94 \) Å then this will not be within the hard sphere diameter of 3.93 Å, however, they will contribute a non-zero Boltzmann factor in the index of \( r_i = 39 \).

Figure 5.12: Comparison of One-Chain, Two-Chain and Monte Carlo results for Model I
5.3.2 Model II results

The results for Model II are displayed in Figure 5.13. While Model I can in principle be solved using the scalar form of the Two-Chain theory, solutions for Model II and Model III require the multi-site form of the Two-Chain theory. Model II was run with the same computational parameters as for the Model I case, but with the density, site diameters and bond lengths given in Table 5.1. Similar results to Model I can be seen in Figure 5.13. In Panel A) the RDF for the $AA$ interaction results shows that the theories mostly give similar results, with the HNC solvation potential the most accurate Two-Molecule theory particularly at lower values of $r$. In Panel B) the RDF for the $AB$ interaction has the Two-Molecule theory with PY solvation potential giving reasonably good results until close to $r/\sigma = 1$ from above, where the slope for $g(r)$ near the HS exclusion area is less than for either the Monte Carlo simulation or the other 2 theories. In panel C) the RDF for the $BB$ interaction from the theory with the PY solvation potential seems to give a lower slope near $r/\sigma = 1.0$. The HNC solvation potential gives an overall qualitatively good result for $g(r)$, but is consistently a little low in the first solvation peak. Again it appears that all theories underestimate the sharp cusps. C) shows this best at $r/\sigma = 1.3$.

5.3.3 Model III results

Up to now, there is not a great difference between the different theories for Model I and II. The results for Model III in Figure 5.14, however, do start to show some noticeable improvement of the Two-Molecule theory over the One-Molecule counterpart. The low $r/\sigma$ values show improvement and match much better, especially in panels B) and C), though the One-Molecule SC-PRISM is somewhat better beyond the first solvation peak. Panel A) again shows that the Two-Molecule theory gives results very similar in both character and quality to the One-Molecule PY results.

Overall, all the theories give similar results to the Monte Carlo simulation. Again the PY for all but Panel C) seems to have a more shallow slope when approaching $r/\sigma = 1.0$ while
Figure 5.13: Comparison of One-Chain, Two-Chain and Monte Carlo Results for Model II. The three panels A), B) and C) give the RDF for the $AA$, $AB$ and $BB$ site pairs respectively.
HNC and One-Molecule PY show good agreement with the simulation results at low $r$. One place the two-molecule theory shows improvement is in Panel B) for the $g(r)$ nearing the HS diameter for AB interactions. Here the Two-Molecule theory captures the positive slope of $g(r)$ at $r = 0$ seen in simulation, but the One-Molecule theory does not. This is again shown in Panel C) where near the BB hard sphere diameter, the shape of $g(r)$ is matched almost exactly by the two-molecule theory.

Figure 5.14: Comparison of One-Chain, Two-Chain and Monte Carlo Results for Model III. The three panels A), B) and C) give the RDF for the AA, AB and BB site pairs respectively.
5.3.4 Density range for Model I

I examine here the accuracy of the Two-Molecule theory as a function of density. In the limit of low density the two-molecule theory reduces to the explicit sampling of the bare interaction between two molecules. The Two-Molecule theory should thus be exact in the zero density limit. It is hoped that this accuracy at low density persists to higher densities. In Figure 5.15 I show the results at 4 different reduced densities: A) $\rho d^3 = 0.1$, B) $\rho d^3 = 0.2$, C) $\rho d^3 = 0.4$, D) $\rho d^3 = 0.7$. As is expected at the zero density limit shown in Panel A) the Two-Molecule theory and the simulation are in good agreement. At $1.5 < r/\sigma < 3.0$. The deviation from the simulation is likely due to the larger, 512 molecule simulation, not being run long enough rather than a problem with the Two-Chain code. Panel B) is more dense, with $\rho = 0.0018 \text{ Å}^{-3}$, compared with Panel A) with $\rho = 0.0092 \text{ Å}^{-3}$, but still shows Two-Molecule is more accurate than the One-Molecule theory at small $r$. Panel C) again shows the Two-Molecule captures the simulation results more faithfully, however, at the highest density as shown in Panel D), the One-Molecule and Two-Molecule theory are comparable in accuracy.

Summarizing the comparisons with Model I-III, the Two-Molecule theory has shown similar results to the One-Molecule theory, and in some instances better results. However, while the Two-Molecule theory does show some improvement over One-Molecule theory, the hope was for more dramatic improvement. In studies of systems similar to Model I-III, Ladani et al. argued that there are preferred orientations at higher densities due to packing that are not captured by One-Molecule RISM [90]. In approximating the solvation potential, this packing physics due to the presence of other molecules may also be lacking, though a Two-Molecule theory could in principle more explicitly capture packing, compared to a One-Molecule theory. I can then compare the results for $g(r)$ as density increases to see when the Two-Molecule theory seems to break down. For clarity, I have only displayed the results for the Two-Molecule PRISM using the MS solvation potential, which was the solvation potential that was most accurate compared to simulation. A) and B) show results with the
Two-Molecule theory matching must better especially at low $r$ values. C) again shows more consistent agreement with the simulation over the One-Molecule results especially showing the decreasing value near $r/\sigma = 1.0$ consistent with the simulation. At the highest density the two chain begins to show more obvious disagreement with the simulation.

Figure 5.15: Radial distribution functions for Model I for varying densities. A) $\rho = 0.00092 \text{Å}^{-3}$, B) $\rho = 0.0018 \text{Å}^{-3}$, C) $\rho = 0.0046 \text{Å}^{-3}$ and D) $\rho = 0.0064 \text{Å}^{-3}$.

5.4 Lennard-Jones heterodimers

I further tested the accuracy and applicability of the Two-Molecule theory by applying it to attractive asymmetric dimers. LJ dimer systems were set up based on the Model II
and Model III systems above with the LJ parameters borrowed from the \( N_2 \) system tested earlier. Model I was not studied as \( N_2 \) had already given good results. The parameters for both systems are found in Table 5.2 and the the LJ potential is defined to be

\[
U_{\text{LJ}}(r) = 4\epsilon \left( \frac{\sigma}{r} \right)^{12} - \left( \frac{\sigma}{r} \right)^{6}
\]

These systems were then compared with molecular dynamics (MD) studies as well as One-Molecule SC-PRISM PY calculations on the same system. All simulations and RISM/PRISM calculations were run at a temperature of 60\( K \).

<table>
<thead>
<tr>
<th></th>
<th>Model II</th>
<th>Model III</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \sigma_A (\text{Å}) )</td>
<td>3.930</td>
<td>3.930</td>
</tr>
<tr>
<td>( \sigma_B (\text{Å}) )</td>
<td>3.105</td>
<td>2.653</td>
</tr>
<tr>
<td>( \epsilon_A (\text{kcal/mol}) )</td>
<td>0.074</td>
<td>0.074</td>
</tr>
<tr>
<td>( \epsilon_B (\text{kcal/mol}) )</td>
<td>0.058</td>
<td>0.050</td>
</tr>
<tr>
<td>( L_{\text{Bond}} (\text{Å}) )</td>
<td>1.923</td>
<td>1.360</td>
</tr>
<tr>
<td>( \rho_{\text{Mol}} (\text{Å}^{-3}) )</td>
<td>0.0102</td>
<td>0.01249</td>
</tr>
</tbody>
</table>

It has been shown that local structure for high density systems is mostly controlled by packing constraints. Therefore, the short-ranged repulsive interactions between the sites is perhaps the most important one to consider. The long range correlations tend to be controlled by the long-ranged attractive tails. For Models II and Model III LJ, both the full LJ as well as the repulsive LJ were used. The the Lennard-Jones potential was decomposed by Weeks, Chandler and Anderson\(^91, 92\) into a repulsive LJ contribution

\[
U_{\text{Rep}}(r) = \begin{cases} 
4\epsilon \left( \left( \frac{\sigma}{r} \right)^{12} - \left( \frac{\sigma}{r} \right)^{6} \right) + \frac{1}{4} & r \leq 2^{1/6}\sigma \\
0 & r > 2^{1/6}\sigma
\end{cases}
\]

and an attractive tail contribution

\[
U_{\text{Att}}(r) = \begin{cases} 
-\epsilon & r \leq 2^{1/6}\sigma \\
4\epsilon \left( \left( \frac{\sigma}{r} \right)^{12} - \left( \frac{\sigma}{r} \right)^{6} \right) & r > 2^{1/6}\sigma
\end{cases}
\]
5.4.1 Model II LJ

As shown in Figure 5.16, the Two-Molecule theory for the repulsive LJ Model II is clearly in much better agreement with simulation than the One-Molecule SC-PRISM theory. Looking at Panels A) and B), while there are some differences, there is good overall agreement with the MD result for any of the three different solvation potentials. The HNC tends to peak slightly differently than the other methods as it is shifted to the right. It appears to be shifted in all three panels, but most dramatically in Panel A). Panel C) shows the largest different between the MD and Two-Molecule theory results. The MD results show two distinct peaks at \( r = 3.4 \) Å and \( r = 5.2 \) Å. The Two-Molecule theory shows these peaks however they are much more flattened.

Results using the full LJ Model II are shown in Figure 5.17 Again, the Two-Chain theory is doing a much better job overall than the One-Chain theory at matching simulation. The agreement between the Two-Chain theory and the MD simulation are still good, though not as good as for the repulsive case. This is especially true for plot C) where the plot is no longer flattened. Interestingly the shift to lower \( r \) is not present here for the HNC solvation potential. The full LJ results appear to not capture the magnitude of the peaks and valleys as well as in the case of the repulsive LJ model. For both Model II and Model III Full LJ there appears to be an issue where the Two-Chain results are all offset slightly. While it doesn’t change the overall results it does suggest there may be an issue with either the proper binning of results from the Two-Chain simulation or could again be a result of the the incorrect normalization addressed earlier for the HS dimers Model I-III.

5.4.2 Model III LJ

Again the Two-Chain theory is significantly better than the One-Chain theory, in comparing results for the repulsive LJ Model III, as shown in Figure 5.18. The Two-Chain theory is in overall good and much improved, though not perfect, agreement with MD simulations. Just as with the Model II repulsive LJ results, the HNC solvation potential Two-Molecule
Figure 5.16: Comparison of One-Chain, Two-Chain and Molecular Dynamics Results for Model II repulsive LJ. The three panels A), B) and C) give the RDF for the AA, AB and BB site pairs respectively.
Figure 5.17: Comparison of One-Chain, Two-Chain and Molecular Dynamics results for Model II full LJ. The three panels A), B) and C) give the RDF for the \textit{AA}, \textit{AB} and \textit{BB} site pairs respectively.
theory results appear to be slightly shifted when compared with the other models. This is again most present in plot A). Panel C) shows very good comparison with MD results except for the region $r < 4.0$ where the shoulder falls off much sooner in the Two-Molecule theory results. Again these results appear to underestimate the first peak to an extent. However, even more than for Model II, the Model III results show a large improvement over the One-Molecule theory results.

![Diagram](image)

Figure 5.18: Comparison of One-Chain, Two-Chain and Molecular Dynamics Results for Model III Repulsive LJ. The three panels A), B) and C) give the RDF for the $AA$, $AB$ and $BB$ site pairs respectively. Possible reasons for the offset of the full LJ results are mentioned previously about Model II full LJ.
Comparison of the theories and simulation for the full LJ Model III are shown in Figure 5.19. Again, there is a clear improvement from the Two-Chain theory compared to the One-Chain theory. There is again an overestimate of $g(r)$ by the theories, especially at the first peak. However, at larger $r$ values, $r > 5.0 \, \text{Å}$, the full LJ appears to approximate the MD results much tighter. Again, Panel A) shows that there is a shift in the HNC results when compared to the other two solvation potentials. In Panel C) the shoulder drops off much faster than the MD simulations near $r = 3.0 \, \text{Å}$.

Figure 5.19: Comparison of One-Chain, Two-Chain and Molecular Dynamics Results for Model III Full LJ. The three panels A), B) and C) give the RDF for the $AA$, $AB$ and $BB$ site pairs respectively.
Curiously, an average of Model III Repulsive LJ and Model III Full LJ $g(r)$ curves generates the best agreement with simulation, as seen in Figure 5.20. Likely the full LJ is too strongly attractive while the repulsive LJ suffers from having no attractive potential. The full LJ being too strong may be suffering from inaccuracies in the solvation potential to account for the steric effects of the other molecules to the correct degree. Similarly, the repulsive is not showing preference as strongly at certain distances because of the attraction, resulting in a less pronounced and more flat $g(r)$.

Figure 5.20: Comparison of One-Chain, Two-Chain and Molecular Dynamics Results for Model III, averaging $g(r)$ from the full and repulsive LJ cases. The three panels A), B) and C) give the RDF for the AA, AB and BB site pairs respectively.
5.5 Trimer systems

In this section, I test the accuracy and applicability of the Two-Molecule theory to bent and linear multi-site type trimer systems. This poses a challenge at the next level of molecular complexity. The geometrical richness of a trimer compared to a dimer also makes packing more complex, possibly suggesting an advantage for Two-Molecule theories over One-Molecule theories for accurately capturing packing.

A trimer, or three-site molecule, can have a bent or linear conformation, as illustrated below in Figure 5.21. For atomic sites, the trimer is simply a triatomic molecule. The trimer systems studied here have two site-types and are of the B-A-B topology form. A BAB trimer has two intramolecular lengths that characterize the intramolecular correlation function. As illustrated in Figure 5.21, $L$ denotes the BA bond length, which is taken to be fixed. The BB bond length is $2L - y$, where $y$ depends on the bending angle of the trimer. For a linear molecule the distance from the two B sites is $2L$, and so $y = 0$ for a linear trimer. As the trimer becomes more bent, the parameter $y$ increases from the linear value of $y = 0$.

Figure 5.21: Bent and linear trimer models having two site-types. The left panel depicts a linear trimer and the right panel depicts a bent trimer.

The Two-Chain theory for molecules with three atoms and two different sites can be solved using extensions of the techniques detailed above for dimers. As for dimers, the coefficients for the $k^{-n}$ terms need to equal zero to avoid spurious divergences due to incorporation of noisy simulation data for $h_{\alpha\gamma}(r)$. The exact $h_{\alpha\gamma}(r)$ functions would satisfy these conditions automatically, but the noisy simulation data can be “tweaked”, i.e. a minimally small correction can be added to each function value, to project the noisy data back onto
the space of functions that don’t cause divergences. This all follows the same procedure as for dimers above.

Using the bond length geometries discussed above, the intramolecular correlation function for the trimer is

\[
\hat{\omega}(k) = \begin{bmatrix}
\frac{1}{2(\sin(kL)^2)} & \frac{2\sin(kL)}{kL} \\
\frac{2\sin(kL)}{kL} & 2(1 + 2\left(\frac{\sin(kL(2L-y))}{kL(2L-y)}\right))
\end{bmatrix}
\] (5.39)

and the pair correlation expansion for the trimers is

\[
\hat{h}(k) = \begin{bmatrix}
\hat{h}_{AA}(0) + \frac{\hat{h}_{AA}(2)(0)k^2}{12} + \frac{\hat{h}_{AA}(4)(0)k^4}{360} & 2\hat{h}_{AB}(0) + \frac{\hat{h}_{AB}(2)(0)k^2}{6} + \frac{\hat{h}_{AB}(4)(0)k^4}{120} \\
2\hat{h}_{BA}(0) + \frac{\hat{h}_{BA}(2)(0)k^2}{12} + \frac{\hat{h}_{BA}(4)(0)k^4}{360} & 4\hat{h}_{BB}(0) + 2\hat{h}_{BB}(2)(0)k^2 + \frac{\hat{h}_{BB}(6)(0)k^6}{180}
\end{bmatrix}
\] (5.40)

Again using the series expansion of Equation 5.39, the \(k^{-n}\) terms and conditions that set them equal to zero can be found. For bent trimers, there is again a single \(k^{-4}\) term and then four \(k^{-2}\) terms corresponding to each respective \(\alpha\gamma\) site-type pair. Four linearly independent equations can be found by using the single \(k^{-4}\) term, and a combination of three of the \(k^{-2}\) terms.

\[
\frac{144(\hat{h}_{AA}(0) - \hat{h}_{AB}(0) - \hat{h}_{BA}(0) + \hat{h}_{BB}(0))}{k^4y^2(y - 4L)^2}
\] (5.41)

which is similar to the dimer case (see Equation 5.9 for an example of the \(k^{-4}\) term). The \(k^{-2}\) terms that need to have zero coefficient for the calculation of \(\delta\hat{C}(k)\) are rather long and not shown here. (See Appendix C.4.2 for the full set of equations used.) However, again they are similar to the dimer case in that they are functions dependent on \(\hat{h}_{\alpha\gamma}(0)\) and \(\hat{h}_{\alpha\gamma}(r)\) terms. As for dimers, the 4 linearly independent equations are solved using the scheme outlined in Figure 5.9.

### 5.5.1 Sulfur dioxide

The linear trimer case introduces new divergences, and is analyzed here. We first test the Two-Molecule theory on sulfur dioxide as a model bent triatomic. Borstnik et al. has
compared RISM calculations with Molecular Dynamics simulations of sulfur dioxide at two different temperatures and densities [93]. The sulfur dioxide was modeled as rigid molecules with atomic sites interacting via full LJ potentials. Using the trimer correction scheme derived above, I solved the multi-site Two-Molecule PRISM for SO$_2$. Table 5.3 contains parameters for SO$_2$, with the bond angle = 119.5° resulting in $y = 0.39$ Å for $\tilde{\omega}(k)$ in Equation 5.39. The One-Molecule and MD results were taken from Ref. [93].

**Table 5.3: System and Lennard Jones parameters for SO$_2$**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>$T = 350K$</th>
<th>$T = 250K$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\sigma_S$(Å)</td>
<td>3.625</td>
<td>3.625</td>
</tr>
<tr>
<td>$\sigma_O$(Å)</td>
<td>3.015</td>
<td>3.015</td>
</tr>
<tr>
<td>$\epsilon_S$(kcal/mol)</td>
<td>0.332</td>
<td>0.332</td>
</tr>
<tr>
<td>$\epsilon_O$(kcal/mol)</td>
<td>0.132</td>
<td>0.132</td>
</tr>
<tr>
<td>$L$(Å)</td>
<td>1.434</td>
<td>1.434</td>
</tr>
<tr>
<td>$\rho_{Mol}$(Å$^{-3}$)</td>
<td>0.0109</td>
<td>0.0136</td>
</tr>
</tbody>
</table>

The SO$_2$ systems studied have relatively strong LJ potentials. Finding the solution to self-consistent non-linear equations, such as those in the Two-Molecule theory, by iteration can sometimes be highly dependent on finding an initial guess which is not too far from the converged solution. Typically for these types of simulations the initial guess is $C(r) = 0.0$, which results in an initial solvation potential of $W(r) = 0.0$. With no solvation potential to the high attractive forces, the next iteration is very different from the prior iteration and the iterative search to converge to a solution fails.

To handle the strong attractions, we adopt a strategy of slowing turning on the strength of the attractive potential, and solving the Two-Molecule theory equations at intermediate stages of growing in the attraction. In this way, we can find a converged solution for a weak attraction, and then use that solution to find the solution to a slightly stronger attraction, and so on until the full attraction strength is attained. Thus, as an initial system with a weak attraction to solve, we choose to reduce the strength of the LJ potentials to 10% their full value. The scheme for converging to a solution was then run for a subsequent 300
iterations. The (partially) converged solution from this reduced LJ potential was then used as input for another 300 iterations solving a system with a 0.2 reduced LJ potential. This was repeated again, increasing the LJ strength by 0.1 factor each time until the LJ was at 100%. This scheme was able to stabilize the calculation and allow finding a converged solution. However, this could only be done for the HNC solvation model. Attempts for using the PY and MS solvation potential were unable to be completed because the $C(r) S(r) C(r)$ term at low $r$ values would become negative enough in both cases to result in negative values as arguments for the Log and square root terms of the solvation potentials, rendering them unusable.

Given the results from the density range for the Hard Sphere models where the lower density Two-Molecule theory corresponded with the simulation data better, I hypothesized that the lower density SO$_2$ at the higher temperature ($\rho = 0.0109 \text{Å}^{-3}$ and $T = 350K$) would give the better result. Those results are shown below in Figure 5.22. The Two-Molecule HNC theory is perhaps a little but not substantially better than the One-Molecule theory in matching MD, as had been the case for the lower density hard sphere cases.

The lower temperature higher density results are given in Figure 5.23. These result show a significant improvement of the Two-Molecule theory over the One-Molecule Theory. Another interesting note is that in the Model II and Model III LJ case using the full Lennard Jones tended to over emphasize the peaks. This is not the case for any of the terms here. However, for both the Two-Molecule and One-Molecule RISM cases the peaks and valleys are not nearly as pronounced, though less so with the Two-Molecule case. The MD simulation here was taken directly from [93]. Looking at this simulation data there may be reason to think the simulation could be run again. For instance, panel C) from Figure 5.22 looks to be too high above 1 for $g(r)$ and may not be normalized correctly. In future work these simulations will be run again to check for accuracy.
Figure 5.22: Comparison of One-Chain, Two-Chain and Molecular Dynamics results for sulfur dioxide for $T = 350$ K and $\rho = 0.0109 \ \text{Å}^{-3}$. A) SS, B) SO/OS, and C) OO site-site elements in $g(r)$. 
Figure 5.23: Comparison of One-Chain, Two-Chain and Molecular Dynamics Results for SO$_2$ at $T = 250$ K $\rho = 0.0136$ Å$^{-3}$. A) SS, B) SO/OS, and C) OO site-site elements in $g(r)$
5.5.2 Linear hard sphere trimers

Finally, I test the Two-Molecule theory on Linear Hard Sphere Trimers. As mentioned above, the linear geometry introduces a new divergence term, not present for the bent geometry, that needs to be controlled. This is obvious if one looks at the $k^{-4}$ term derived for the trimer series expansion (see Equation 5.41). Recall from Figure 5.21 that $y = 0$ for the linear trimer case. Setting $y = 0$ in the trimer intramolecular structure factor in Equation 5.39 results in an additional zero in the denominator in Equation 5.41. The symmetry of the molecule actually results in higher order $k^{-n}$ terms in the resulting series expansion. For the linear triatomic case there are now terms for $k^{-8}$ and $k^{-6}$ in addition to the typical $k^{-4}$ and $k^{-2}$ terms. The matrix components for the linear triatomic calculation of $\delta \hat{C}_{New}$ are much larger since they now include 4 separate $k^{-n}$ terms. These terms are again equations based on $\hat{h}_{\alpha\gamma}^{(m)}(0)$ where now the derivatives of higher order include $m = 4$ and $m = 6$ where the definition of the higher order derivatives can be found in Equation 5.18. The goal is to adjust these values to eliminate the the $k^{-n}$ divergences. Because of the additional higher order $k^{-n}$ terms there are now 13 linearly independent equations to solve. In addition, the higher order terms mean the numerical solution must be solved to higher precision. The first indexed $k$ value gives $k_1^{-8} = 3.26 \times 10^{14}$. This requires having the pseudo inverse solution find tweaking vector solutions to a much higher tolerance. Since the solutions are solved in Mathematica the precision is for all calculations set internally to be 32-digit precision to ensure the tolerance is met. This does have one drawback in that it results in much longer discrete FFT calculations.

The system tested is based on calculations by Yethiraj et al. [94] to compare different theoretical models to Monte Carlo simulations in modeling a rigid, linear, tangent hard sphere chain. As was the case for my hard sphere dimers systems, this linear system was tested over a range of densities from the zero density limit to a reduced density of $\rho d^3 = 0.0850$. In the following systems, the sites are hard spheres of identical sizes. However, the sites are designated as middle sites (M) and end sites (E). For a triatomic system there are 2 end sites
and 1 middle site. The four systems tested have their parameters listed in table Table 5.4 below.

<table>
<thead>
<tr>
<th>System</th>
<th>$\eta$</th>
<th>$d^2\rho$</th>
<th>$\rho_{Molecule}(A^{-3})$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.000</td>
<td>0.000</td>
<td>0.00000</td>
</tr>
<tr>
<td>2</td>
<td>0.200</td>
<td>0.380</td>
<td>0.00209</td>
</tr>
<tr>
<td>3</td>
<td>0.350</td>
<td>0.670</td>
<td>0.00367</td>
</tr>
<tr>
<td>4</td>
<td>0.445</td>
<td>0.850</td>
<td>0.00466</td>
</tr>
</tbody>
</table>

Solutions for this were again attempted with all solvation potentials; however, similar problems arose when using the PY and MS systems. Therefore, the only solvation potential leading to a solution was again the HNC type. Also, this system being tested is the one specifically outlined in both the sampling code and Mathematica Two-Molecule numerical scheme in Appendix C.1 and C.2 respectively. The equations from the Taylor expansion used for correcting the simulation data are included in Appendix C.4.3.

5.5.3 Testing density range

Earlier for the initial overlapping hard sphere Models I - III, I showed how at lower densities the Two-Molecule theory outperformed the One-Molecule results for the Model I system. Yethiraj et al. also looked at a density range for their different theoretical methods to compare with One-Molecule SC-RISM PY. Again the zero density limit is identical to the zero density limit of the Monte Carlo simulation. The One-Molecule theory tends to only agree at long range $r$. Also, the One-Molecule results do not always get the $g(r)$ cusps in the correct place (see Figure 5.24 A).

Increasing the density, the Two-Molecule theory results still show overall improvement over the One-Molecule results. Once again at low $r$ values, the Two-Molecule theory results match the simulation $g(r)$ much more accurately. Figure 5.25 B) does show more deviation for the Two-Molecule theory than the previous results, however there is still better overall agreement with the Monte Carlo simulation than the One-Molecule theory. The
Figure 5.24: Linear Trimer: Comparison of One-Chain, Two-Chain and Molecular Dynamics results for $\eta = 0.0$: A) End to End, B) End to Middle, C) Middle to Middle site-type correlations. The Two-Molecule results are found using the HNC solvation potential.
One-Molecule results show a more sharp upturn at low $r$ whereas the simulation is fairly flat. The Two-Molecule theory does show an upturn but it is much less steep. The same is true for Figure 5.25 C) where the One-Molecule results over estimate the low $r$ results.

Figure 5.25: Linear Trimer: Comparison of One-Chain, Two-Chain and Molecular Dynamics Results for $\eta = 0.2$ : A) End to End, B) End to Middle, C) Middle to Middle site-type correlations

An interesting results starts to show up at $\eta = 0.35$, as shown in Figure 5.26. At the lowest $r$ values, the One-Molecule results ($r/\sigma < 1.5$) are slightly better at approximating the slope and magnitude of $g(r)$, however, some subtle structure at low $r$ is starting to emerge in the Monte Carlo simulation. For both End-End and End-Middle correlations in Figure 5.26 A) and B) respectively, a slight peak at $r/\sigma \approx 1.7$ can be seen. The Two-Molecule theory
does capture this peak, if slightly larger than the simulation results for the End-Middle. However, there is a complete lack of these peaks in the One-Molecule results. Figure 5.26 D) is a zoom in of Figure 5.26 B) for the End-Middle pair distribution functions.

Figure 5.26: Linear Trimer: Comparison of One-Chain, Two-Chain and Molecular Dynamics Results for $\eta = 0.35$: A) End to End, B) End to Middle, C) Middle to Middle site-type correlations.

In Figure 5.27 A) and B) there is even more distinct structure in this most dense system tested. The One-Molecule theory results do not capture in any way these secondary peaks near $r/\sigma \approx 1.7$.

Figure 5.28 shows the Middle-Middle $g(r)$ results. Here there is good agreement for both the One and Two-Molecule RISM/PRISM results and the Monte Carlo simulation. The
Figure 5.27: Linear Trimer: Comparison of One-Chain, Two-Chain and Molecular Dynamics Results for $\eta = 0.445$ A) End to End, B) End to Middle, C) End to End zoomed in, D) End-Middle zoomed in,
zoomed in versions of Panels A) and B) are seen in Panels C) and D respectively. The Two-Molecule theory clearly does a much better job at matching all the cusps seen in the Monte Carlo simulation. However, the simulation peaks tend to be a little more broad and the Two-Molecule theory results are more sharp. The One-Molecule results do not show all the peaks and also seem to overestimate some of the peak heights.

![Graph of g(r) vs r/σ](image)

**Figure 5.28:** Linear Trimer: Comparison of One-Chain, Two-Chain and Molecular Dynamics Results for \( n = 0.45 \): Middle to Middle

### 5.6 Summary

A new method has been developed to modify simulation results to allow for multi-site two-molecule SC-PRISM. This method involves adding a small “tweaking” vector to raw simulation data to remove \( k^{-n} \) divergence terms in the calculation of the direct correlation function in a manner that satisfy sets of equations to project out nonphysical noise. The method also uses the pseudo inverse in determination of the “tweaking” vector values to ensure the simulation data is modified in the smallest amount possible.
The initial test for the new method was applied to a hard sphere homonuclear diatomic where the $A$ and $B$ sites were identical. The symmetry of this system allowed it to be directly compared with the one-site scalar two-molecule SC-PRISM. Molecular nitrogen was then tested as it also included attractive forces from the LJ potential. Again the $g(r)$ for the multi-site system compared well with the scalar calculations.

Models I-III for hard sphere systems and Models II and III for LJ did not show large variation when comparing solvation potential choice. When there were differences there was not one of the 3 potentials that was consistently better than the other models. For the athermal Models I-III the Two-Molecule typically gave slightly better results than the One-Molecule but not in every case. However, Model’s II and III were modified to include Lennard-Jones potentials and calculations for both repulsive and full LJ showed significant improvement over the One-Molecule PY results.

Two different trimer systems were tested. The first were two different sulfur dioxide systems. For both systems the Two-Molecule the $SO_2$ calculations showed improvement over One-Molecule. This was especially true for the lower temperature higher density ($T = 250 \text{ K, } \rho = 0.0136 \text{ Å}^{-3}$) system.

Linear trimers required higher order $k^{-n}$ terms to solve. This resulted in solutions needing to be solved to higher tolerances. The Two-Molecule calculations were much more accurate than the One-Chain when compared with the Monte Carlo simulation at all tested densities. Especially impressive was the linear trimer Two-Molecule RDF results for volume fractions $\eta = 0.35$ and $\eta = 0.445$. Subtle $g(r)$ peaks were captured by the Two-Molecule method and were completely absent in the One-Molecule.
CHAPTER 6
TWO-MOLECULE RISM WATER

Up to this chapter, I have covered athermal (hard sphere) and Lennard Jones systems that involve either no or moderate attractive potentials. Systems that involve ion solvation or hydrogen bonding are very relevant for study but more difficult because of the strong potentials present in them. Here I extend my multi-site Two-Molecule theory to work with strong Coulomb potentials.

6.1 Coulomb potential and appropriate divergence

Stillinger and Lovett, in 1968, showed that the dielectric constant can be defined in terms of the total correlation functions for single atom ionic solutions [95, 96]. Later Mitchel et al. [97] derived this relation rigorously. They both found that electrolyte solutions obey the following relation:

\[-\frac{3\epsilon}{2\pi \beta} = \sum_{\alpha \gamma} z_\alpha z_\gamma \rho_\alpha \rho_\gamma \int_0^\infty h_{ij}(r)4\pi r^4 dr\]  

(6.1)

where \( z_i \) is the charge on species \( i \) and \( \rho_i \) is the concentration of species \( i \), \( \epsilon \) is the dielectric constant and \( \beta = \frac{1}{k_B T} \). They started first by looking at simple systems of atomic ions. I have covered the integral equation treatment of atomic systems using the OZ relation in Equation 3.2, which can be naturally rewritten in \( k \)-space as

\[\hat{h}(k) = [I - \hat{C}(k)]^{-1}\hat{C}(k)\]  

(6.2)

At long range the direct correlation function is assumed to be proportional to the coulomb potential, that is to say \( C_{ij}(r) \approx -\frac{\beta z_i z_j}{\epsilon r} \) for \( r \to \infty \) [97]. Using this asymptotic expression in the Fourier transform of the direct correlation function

\[\hat{C}_{\alpha\gamma}(k) = 4\pi(\rho_\alpha \rho_\gamma)^{1/2} \int_0^\infty C_{\alpha\gamma}(r)r^2 \frac{\sin(kr)}{kr} dr\]  

(6.3)

gives in the limit as \( k \to 0 \)
$$\hat{C}_{\alpha\gamma}(k) \approx -\frac{4\pi\beta z_\alpha z_\gamma (\rho_\alpha \rho_\gamma)^{1/2}}{k^2 \epsilon}$$

(6.4)

For the atomic electrolyte system there is thus an expected \( k^{-2} \) divergence for the direct correlation function at small \( k \). This is an interesting result in the context of the methods developed in the prior chapters to remove any divergence with \( k \) to a negative power. The question then becomes if this divergence is also expected for the case of charged molecular systems, and if so, what equations need to be satisfied to control the divergence that is analogous to the Stillinger-Lovett rules found in Equation 6.1 while suppressing spurious divergent behavior due to numerical simulation noise.

Using linear response theory, Chandler derived an expression relating the dielectric constant of the medium in terms of sums of the correlation functions for flexible molecules [98].

$$\left( \frac{1}{\epsilon} - 1 \right) = \lim_{k \to 0} \frac{4\pi\beta}{k^2} \sum_{\alpha\gamma} z_\alpha z_\gamma \hat{S}_{\alpha\gamma}(k)$$

(6.5)

Here, \( \hat{S}_{\alpha\gamma}(k) \) is the structure factor and is defined previously in Equation 3.16 and \( z_\alpha \) is the charge on site type \( \alpha \). This expression relates the \( k^{-2} \) coulomb divergence of the direct correlation function to the inter and intra-molecular correlation functions as well as the dielectric constant. Using this definition for the dielectric constant means that the expected divergence can be controlled by correctly estimating the dielectric from the correlation functions. \( \hat{S}_{\alpha\gamma}(k) \) can be expanded in a Taylor series in \( k \) :

$$\sum_{\alpha\gamma} z_\alpha z_\gamma \hat{S}_{\alpha\gamma}(k) = \frac{k^2}{2} \sum_{\alpha\gamma} z_\alpha z_\gamma \hat{S}_{\alpha\gamma}^{(2)}(0) + \ldots$$

(6.6)

Here, the value for the zero wave vector second derivative of the structure factor can be found by solving the integral for the second-moment in terms of the correlation functions. Using the definition in Equation 3.16, the structure factor can be determined. The correction scheme used for the Two-Molecule RISM theory will be very similar to that presented in Chapter 5 in that minimal modifications or “tweaks” to the simulation data are made in order to correct
the total correlation functions to eliminate unphysical behavior resulting from numerical noise of the simulation data. However, while the $k^{-4}$ correction will be done exactly as for the bent trimer derivation, since we will be considering water, the other $k^{-2}$ divergence terms will not be included since this divergence is expected and not to be eliminated. Rather, the amplitude of the $k^{-2}$ divergence will need to be controlled to correspond to a dielectric constant. In previous work from Chapter 5, the Two-Molecule simulation was done over a limited range, and then a tail was pieced on from the One-Molecule SC-PRISM results. For the Coulomb potential, as shown above, the long range $r$ dependence of the real space correlation functions is important in correctly estimating the dielectric constant. In this way the simulation data $h_{\alpha\gamma}(r)$ will be corrected for both simulation noise as well as expected long range behavior.

To determine the long range behavior character I start with the radial distribution function, which can be defined as

$$g(r) = e^{-\psi(r)}$$

where $\psi(r)$ is the potential of mean force. A series expansion for the exponential yields

$$g(r) = 1 - \psi(r) + ...$$

Redefining in terms of $h(r)$ leads to

$$h(r) \approx -\psi(r)$$

At long range, I can approximate the potential of mean force in terms of an effective potential, $U(r)$, acting on two molecules at long range in a dielectric medium. This estimation will depend on the systems being studied. Once I determine the form that the potential will take at long range, the simulation data can be fit and the tail added similarly to how the initial solution for the Two-Molecule problem was solved at the end of section 5.1. There will be a least squares fit over the appropriate data range as well as the addition of a tweaking vector. (See Figure 6.1 for the tail fitting range and further explanation.) For a long range water potential of mean force, the water molecules can be treated as two permanent, freely
rotating, dipoles interacting in a dielectric medium. This type of system of permanent fixed
dipoles corresponds to the Keesom potential [99].

\[ U(r) = 4\epsilon((\sigma_r)^{12} - (\sigma_r)^6) - \frac{\mu_1^2\mu_2^2}{3(4\pi\epsilon_0^2)k_BTr^6} \]  \hspace{1cm} (6.9)

where \( \mu_i \) are the respective dipole moments, \( \epsilon_0 \) is the permitivity of free space, and \( \sigma \) is
the diameter of the interacting sites. At large distances the LJ portion of the potential is
basically zero so the potential can be approximated with the following expression.

\[ U_{\text{Keesom}}(r) = -\frac{\mu_1^2\mu_2^2}{3(4\pi\epsilon_0^2)k_BTr^6} \]  \hspace{1cm} (6.10)

So the long range \( r \) behavior for \( h(r) \) will be approximated by the Keesom potential, asymptotically behaving as \( r^{-6} \). The solution method will work just as before, in terms of most of
the method for fixing the simulation data involve the tweaking of simulation points; however
now, the \( k^{-2} \) divergence will be controlled to be in accordance with the dielectric constant.
In addition, the tails for the \( h(r) \) functions will be fit to potentials with the form of Equation
6.10 that will be approximated as functions with \( r^{-6} \) behavior.

The construction of the matrix for using the pseudo inverse is done in a similar manner,
corresponding to the equations below. The first row is just the equation that must be
satisfied for bent trimers (see Appendix C.4.2 for full derivation) to avoid the \( k^{-4} \) divergence
found in Equation 5.41. The second row is the dielectric condition for the \( k^{-2} \) divergence
from Equations 6.5 and 6.6. The last four equations are from the least squares fit for the
tail region

\[ \hat{h}_{OO}(0) - \hat{h}_{OH}(0) - \hat{h}_{HO}(0) + \hat{h}_{HH}(0) = 0 \]  \hspace{1cm} (6.11)
\[ z_O z_O \hat{S}^{(2)}_{OO}(0) + z_O z_H \hat{S}^{(2)}_{OH}(0) + z_H z_O \hat{S}^{(2)}_{HO}(0) + z_H z_H \hat{S}^{(2)}_{HH}(0) = - \left( \frac{1}{4} - 1 \right) \frac{1}{4 \pi \beta} \]  \hspace{1cm} (6.12)
\[ 2 \sum_{i=FS}^{FP} (\hat{h}_{OO}(r_i) - U^{A}_{OO}(r_i)) = 0 \]  \hspace{1cm} (6.13)
\[ 2 \sum_{i=FS}^{FP} (\hat{h}_{OH}(r_i) - U^{B}_{OH}(r_i)) = 0 \]  \hspace{1cm} (6.14)
\[ 2 \sum_{i=FS}^{FP} (\hat{h}_{HO}(r_i) - U^{C}_{HO}(r_i)) = 0 \]  \hspace{1cm} (6.15)
\[ 2 \sum_{i=FS}^{FP} (\hat{h}_{HH}(r_i) - U^{D}_{HH}(r_i)) = 0 \]  \hspace{1cm} (6.16)
Here \( U^{x}_{\alpha\gamma}(r) \) is the potential to be fit at large \( r \) to approximate the potential of mean force. The superscripts \( x = A, B, C \) and \( D \) are the parameters to be fit. For water, I define these potentials as \( Ar^{-6}, Br^{-6}, Cr^{-6} \) and \( Dr^{-6} \). The form of \( r^{-6} \) is in keeping with the form of the Keesom potential. Here \( h_{\alpha\gamma}(r) \) is defined as \( h_{\alpha\gamma}(r_{i}) = h_{\alpha\gamma}(r_{i}) + \{q_{S_{\alpha\gamma}}, q_{S_{\alpha\gamma}+1} , ... , q_{P_{\alpha\gamma}}\} \). 

\( FS \) and \( FP \), in the above relations, refer to the index of \( h(r_{i}) \) simulation data that are going to be fit. These fitted values will also have a tweaking value included. This ensures that the tail region is fit smoothly with the simulation data. Also, the terms \( \hat{h}_{\alpha\gamma}(0) \) will have to be slightly modified from their definition in Equation 5.23. Previously the values for \( \hat{h}_{\alpha\gamma}(0) \) were generated by including the unmodified \( \hat{h}_{\alpha\gamma}(0) \) which can be found from doing the numerical integration using the appropriate definition of the zero wave vector integral using Equation 5.19 and the respective equation from 5.18 with the additional portion due to the “tweaking” \( q \) vector, which for the trapezoidal integral definition used is \( \Delta r \sum_{i=S_{\alpha\gamma}}^{P_{\alpha\gamma}} (4\pi)(q_{i})r_{i}^{2} \). It was this portion that would end up giving the matrix elements to solve for the \( q_{i} \) values using the pseudo inverse. However, this method, when applied previously to the HS system, relied on using the tail portion from the One-Molecule RISM, whereas now the tail will be fit to a potential by a single or multiple parameter fits. For the case of water there will only be a single parameter to fit. This requires the integral definition of \( \hat{h}_{\alpha\gamma}(0) \) to be expanded to an additional term for the tail fitting parameters. Below \( TS \) is defined as the index where the fitted tail elements will be grafted on.

\[
\hat{h}_{\alpha\gamma}(0) = 4\pi \frac{\Delta r}{2} (h_{\alpha\gamma}(r_{1})r_{1}^{2}) + 2 \sum_{i=2}^{TS-1} h_{\alpha\gamma}(r_{i})r_{i}^{2} + \Delta r \sum_{i=S_{\alpha\gamma}}^{P_{\alpha\gamma}} (4\pi)(q_{i})r_{i}^{2} + \Delta r (4\pi)(\sum_{i=TS}^{N_{grad}} U^{x}_{\alpha\gamma}(r_{i})r_{i}^{2} + \frac{1}{2} U^{x}_{\alpha\gamma}(r_{N_{grad}})r_{N_{grad}}^{2})
\]

(6.17)

Again, the other higher derivative terms of \( \hat{h}_{\alpha\gamma}(0) \) can be found in a similar manner using their definitions in Equation 5.18. The definition of \( \hat{S}^{(2)}_{\alpha\gamma}(0) \) is found using Equations 3.16, 3.13 and 3.14:

\[
\hat{S}^{(2)}_{\alpha\gamma}(0) = \rho_{\alpha}\rho_{\gamma}\hat{h}^{(2)}_{\alpha\gamma}(0) + \rho_{\alpha}\hat{\omega}^{(2)}_{\alpha\gamma}(0)
\]

(6.18)
The intramolecular structure factor term, \( \rho_0 \hat{\omega}_{\alpha \gamma}^{(2)}(0) \), can be solved analytically for rigid molecules or determined from simulation data. For simulated data the statistics are considerably better than the \( g(r) \) because the size of the molecule is limited in size and so can be sampled more accurately. The result is that Equation 6.12 can be reduced instead to satisfying the dielectric conditions based only on the \( \rho_0 \rho_\gamma \hat{h}_{\alpha \gamma}^{(2)}(0) \) terms. The equations are now set up such that I can define the matrix and vector elements to solve the problem using the pseudo inverse method as before. First, I will define the RHS equations for the 6 equations (6.11 through 6.16).

\[
RHS_1 = -\left( \sum_{O,H} 4\pi \frac{\Delta r}{2} (h_{\alpha\gamma}(r_1)r_1^2 + 2 \sum_{i=2}^{TS-1} h_{\alpha\gamma}(r_i)r_i^2) \right)
\]

\[
RHS_2 = -\frac{1}{2} \left( \sum_{O,H} \rho_\alpha \rho_\gamma \left( \frac{-1}{3} \right) (4\pi \frac{\Delta r}{2} (h_{\alpha\gamma}(r_1)r_1^4 + 2 \sum_{i=2}^{TS-1} h_{\alpha\gamma}(r_i)r_i^4) \right)
\]

\[
RHS_3 = -2 \sum_{i=FS}^{FP} h_{OO}(r_i)
\]

\[
RHS_4 = -2 \sum_{i=FS}^{FP} h_{OH}(r_i)
\]

\[
RHS_5 = -2 \sum_{i=FS}^{FP} h_{HO}(r_i)
\]

\[
RHS_6 = -2 \sum_{i=FS}^{FP} h_{HH}(r_i)
\]

The matrix is constructed similarly to Equation 5.30 earlier for the non-Coulombic systems.

\[
L_{\alpha\gamma} = -2 \sum_{i=FS}^{FP} r_i^{-6}
\]

The matrix entries are also defined for where the tweaking indexed values overlap with the tail fitting indices. Each row will have zeros for the most part but will have the value of 2 from the linear fit (see Equations 6.13 - 6.16). The values \( ZRO_{\alpha\gamma} \) will indicate the matrix elements are a row of zeros associated with the tweaking indexes \( S_{\alpha\gamma} \) and \( P_{\alpha\gamma} \) where the
length of each respective ZRO\(\alpha\gamma\) row is \(P_{\alpha\gamma} - S_{\alpha\gamma}\). (Note here this \(S_{\alpha\gamma}\) is defined in Chapter 5 and is not related to the structure factor above.) The term \(F_{\alpha\gamma}\) below will indicate where the value 2 will be for the indices that have both an associated \(q_i\) but are also used in the least squares fit for the tails. The subscript here for \(O_i\) and \(2_i\) is merely used to indicate its place in the row. The actual location in the matrix would have to be adjusted depending on the relative lengths of the sub-rows defined with \(P_{\alpha\gamma}\) and \(S_{\alpha\gamma}\)

\[
F_{\alpha\gamma} = \{0_1, 0_2, \ldots 2_{FS}, 2, \ldots 2_{FP}\}
\]  

\[
\begin{array}{cccccccc}
\Delta r 4\pi r^2_{S_{OO}} & -\Delta r 4\pi r^2_{S_{OH}} & \ldots & \Delta r 4\pi r^2_{S_{HH}} & T_A & T_B & T_C & T_D \\
-(4\Delta r^3 \pi)\rho_{O}z_{O}^2 r^4_{S_{AA}} & -(4\Delta r^3 \pi)\rho_{O}z_{O}^2 r^4_{S_{SOH}} & \ldots & -(4\Delta r^3 \pi)\rho_{H}z_{H}^2 r^4_{S_{SHH}} & T_{A}^{(2)} & T_{B}^{(2)} & T_{C}^{(2)} & T_{D}^{(2)} \\
F_{OO} & ZRO_{OH} & \ldots & ZRO_{HH} & L_{OO} & 0 & 0 & 0 \\
ZRO_{OO} & F_{OH} & \ldots & ZRO_{HH} & 0 & L_{OH} & 0 & 0 \\
ZRO_{OO} & ZRO_{OH} & \ldots & ZRO_{HH} & 0 & 0 & L_{HO} & 0 \\
ZRO_{OO} & ZRO_{OH} & \ldots & F_{HH} & 0 & 0 & 0 & L_{HH} \\
\end{array}
\]  

The values of \(T_x\) and \(T_x^{(2)}\) are merely the integral values of the tails and defined below

\[
T_x = \Delta r 4\pi \left( \sum_{i=TS}^{N_{Grid}-1} r_i^{-6} r_i^2 + \frac{1}{2} r_{N_{Grid}}^{-6} r_{N_{Grid}}^2 \right)
\]

\[
T_x^{(2)} = -\Delta r^4 3\pi \left( \sum_{i=TS}^{N_{Grid}-1} r_i^{-6} r_i^4 + \frac{1}{2} r_{N_{Grid}}^{-6} r_{N_{Grid}}^4 \right)
\]

The system is then set up in the same way as Equation 5.32, however, the \(q\) tweaking vector has 4 additional terms for the curve fitting. The length of \(Q\), and the number of columns in the matrix will be

\[
Q = (P_{AA} - S_{AA}) + (P_{AB} - S_{AB}) + (P_{BA} - S_{BA}) + (P_{BB} - S_{BB}) + 4
\]

The new “tweaking” vector \(q\) is defined below:
Figure 6.1: Diagram of “tweaking” vector and tail fit for Coulombic potentials for use with Two-Molecule RISM.

6.1.1 Results for water

The One-Molecule and Two-Molecule theories and MD simulation of SPC model was run using the parameters found in Table 6.1. Just as was done for the sulfur dioxide system
where the LJ was fairly strong, the water system must have the solvation potential grown in slowly. To do this, the Coulomb potential is initially turned down to only 5% of its strength. The Two-Molecule RISM is then run for 200 iterations. The potential is increased to 10% and run for 200 steps again. This is continued where the potential is increased by 5% until it is at 100% at which point it is run until convergence. The HNC solvation potential was used in these calculations.

Table 6.1: Parameters for SPC and SPC/E Models

<table>
<thead>
<tr>
<th></th>
<th>SPC</th>
<th>SPC/E</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\rho_{H_2O}(\text{Å}^{-3}))</td>
<td>0.0333</td>
<td>0.0333</td>
</tr>
<tr>
<td>Temp. (K)</td>
<td>298</td>
<td>298</td>
</tr>
<tr>
<td>(\sigma_O(\text{Å}))</td>
<td>3.166</td>
<td>3.166</td>
</tr>
<tr>
<td>(\epsilon_O(\text{kcal/mol}))</td>
<td>0.15535</td>
<td>0.15535</td>
</tr>
<tr>
<td>(L_{Bond})</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>H-O-H Angle</td>
<td>109.47</td>
<td>109.47</td>
</tr>
<tr>
<td>(q_O(e))</td>
<td>-0.8200</td>
<td>-0.8476</td>
</tr>
<tr>
<td>(q_H(e))</td>
<td>0.4100</td>
<td>0.4238</td>
</tr>
</tbody>
</table>

My Two-Molecule RISM results are compared to work done by Lue et al. [100] calculating SPC water results using the One-Molecule RISM method with an HNC closure as well as molecular dynamics simulations in Figure 6.2. The first thing to notice is that for all the \(g(r)\) functions, the peak location is correctly predicted by the Two-Molecule theory while the the One-Molecule theory tends to have the peaks shifted slightly. Neither scheme gets all the relative peaks height correctly, but the Two-Molecule theory does an overall better job. Panel A) shows that both the One- and Two-Molecule theories lack the weak peak near 4.5 Å.

Overall the Two-Molecule RISM method gives better results than that of the One-Molecule RISM method. In Figure 6.3, I show the divergent \(\hat{C}(k)\) results. These results appear to be well controlled by using the correlation function definition for the dielectric in Equation 6.5. The divergence stays very constant in its overall magnitude of O(10^7) and does not grow uncontrollably with each subsequent iteration.

136
Figure 6.2: Comparison of MD, One-Molecule and Two-Molecule RISM for the SPC model of water. A) OO, B) OH/HO, C) HH site-site distribution functions
Figure 6.3: $\hat{C}(k)$ results showing the controlled divergence using the dielectric and $h(r)$ sum rules. Both A) and B) are of the same plot with B) being zoomed in.

### 6.2 Summary

The “tweaking” vector method was modified to work for charged systems with coulomb potentials. The method involved only solving the equations for projecting out the $k^{-4}$ divergent terms. All $k^{-2}$ were left alone as it is expected that for strongly interacting systems, $\hat{C}(k)$ will have a small $k^{-2}$ divergence. Instead, the $k^{-2}$ divergence is controlled using an additional the $h(r)$ definition of the dielectric for molecules. Along with the “tweaking” vector, a fitted tail was included to more accurately reflect the long range behavior of the water molecules. Two-Molecule water calculations were more accurate than One-Molecule at finding $g(r)$ peak locations as well as peak magnitude. Even more impressive is that long range, strong attractions were able to be handled properly by RISM by using this new multi-site method.
CHAPTER 7
CONCLUSIONS AND FUTURE WORK

In the first part of this thesis, surface segregation in blends of short cyclic and linear polymers were studied theoretically and compared with experiment. A new cyclic polymer model based on a linear non-overlapping freely jointed chain model was developed. A series of Wall-PRISM calculations were done using a blend of the cyclic and linear polymers. It was theorized that packing effects were more important than conformational entropy reductions for short chains, explaining the experimentally observed enrichment of short linear chains at the surface compared to the bulk. In agreement with NR experiments, Wall-PRISM calculations predicted that linear chains are enriched directly at the surface, and have a higher net integrated excess. Further studies were done to consider surface segregation as a function of system density. For all densities, the linear chains were preferred directly at the surface. For most densities, the net integrated excess of linear chains was positive (linear enriched), but for the lowest densities, the cyclic chains’ integrated excess was net positive. This was likely due to the smaller size of cyclic chains allowing them to more closely approach the wall under dilute conditions. Further surface segregation studies were conducted on the effect of the wall stiffness. The surface potential was adjusted to allow the polymer chains to penetrate to varying degrees. A soft wall may better approximate the effective potential felt by a monomer at a free surface. These “soft” Wall-PRISM calculations showed a slight decrease in integrated excess of linear chains with increasing softness, as packing effects were reduced at a softer surface.

Since only a single size of polymer chain was considered, future work could involve looking at larger sizes of chains. The method outlined here might not continue to match experiment as well because it has been shown that at larger chain sizes the configurational entropy dominates in determining surface segregation, as evidenced by both experiment and SCFT.
calculations. Currently the chain models used in the Wall-PRISM calculations are the same for the bulk as well as near the surface. Modifications to the current models could be made to give a more realistic model of the polymer chains in a more confined geometry. There are however some limitations to this model, as the intramolecular correlation functions for the polymers are the same as used for the bulk. It has been shown through MC simulation that packing is important but configurational entropy cannot be ignored and is more pronounced at lower densities because of the reduction of packing effects[42]. Likewise, the conformational entropy changes could be captured by moving to a classical density functional theory.

I developed a multi-site Two-Molecule RISM theory, a hybrid simulation/theory approach that is more accurate than pure theory and faster than pure molecular simulation. Instead, a simulation of two chains of interest allows more realistic determination of molecular correlation functions at small distances, with RISM theory providing the influence of the rest of the molecules. Key to the implementation of the Two-Molecule theory is a new method I developed for correcting simulation noise for use in Two-Molecule RISM theory. This new method involves using $h(r)$ sum rules to generate a vector of small “tweaking” adjustments to correct the divergences caused when solving RISM/PRISM equations using simulation data. The method was tested on various small molecule systems of both athermal and Lennard-Jones type. Overall, the Two-Molecule theory results showed improvement over the current state-of-the-art single chain SC-PRISM method. The improvements tended to be most significant at small $r$, showing improved physics for two molecules interacting. This new method was also expanded to include Coulomb potentials. The Coulomb potentials were corrected with not only the “tweaking” adjustments but also the long range $r$ portion of the radial distribution function was modeled on a Keesom potential rather than using the previously used method of stitching the tail region on from One-Molecule SC-PRISM solutions. This tail region was fit to control the expected $k^{-2}$ divergence of the direct correlation function for a system with Coulomb potentials. An advantage of the Two-Molecule
RISM hybrid/simulation approach is that the simulation portion is trivially parallelizable. In addition to the theoretical developments, I also parallelized the direct sampling Monte Carlo method to be used for the Two-Molecule theory and to decrease overall computation time.

One area of concern for the Two-Molecule method over traditional simulations is that it relies on a solvation potential to faithfully represent the effect of other molecules in the solution. There are instances where the Two-Molecule theory is only marginally better than the One-Molecule theory, and likely this is because of deficiencies in the solvation potential approximation. In deriving the solvation potential approximation additional, higher order terms, can be included. One of these terms is the bridge function, \( d_{ij}(r) \), defined as

\[
d_{ij}(r) = \ln(h_{ij}(r) + 1) + \beta u_{ij}(r) - h_{ij}(r) + C_{ij}(r)
\] (7.1)

where \( u_{ij}(r) \) is the pair potential and \( h(r) \) and \( C(r) \) are the total correlation function and direct correlation function. In the HNC approximation, the bridge function is set to equal zero. However, recent work by Chuev et al. [101] has shown marked improvement for the One-Molecule RISM theory for water when the bridge functions from MD simulations of water are included. As a first step toward improving the theory, the bridge function from either MD simulation results and calculated from the Two-Chain simulation will be calculated for hard sphere, LJ and water systems and included with the calculation of the solvation potential.

Future work for the multi-site Two-Molecule RISM/PRISM theory will also involve moving to larger, flexible molecules and polymers. Likely a first step will be to look at simple multi-site systems such as isotactic polypropylene as well as expand to looking at polyolefin blends. The miscibility and blend thermodynamics of polyolefin blends is known to depend sensitively on packing effects. Currently PRISM studies of blends have the two different polymer types interacting only through the solvation potential. With the ability to now do multi-site Two-Chain SC-PRISM, the two chains will be allowed to interact explicitly and better capture packing effects. This will hopefully allow for more accurate property predic-
tion for polymer blends compared to One-Molecule SC-PRISM while still being more efficient computationally than MD. Lastly, a major motivation in developing the Two-Chain theory is to study $PEO/Li^+$ solid polymer electrolyte systems typically associated with batteries. By utilizing the efficiency of Two-Chain SC-PRISM theory, one can generate configurations of complexed $Li^+$ in PEO chains to determine the free energy of complexation and, from that, the rates of “hopping” of $Li^+$. The large size, long interaction lengths of charged species and long computational time involved in capturing interchain hopping have made study of these important systems limited. Future work with the newly enhanced Two-Chain method will involve study of ion solvation and complexation by polymers as well as study of the effect of polymer conformations on ion complexation as well as the determination of rates of interchain diffusion in solid polymer electrolytes. Finally, further work in improving sampling methods with parallelization will further increase the efficiency of current Two-Chain SC-PRISM.
REFERENCES CITED


sphere mixtures: Monte Carlo simulation and integral equation results. *Molecular


[67] John G. Curro, Jeffrey D. Weinhold, John D. McCoy, and Arun Yethiraj. The
structure of amorphous polymers near surfaces: athermal systems. *Computational
and Theoretical Polymer Science*, 8(1-2):159–168, jan 1998. ISSN 10893156. doi:
article/pii/S1089315698000294.

[68] John G Curro. Intermolecular Structure and Thermodynamics of Vinyl Polymer Liq-

for polyolefins: Comparison to molecular dynamics simulations and x-ray scattering.
*The Journal of Chemical Physics*, 114(6):2847, 2001. ISSN 00219606. doi:

[70] David Chandler, Yashwant Singh, and Diane M. Richardson. Excess electrons in simple
http://scitation.aip.org/content/aip/journal/jcp/81/4/10.1063/1.447820.

Excess electrons in simple fluids. II. Numerical results for the hard sphere solvent.
*The Journal of Chemical Physics*, 81(11):5109, 1984. ISSN 00219606. doi:

[72] Daniel Laria, David Wu, and David Chandler. Reference interaction site model polaron

[73] K. S. Schweizer and J.G. Curro. Integral Equation Theories of the Structure, Thermo-
dynamics, and Phase Transitions of Polymer Fluids. In I. Prigogine and Stuart A. Rice,


APPENDIX A - WALL-PRISM

This section of the Appendix includes the code for calculating the linear and cyclic polymer intramolecular correlation functions as well as the actual Wall-PRISM calculations.

A.1 Non-overlapping freely jointed chain calculations

Here is the code for calculating both the linear and cyclic chains.

First is for the linear chain

Listing A.1: Non-overlapping Freely Jointed Chain Linear

(* define grid parameters *)
ngrid=2048;
delr=4/100;
delk=Pi/delr/ngrid;
rvalues=Table[(i-1/2)*delr,{i,1,ngrid}] ;
kvalues=Table[(i-1/2)*delk,{i,1,ngrid}] ;
(*omega tau*)
wtauid3[r,\[Tau]] := 1/(2^((\[Tau]+1)\[Pi])((\[Tau]-2)!))!

(*UnderoverscriptBox[(\[Sum]](s=0),\[Tau]-2,s)Binomial[\[Tau],s]UnitStep[\[Tau]-2-m]*

(*SuperscriptBox[(\[Sum]](m=0),\[Tau]-2,m)Binomial[\[Tau],m]*

(*20 unit monomer so n = 20*)
n=20;
omegaideald=Table[Table[N[wtauid3[(i-1/2)*delr,j],32],{i,1,ngrid}],[j,2,n-1]];
{20.5614,Null}

omegardno=Table[Join[Table[0,{25}],Drop[omegaideald[[i]],25]],{i,1,Length[omegaideald]}];

omegakd=Table[(2*Pi)*delr*Sqrt[2*ngrid*FourierDST[N[omegardno[[i]]]*rvalues,60],4])/kvalues,\{i,1,Length[omegaideald]\}];
(*finding the normalization constant by integrated the r space version. It will probably give errors about asking for values out of range *)

154
norm=Table[Interpolation[Table[{kvalues[[j]],omegakd[[i,j]]},{j,1,ngrid}]],0],{i,1,Length[omegarideald]}];
(* the tau=1 is found separately *)
(* l=1 since we are working in units of sigma *)
l=1;
tauonek=Table[Sin[((i-1/2)*delk 1)/((i-1/2)*delk 1)},{i,1,ngrid}];
ometagkdtotal=Sum[(2*(n-i)*omegakd[[i-1]])/norm[[i-1]],{i,2,n-1}]+n
   *1+2(19)*tauonek)/n;
(* save output values for both ideal and nonoverlapping *)
ometagrideald>>ometagrd20merchainfjcno.dat
ometagkdtotal>>ometagkd20merchainfjcnormno.dat

Second is for the cyclic chain

Listing A.2: Non-overlapping Freely Jointed Chain Cyclic

ngrid=2048;
delr=4/100;
delk=\[Pi]/delr/ngrid;
rvalues=Table[(i-1/2)*delr,{i,1,ngrid}];
kvalues=Table[(i-1/2)*delr,{i,1,ngrid}];

wtauid3[r_\[Tau]_] := 1/((2^((\[Tau]+1)\[Pi])((\[Tau]-2)!)) /!
\!*UnderoverscriptBox[\((\((\[Sum]\))\), (s = 0), ((\[Tau] - 2)\)]\)\)
\!*SuperscriptBox[\((((\(-1\)))\), (s\)) \] Binomial[\[Tau] - 2, s]
\!*SuperscriptBox[\((r\), (s - 1)\) \] \)
\!*UnderoverscriptBox[\((\((\[Sum]\))\), (m = 0), (Floor[\((((\[Tau] - 1)\)
/2)\)]\) \)
\!*SuperscriptBox[\((((\(-1\)))\), (m\)) \] Binomial[\[Tau], m] \] UnitStep[\[Tau] - 2 \] m \] \)
\!*SuperscriptBox[\((((\[Tau] - 2) \) \] \) \] \)

n=20;
ometagrideald=Table[Table[N[wtauid3[(i-1/2)*delr,j],64],{i,1,ngrid}],{j,2,n-2}];
ometagrd=Table[ometagrideald[[i]]*ometagrideald[[Length[ometagrideald]-i+1]],{i,1,Length[ometagrideald]}];
ometagrdno=Table[Join[Table[0,{25}],Drop[ometagrd[[i]],25]],{i,1,Length[ometagrd]}];
ometagkd=Table[(2\[Pi])delr Sqrt[2*ngrid]*FourierDST[N[ometagrdno[[i]]*rvalues,64],4])/kvalues,{i,1,Length[ometagrd]}];
norm=Table[Interpolation[Table[{kvalues[[j]],omegakd[[i,j]]},{j,1,ngrid}]],0],{i,1,Length[ometagrd]}];
l=1;
tauonek=Table[Sin[((i-1/2)*delk 1)/((i-1/2)*delk 1)},{i,1,ngrid}];
ometagkdtotal=Sum[(n*omegakd[[i]])/norm[[i]],{i,1,Length[ometagrd]}]+n
   *1+n*tauonek+n*tauonek)/n;

155
SetDirectory ["C:\\Users\\themadchemist\\Documents\\My_Dropbox\\PRISM\\6-13-2011" ];
ogard>>ogard20merringfjcnodata
ogakdtotal>>ogakd20merringfjenormdata

A.2 Wall-PRISM

Here is the code for doing the Wall-PRISM calculations. It has two parts. The first part calculates the bulk blend then the second part uses the results from the first part in the calculation of the surface behavior.

Listing A.3: wall-PRISM code

(* define grid *)
ngrid=2048;
delt=4/100;
delk=\[Pi\]/delt/ngrid;
rvvalues=Table[(i-1/2)delt,{i,1,ngrid}];
kvalues=Table[(i-1/2)delk,{i,1,ngrid}];

(* import results from wall PRISM *)
gzz=<<gzzaafjcf20merringno.dat;
gzzbb=<<gzbbfjcf20merchainno.dat;

(* include density values pa and pb *)
denaa=gzzi pa;
denbb=gzzi pb;

h = 30;

(* reslen determines the resolution and how large the smoothing windows will be *)
reslen=0.25;
truncmax=Floor[h/2/delt];
hwzerrun=gzzi [[Table[i,{i,truncmax,1,-1}]]]-1;
hwzzbtrun=gzzi [[Table[i,{i,truncmax,1,-1}]]]-1;
hwzaatrun=( delr Sqrt[2/\[Pi\]] )/Sqrt[2/ngrid] FourierDCT[hwzerrun,4];
hwzbbtrun=( delr Sqrt[2/\[Pi\]] )/Sqrt[2/ngrid] FourierDCT[hwzzbtrun,4];

kmax=Floor[reslen/delt];
hwzaatrunwin=Join[hwzaatrun[[Range[kmax]]],Table[((1/2)(Cos[\[Pi\](ik-1/2)/

kmax)])+1)},{ik,1,kmax}];

hwzbbtrunwin=Join[hwzbbtrun[[Range[kmax]]],Table[((1/2)(Cos[\[Pi\](ik-1/2)/

kmax)])+1)},{ik,1,kmax}];

hwzaatrunwin=FourierDCT[hwzaatrunwin,4]/(( delr Sqrt[2/\[Pi\]])/Sqrt[2/engan]);
gwzbbtrunwin=FourierDCT[hwkbbtrunwin,4]/((delr Sqrt[2/Pi]) / Sqrt[2/ngrid]) + 1;

A.3 Data smoothing

Code for doing the Hann window smoothing. Based on original smoothing code made by David T. Wu.

Listing A.4: Hann Window Smoothing

(* defining grid*)
ngrid = 2048;
delr = 4/100;
delk = \[Pi] / delr / ngrid;
rvvalues = Table[(i - 1/2) delr, {i, 1, ngrid}];
kvalues = Table[(i - 1/2) delk, {i, 1, ngrid}];

(* importing omega files*)
wa = <<omegakd20merringfjcnormno.dat;
bw = <<omegakd20merchainfjcnormno.dat;
wd = Table[{wa[[i]], 0}, {0, wb[[i]]}, {i, 1, ngrid}];
sigmaAA = 1;
sigmaBB = 1;
sigmaAB = (sigmaAA + sigmaBB) / 2;
sigmaBA = (sigmaAA + sigmaBB) / 2;

(* defining closures PY*)
Clear[hcloseAA]
hcloseAA = {};
i = 1;
While[(i - 1/2)*delr < sigmaAA, hcloseAA = Append[hcloseAA, -1]; i++]
hcloselenAA = Length[hcloseAA];
Clear[hcloseBB]
hcloseBB = {};
i = 1;
While[(i - 1/2)*delr < sigmaBB, hcloseBB = Append[hcloseBB, -1]; i++]
hcloselenBB = Length[hcloseBB];
Clear[hcloseAB]
hcloseAB = {};
i = 1;
While[(i - 1/2)*delr < sigmaAB, hcloseAB = Append[hcloseAB, -1]; i++]
hcloselenAB = Length[hcloseAB];
Clear[hcloseBA]
hcloseBA = {};
i = 1;
While[(i - 1/2)*delr < sigmaBA, hcloseBA = Append[hcloseBA, -1]; i++]
hcloselenBA = Length[hcloseBA];
C(r) closures
Clear [ccloseAA ]
ccloseAA ={};
i=ngrid ;
While[ ( i−1/2 )∗delr >= sigmaAA , ccloseAA =Append[ ccloseAA , 0 ]; i -- ]
closedlenAA =Length [ ccloseAA ];
Clear [ ccloseBB ]
ccloseBB ={};
i=ngrid ;
While[ ( i−1/2 )∗delr >= sigmaBB , ccloseBB =Append[ ccloseBB , 0 ]; i -- ]
closedlenBB =Length [ ccloseBB ];
Clear [ ccloseAB ]
ccloseAB ={};
i=ngrid ;
While[ ( i−1/2 )∗delr >= sigmaAB , ccloseAB =Append[ ccloseAB , 0 ]; i -- ]
closedlenAB =Length [ ccloseAB ];
Clear [ ccloseBA ]
ccloseBA ={};
i=ngrid ;
While[ ( i−1/2 )∗delr >= sigmaBA , ccloseBA =Append[ ccloseBA , 0 ]; i -- ]
closedlenBA =Length [ ccloseBA ];
(*define densities below. be careful to know if it is chain density or monomer site density*)
p = {{ na , pa , 0 }, { 0 , nb , pb }};
id = { { 1 , 0 } , { 0 , 1 } } ;
AFD = 1 ;
AFDList ={};
gammainitial = Table [ { { 0 , 0 } , { 0 , 0 } } , { ngrid } ];
gammafirst = gammainitial ;
step = 1 ;
tol = 10^-7 ;
old = 95/100 ;
new = 1 − old ;
(*start of picard iteration for bulk PRISM*)
Monitor [ While [ AFD > tol ,
extractgAA = Table [ Extract [ gammafirst , { i , 1 , 1 } ] , { i , closedlenAA } ] ;
extractgBB = Table [ Extract [ gammafirst , { i , 2 , 2 } ] , { i , closedlenBB } ] ;
extractgAB = Table [ Extract [ gammafirst , { i , 1 , 2 } ] , { i , closedlenAB } ] ;
extractgBA = Table [ Extract [ gammafirst , { i , 2 , 1 } ] , { i , closedlenBA } ] ;
csolutionAA = ( Join [ closedAA − extractgAA , ccloseAA ] ) rvalues ;
csolutionBB = ( Join [ closedBB − extractgBB , ccloseBB ] ) rvalues ;
csolutionAB = ( Join [ closedAB − extractgAB , ccloseAB ] ) rvalues ;
csolutionBA = ( Join [ closedBA − extractgBA , ccloseBA ] ) rvalues ;
chatAB=2 \[\pi\] delr Sqrt[2*ngrid](FourierDST[SetPrecision[csolutionAB,32],4]/kvalues);
chatBA=2 \[\pi\] delr Sqrt[2*ngrid](FourierDST[SetPrecision[csolutionBA,32],4]/kvalues);
chatmatrix=Table[{{chatAA[[i]],chatAB[[i]]},{chatBA[[i]],chatBB[[i]]}},{i,ngrid}];
gammahat=Table[(wd[[i]].chatmatrix[[i]].Inverse[id−p.wd[[i]].chatmatrix[[i]]].wd[[i]]−chatmatrix[[i]]).kvalues[[i]],{i,ngrid}]./rvalues 2\[\pi] delr Sqrt[2*ngrid]);
gammaAA=FourierDST[SetPrecision[Table[gammahat[[i,1,1]],{i,ngrid}]],32,4]/rvalues 2\[\pi] delr Sqrt[2*ngrid]);
gammaBA=FourierDST[SetPrecision[Table[gammahat[[i,2,1]],{i,ngrid}]],32,4]/rvalues 2\[\pi] delr Sqrt[2*ngrid]);
gammaAB=FourierDST[SetPrecision[Table[gammahat[[i,1,2]],{i,ngrid}]],32,4]/rvalues 2\[\pi] delr Sqrt[2*ngrid]);
gammaBB=FourierDST[SetPrecision[Table[gammahat[[i,2,2]],{i,ngrid}]],32,4]/rvalues 2\[\pi] delr Sqrt[2*ngrid]);
gammasecond=Table[{{gammaAA[[i]],gammaAB[[i]]},{gammaBA[[i]],gammaBB[[i]]}},{i,ngrid}];
AFD=Sqrt[1/ngrid Sum[((gammafirst[[i,1,1]]−gammasecond[[i,1,1]])/(gammafirst[[i,1,1]]+gammasecond[[i,1,1]]))ˆ2+((gammafirst[[i,2,2]]−gammasecond[[i,2,2]])/(gammafirst[[i,2,2]]+gammasecond[[i,2,2]]))ˆ2+((gammafirst[[i,1,2]]−gammasecond[[i,1,2]])/(gammafirst[[i,1,2]]+gammasecond[[i,1,2]]))ˆ2+((gammafirst[[i,2,1]]−gammasecond[[i,2,1]])/(gammafirst[[i,2,1]]+gammasecond[[i,2,1]]))ˆ2,{i,1,ngrid}}];
gammafirst=old gammafirst+new gammasecond;
AFDList=Append[AFDList,AFD];
If[Mod[step,500]==0,
gammafirst>>gammasonlofjchighprecisionTMP.dat;;
AFDList>>AFDListTMP.dat;
step>>stepTMP;
step++;]
{step,AFD]
Print["Number_of_steps",step];
Print["AFD=",AFD];
hsolutionAA=Table[gammafirst[[i,1,1]],{i,ngrid−ccloselenAA+1,ngrid}]+ccloseAA;
hAA=Join[heloseAA,hsolutionAA];
hofrAA=Table[{rvalues[[i]],hAA[[i]]},{i,1,ngrid}];
hsolutionAB=Table[gammafirst[[i,1,2]],{i,ngrid−ccloselenAB+1,ngrid}]+ccloseAB;
hAB=Join[heloseAB,hsolutionAB];
hofrAB=Table[{rvalues[[i]],hAB[[i]]},{i,1,ngrid}];
hsolutionBA=Table[gammafirst[[i,1,2]],{i,ngrid−ccloselenBA+1,ngrid}]+ccloseBA;
hBA=Join[heloseBA,hsolutionBA];
hofrBA=Table[{rvalues[[i]],hBA[[i]]},{i,1,ngrid}];
hsolutionBB = Table[ gammafirst[\[i\],2,2], \{i,ngrid-ccloselenBB+1,ngrid\}] + ccloseBB;

hBB = Join[hcloseBB, hsolutionBB];

hofrBB = Table[rvalues[\[i\]],hBB[\[i\]], \{i,1,ngrid\}];

(* save output files *)
gammafirst >> gammasolnnofjchighprecisionTMP.dat;

step >> stepTMP;

AFDList >> AFDListTMP.dat;

chatmatrix >> chatmatrixfjc20merno.dat;

(* define width of polymer melt *)
h = 30;

omegaazero = Interpolation[Table[kvalues[\[i\]],wd[\[i\],1,1]], \{i,1,ngrid\},0];

omegabzero = Interpolation[Table[kvalues[\[i\]],wd[\[i\],2,2]], \{i,1,ngrid\},0];

chataazero = Interpolation[Table[kvalues[\[i\]],chatmatrix[\[i\],1,1]], \{i,1,ngrid\},0];

chatabzero = Interpolation[Table[kvalues[\[i\]],chatmatrix[\[i\],1,2]], \{i,1,ngrid\},0];

chatbzero = Interpolation[Table[kvalues[\[i\]],chatmatrix[\[i\],2,1]], \{i,1,ngrid\},0];

chatbbzero = Interpolation[Table[kvalues[\[i\]],chatmatrix[\[i\],2,2]], \{i,1,ngrid\},0];

lambdazero = 1 - pa na omegaazero chataazero - pb nb omegabzero chatbzero + pa na pb nb omegazero omegabzero (chataazero chatbbzero - chatabzero^2);
shatabzero = (pa na omegazero(1 - pb nb omegabzero chatbbzero))/lambdazero;
shatabzero = (pa na pb nb omegazero omegabzero chatbzero)/lambdazero;
shatabzero = (pb nb omegabzero(1 - na pa omegazero chatazero))/lambdazero;

a1AA = -(pa na shatabzero - pb nb shatabzero)/(shatabzero^2 - shataazero shatbbzero);

a1BB = -(pb nb shatabzero - pa na shatabzero)/(shatabzero^2 - shataazero shatbbzero);

(* defining wall PRISM closures *)

cpaaclosure = \{\};
i = 1;
While[(i - 1/2)*delr < h/2, cpaaclosure = Append[cpaaclosure, a1AA]; i++]
cpaacloselen = Length[cpaaclosure];

cpbbclosure = \{\};
i = 1;
While[(i - 1/2)*delr < h/2, cpbbclosure = Append[cpbbclosure, a1BB]; i++]

160
cpbbcloselen = Length[cpbbclosure];
(* defining parameters for wall-PRISM picard iteration *)
gammazinitial = Table[{{0}, {0}, {ngrid}}];
gammafirst = gammazinitial;
step2 = 1;
AFD2 = 1;
old2 = 95/100;
new2 = 1 - old2;

\[ \text{tol} = 10^{-7}; \]

nmatrix = Table[
   {na, 0},
   {0, nb},
   {ngrid},
];
structure = Table[Inverse[id.p.wd[[i]].chatmatrix[[i]]].wd[[i]].p, {i, 1, ngrid}];

AFD2List = {};
(* begin wall PRISM picard iteration *)
Monitor[While[AFD2 > tol, extractgammawaa = Table[gammafirst[[i, 1, 1]], {i, cpaacloselen + 1, ngrid}]; extractgammawbb = Table[gammafirst[[i, 2, 1]], {i, cpbbcloselen + 1, ngrid}]; cpaa = Join[cpaaaclosure, -(extractgammawaa)]; cpbb = Join[cpbbclosure, -(extractgammawbb)]; cpaaahat = (delr Sqrt[2/(Pi)] / Sqrt[2/ngrid]) FourierDCT[SetPrecision[cpaa, iterPrecision], 4];
   cpbbhats = (delr Sqrt[2/(Pi)] / Sqrt[2/ngrid]) FourierDCT[SetPrecision[cpbb, iterPrecision], 4];

   cphat = Table[
      {cpaahat[[i]], cpbbhats[[i]]},
      {i, 1, ngrid},
    ];

   gammak = Table[
      Inverse[p].structure[[i]].cphat[[i]] - cphat[[i]],
      {i, 1, ngrid},
    ];

   gammak = Table[
      FourierDCT[SetPrecision[Table[gammak[[i, 1, 1]], {i, 1, ngrid}], iterPrecision], 4],
      {i, 1, ngrid},
      ];

   gammab = FourierDCT[SetPrecision[Table[gammak[[i, 2, 1]], {i, 1, ngrid}], iterPrecision], 4];

   gammasecond = Table[
      {gammab[[i]]},
      {i, 1, ngrid},
    ];

   AFD2 = Sqrt[1/ngrid] Sum[(gammafirst[[i, 1, 1]] - gammasecond[[i, 1, 1]])^2 + ((gammafirst[[i, 2, 1]] - gammasecond[[i, 2, 1]])^2, {i, 1, ngrid};

   gammasecond = gammasecond + new2 * gammasecond;

   gammafirst = old2 * gammafirst + new2 * gammasecond;

   AFD2List = Append[AFD2List, AFD2];
   If[Mod[step2, 1000] == 0, gammafirst >> gammawall.dat; AFD2List >> AFD2List.dat; step2 >> step2.dat; step2 +++, {step2, AFD2, getPlothr, getPlotgammadiff}]

(* save output results *)
gammafirst >> gammawall.dat;
AFD2List >> AFD2List.dat;

(* save output results *)
step2>>step2.dat;

gwkaa=Table[[\!
\!\text{gammak}[[i,1,1]],\{i,1,ngrid\}\!\!]+cpaahat;
gwkbb=Table[[\!
\!\text{gammak}[[i,2,1]],\{i,1,ngrid\}\!\!]+cpbbhat;
gwzaa=FourierDCT[gwkaa,4]/((\text{delr} \text{Sqrt}[2/\text{Pi}])/\text{Sqrt}[2/ngrid]);
gwzbb=FourierDCT[gwkbb,4]/((\text{delr} \text{Sqrt}[2/\text{Pi}])/\text{Sqrt}[2/ngrid]);
denaa=gwzaa pa;
denbb=gwzbb pb;
(* output final g(r) data from wall prism*)
gwzaa>>gwzaafjc20merringno.dat;
gwzbb>>gwzbbfjc20merchainno.dat;
APPENDIX B - PARALLEL DIRECT SAMPLING

For Two-Molecule RISM, 99% of the time is spent in the simulation. Of the simulation time, generation of the configurations of small molecules is a very small portion. Most of the time is instead spent sampling the two molecules. With most modern computers having multiple cores I can take advantage by splitting the sampling time among the cores. To do this, I first generate my population of small molecules. Then a copy of the configurations are made for each thread. Testing has shown that the number of threads should be equal to the number of physical cores. (Typically 4 cores gives maximum performance.) The total number of MC steps is divided by the number of cores and given a specific thread ID (TID). Each thread will then run normal direct sampling runs just a few number of them in total. The results from these samplings are stored in a large array. This array is however many threads there are times the number of grid points. Each thread will have an offset included that is equal to $TID \times ngrid$. This ensures that there are no race conditions of having two threads try to write to the same portion of the array. Care should be taken to have each thread have its own random number seed that is independent of the other threads to ensure good statistics. Once all the threads are completed, the large array is then consolidated down into a single array of appropriate length. See Figure B.1 below.

![Diagram of Parallel Direct Sampling Scheme](image)

Figure B.1: Diagram of Parallel Direct Sampling Scheme
APPENDIX C - TWO-CHAIN CODE

The code for all these systems run was written in two parts. The simulation portion of the code was written in C++. The RISM/PRISM solving features and application of the “tweaking” vector was all written in Mathematica. Mathlink and subsequently Wolfram Symbolic Transfer Protocol (WSTP) were used. These were all compiled as per the instructions for making linked executables from the Mathematica documentation. Here I will give example code for the athermal trimer.

C.1 Direct sampling code

Below is the code for the athermal trimer as well as the needed code to compile it to work with WSTP. This also includes the code, though commented out, to work with water. This should be usable as an example to alter the code to work for the specific system.

Listing C.5: C++ code for HS Trimer Direct Sampling

```cpp
#include "wstp.h"
#include <iostream>
#include <cmath>
#include <thread>
#include <random>
#include <cstdlib>
#include <ctime>
#include <cstring>
using namespace std;

// thread safe random number generators

inline float rng(unsigned int* seedp)
{
    return (float)rand_r(seedp) / (float)RAND_MAX;
}

inline int rand_IntRange(int a, int b, unsigned int* seedp)
{
```

164
return rand_r(seedp) % (b - a + 1) + a;
}
// gives a random vector of length r
void randomonsphere(float c[3], float r, unsigned int* seedp)
{
    const float pi = 3.14159265f;
    float theta = rng(seedp) * 2.0f * pi;
    float u = 2.0f * rng(seedp) - 1.0f;
    float t = r * sqrt(1.0f - u*u);
    c[0] = cos(theta) * t;
    c[1] = sin(theta) * t;
    c[2] = r*u;
}
// matrix multiplication function
void matvectmultiply(float mat1[3][3], float vect[3], float result[3])
{
    for (int q = 0; q < 3; q++)
    {
        result[q] = 0;
    }
    for (int x = 0; x < 3; x++)
    {
        for (int c = 0; c < 3; c++)
        {
            result[x] += mat1[x][c] * vect[c];
        }
    }
}
// moves molecule to origin
void zerodisplace(float chain[][3], int site)
{
    float zero[3];
    for (int i = 0; i < 3; i++)
    {
        zero[i] = -(chain[site][i]);
    }
    for (int i = 0; i < 3; i++)
    {
        for (int j = 0; j < 3; j++)
        {
            chain[i][j] += zero[j];
        }
    }
}
void compute_energy(int start, int stop, int mcmx, unsigned int seed, float* ret, int threadId, int vectlen, int poplen, float* wr0, float* wr1, float* wr2, float sigmaa, float sigmab, float bl, float bondangle, float tp, int numcols, float delr, int ngrid, float solFactor) // if someone need it
{
    // std::cerr << "Thread: " << threadId << "\n";
    // float sited[3];

    float Pi = 3.14159265f;
    float kb = 0.0019833794749f;

    // 1/(4*Pi*epsilon_o) for coulombic term
    float ke = 332.06377868f;

    int rmin = start;
    int rmax = stop;
    // float ljshift = 315426.;
    // float ljcutoff = 1.1224620483f;

    float lj_entemp;
    float coul_entemp;

    float wr[2048][3];

    for(int i = 0; i < 2048; i++){
        wr[i][0] = wr0[i];
        wr[i][1] = wr1[i];
        wr[i][2] = wr2[i];
    }

    // float wr0[ngrid] = {0.0f};
    // float wr1[ngrid] = {0.0f};
    // float wr2[ngrid] = {0.0f};

    float gr[2048][3] = {0.0f, 0.0f, 0.0f};

    float sigma[3] = { 0.5f*(sigmaa + sigmaa), 0.5f*(sigmaa + sigmab) ,
                      0.5f*(sigmab + sigmab) };
    // float epsilon[3] = {sqrt(epsilona*epsilona), sqrt(epsilona*epilsonb ),
                          sqrt(epsilonb*epsilonb) };

    // std::cerr << "Thread: " << threadId << "\n";
}

166
float qlist[3] = {qoxygen*qoxygen, qhydrogen*qoxygen, qhydrogen*qhydrogen};

unsigned int* seedp = &seed;

// Below generates the Dimer configuration population and distance vector orientations

// generate new array for vectors
float vectorlist[veclen][numcols];
float chainpop[poplen][numcols];
// generate new array for vectors
float **vectorlist = new float*[veclen];
for (int i = 0; i < veclen; ++i)
{
    vectorlist[i] = new float[3];
}
// generate new array orientations of dimers
float **chainpop = new float*[3*poplen];
for (int i = 0; i < 3*poplen; ++i)
{
    chainpop[i] = new float[3];
}

// populate orientation and vector arrays
float bondangle_r = bondangle*Pi/180;
float ycoord = sin(Pi-Pi/2 - bondangle_r/2)*bl;
float xcoord = cos(Pi-Pi/2 - bondangle_r/2)*bl;

float trimer_ini[3][3] = {{0.0f, 0.0f, 0.0f}, {xcoord, ycoord, 0.0f}, {2*xcoord, 0.0f, 0.0f}};
float phi;
float theta;
float gamma;
float rx[3][3];
float ry[3][3];
float rz[3][3];
float trimer_temp[3][3];
float siterotationx[3];
float siterotationy[3];
float siterotationz[3];

// define sites
// 0 = Hydrogen
// 1 = Oxygen
int sitetype[3] = {1,0,1};
```c
int pair_index[3][3];
for(int i = 0; i < 3; i++){
    for(int j = 0; j < 3; j++){
        if(sitetype[i] == 0 && sitetype[j] == 0){
            pair_index[i][j] = 0;
        } else if(sitetype[i] == 0 && sitetype[j] == 1){
            pair_index[i][j] = 1;
        } else if(sitetype[i] == 1 && sitetype[j] == 0){
            pair_index[i][j] = 1;
        } else if(sitetype[i] == 1 && sitetype[j] == 1){
            pair_index[i][j] = 2;
        }
    }
}

// Generating the randomly oriented trimers
for(int i = 0; i <(poplen); i++){
    phi = rng(seedp) * 2 * Pi;
    theta = rng(seedp) * 2 * Pi;
    gamma = rng(seedp) * 2 * Pi;
    float rx[3][3] = {{1.0f, 0.0f, 0.0f}, {0.0f, cos(phi), -sin(phi)},
                      {0.0f, sin(phi), cos(phi)}};
    float ry[3][3] = {{cos(theta), 0.0f, sin(theta)}, {0.0f, 1.0f, 0.0f},
                      {-sin(theta), 0.0f, cos(theta)}};
    float rz[3][3] = {{cos(gamma), -sin(gamma), 0.0f}, {sin(gamma),
                      cos(gamma), 0.0f}, {0.0f, 0.0f, 1.0f}};

    for(int j = 0; j < 3; j++){
        matvectmultiply(rx, trimers_i[j], siternotationx);
        matvectmultiply(ry, siternotationx,
                        siternotationy);
        matvectmultiply(rz, siternotationy,
                        siternotationz);
        for(int k = 0; k < 3; k++){
            trimer_temp[j][k] = siternotationz[k];
        }
    }
}
for(int j = 0; j < 3; j++){
    for(int k = 0; k < 3; k++){
        chainpop[3*i + j][k] = trimer_temp[j][k];
    }
}
```

168
for (int p = 0; p < vectlen; p++)
{
    float sited[3];
    randomonsphere(sited, 1, seedp);
    for (int c = 0; c < numcols; c++)
    {
        vectorlist[p][c] = sited[c];
    }
}

// Actual sampling. The outer loop does the total number of MC steps.
The inner loop for n is what controls the two molecules over a
specified distance (rmin and rmax)
for (int iii = 0; iii < mcmax; ++iii)
{
    for (int n = rmin; n <= rmax; n++)
    {
        int choice1 = rand_IntRange(0, poplen - 1, seedp);
        int choice2 = rand_IntRange(0, poplen - 1, seedp);
        int choice3 = rand_IntRange(0, vectlen - 1, seedp);

        float chain1[3][3];
        float chain2[3][3];
        float dis[3];
        for (int q = 0; q < 3; q++)
        {
            for (int qq = 0; qq < 3; qq++)
            {
                chain1[q][qq] = chainpop[choice1 * 3 + q][qq];
                chain2[q][qq] = chainpop[choice2 * 3 + q][qq];
            }
            dis[q] = vectorlist[choice3][q] * delr * n;
        }
        float solE = 0.0f;

        // zero random site
        int choice_site1 = rand_IntRange(0, 3 - 1, seedp);
        int choice_site2 = rand_IntRange(0, 3 - 1, seedp);
        zero_displace(chain1, choice_site1);
        zero_displace(chain2, choice_site2);

        int temp[3][3] = {{ 0, 0, 0 }, { 0, 0, 0 }, { 0, 0, 0 }};
        float energy = 0.0f;
        float entemp = 0.0f;
for (int j = 0; j < 3; j++)
{
    for (int k = 0; k < 3; k++)
    {
        float sr = 0.0f;
        for (int s = 0; s < 3; s++)
        {
            float vr = chain1[k][s] - chain2[j][s] + dis[s];
            sr += vr * vr;
        }
        sr = sqrt(sr);
        int disindex = round(sr/delr);
        if (disindex > 0){
            temp[j][k] = disindex;
        } else{
            temp[j][k] = 0;
        }
        int rIndex = ceil(sr/delr);
        float rFactor = (rIndex - (sr/delr));
        if (rIndex == 0)
        {
            solE = solE + solFactor*wr[0][pair_index[j][k]];
        }
        else if (rIndex < ngrid)
        {
            solE = solE + solFactor*(wr[rIndex - 1][
                                        pair_index[j][k]] * (rFactor) + (1 - rFactor) *wr[rIndex][pair_index[j][k]]);
        }
    // For Water only LJ is calculated on the oxygen, which is the site in index 1;
        //
        //
        // pair_index[j][k] * sigma[pair_index[j][k]] / (sr*sr);
        // r2inv = r2inv * r2inv;
        //
        // (pair_index[j][k]) * (r6inv*(r6inv - 1.0));
        //
        //
        // The hard sphere is enforced here rather crudely by just making the potential extremly sharp and high.
        if (sr >= sigma[pair_index[j][k]])
        {
            float r2inv = sigma[

            pair_index[j][k]] / (sr*sr);
            float r6inv = r2inv * r2inv;
            float lj_entemp = 4 * epsilon
            lj_entemp = lj_entemp;
            energy = energy +
        }
entemp = 0.0;
}

else{
    entemp = 1000000000.0f;
}
// coul_entemp = scale*ke*qlist[pair_index[j][k]]/sr
    energy = energy + entemp;
}

for (int j = 0; j < 3; j++)
{
    for (int k = 0; k < 3; k++)
    {
        gr[temp[j][k]][pair_index[j][k]] += exp(-(energy + solE) / (kb*tp)) * n*n;
    }
}

// pack and marshall data
int i = 0;
for (int k = 0; k != ngrid; ++k)
{
    for (int q = 0; q != 3; ++q)
    {
        ret[i] = gr[k][q];
        ++i;
    }
}

// free dynamic memory
for (int i = 0; i < vectlen; i++)
{
    delete[] vectorlist[i];
}
}
for (int i = 0; i < 3*poplen; i++)
{
    delete[] chainpop[i];
}
delete[] vectorlist;
delete[] chainpop;
void wstpthreadhstrimer(float *wr0, int wr0len, float *wr1, int wr1len, 
float *wr2, int wr2len, float *plist, int plisrlen, float solFactor 
, int poplen, int vectlen, int mcmmax, int start, int stop)
{
    float sigmaa = plist[0];
    float sigmab = plist[1];
    float bl = plist[2];
    float tp = plist[3];
    float bondangle = plist[4];
    // float epsilona = plist[4];
    // float epsilonb = plist[5];
    // float q_oxygen = plist[6];
    // float q_hydrogen = plist[7];
    // float scale = plist[8];
    float delr = 0.1f;
    int numcols = 3;
    int ngrid = 2048;

    int dim[2];
    dim[0] = ngrid;
    dim[1] = 3;

    constexpr int nt = 4; // four threads

    float* r[nt];
    std::thread t[nt];
    unsigned int seed[nt];
    int preseed;
    preseed = time(NULL);
    // fork multiple threads
    for(int s = 0; s != nt; ++s)
    {
        seed[s] = preseed * (s + 1) + preseed*s; // different for each thread
        r[s] = new float[ngrid*3]; // different for each thread!
        t[s] = std::thread(compute_energy, start, stop, mcmmax/nt, 
            seed[s], r[s], s, vectlen, poplen, wr0, wr1, wr2, sigmaa, 
            sigmab, bl, bondangle, tp, numcols, delr, ngrid, solFactor);
    }
    // (int start, int stop, int mcmmax, unsigned int seed, float* ret, 
    // int threadId, int vectlen, int poplen, float wr0[2048], float wr1 
    // [2048], float wr2[2048], float sigmaa, float sigmab, float bl, float 
    // tp, int numcols, float delr, int ngrid, float solFactor);
    for(int s = 0; s != nt; ++s) // wait till all threads are done
        t[s].join();

    // final result storage

172
float gr[2048][3] = {0.0f, 0.0f, 0.0f};

// unpack marshalled data and merge
for(int s = 0; s != nt; ++s)
{
    float* rr = r[s];
    int i = 0;
    for(int k = 0; k != ngrid; ++k)
    {
        for(int q = 0; q != 3; ++q)
        {
            gr[k][q] += rr[i];
            ++i;
        }
    }
}

WSPutReal32Array( stdlink, (float*)gr, (int*)dim, (const char**)0, 2);
return ;

// this is needed to work with WSTP
#if WINDOWS_WSTP
#endif

int WinMain( HINSTANCE hinstCurrent, HINSTANCE hinstPrevious, LPSTR lpszCmdLine, int nCmdShow)
{
    char buff[512];
    char FAR * buff_start = buff;
    char FAR * argv[32];
    char FAR * FAR * argv_end = argv + 32;

    hinstPrevious = hinstPrevious; /* suppress warning */

    if( !WSInitializeIcon( hinstCurrent, nCmdShow)) return 1;
    WSScanString( argv, &argv_end, &lpszCmdLine, &buff_start);
    return WSMain( (int)(argv_end - argv), argv);
}

#else

int main(int argc, char* argv[])
{

Here is the file needed to make sure the Mathematica functions are linked to the correct libraries. These two files are compiled separately then linked.

Listing C.6: C++ code using WSTP libraries

```c
/*
 * This file automatically produced by wsprep from:
 * WSTP_Thread_Water.tm
 * mprep Revision 18 Copyright (c) Wolfram Research, Inc. 1990-2013
 */

#define MPREP_REVISION 18

#include "wstp.h"

int WSAabort = 0;
int WSDone = 0;
long WSSpecialCharacter = '\0';

WSLINK stdlink = 0;
WSEnvironment stdenv = 0;
#if WSINTERFACE >= 3
WSYieldFunctionObject stdyielder = (WSYieldFunctionObject)0;
WSMessageHandlerObject stdhandler = (WSMessageHandlerObject)0;
#else
WSYieldFunctionObject stdyielder = 0;
WSMessageHandlerObject stdhandler = 0;
#endif /* WSINTERFACE >= 3 */

/*******************************************************************************/

void wstpthreadhstrimer P(( float * _tp1, int _tp11, float * _tp2, int _tp21, float * _tp3, int _tp31, float * _tp4, int _tp41, float * _tp5, int _tp51, int _tp6, int _tp7, int _tp8, int _tp9, int _tp10));

#if WSPROTOTYPES
static int _tr0( WSLINK mlp)
#else
static int _tr0(mlp) WSLINK mlp;
```
#endif
{
    int    res = 0;
    float * _tp1;
    int   _tp1l;
    float * _tp2;
    int   _tp2l;
    float * _tp3;
    int   _tp3l;
    float * _tp4;
    int   _tp4l;
    float _tp5;
    int   _tp6;
    int   _tp7;
    int   _tp8;
    int   _tp9;
    int   _tp10;
    if ( ! WSGetReal32List ( mlp , &_tp1 , &_tp1l) ) goto L0;
    if ( ! WSGetReal32List ( mlp , &_tp2 , &_tp2l) ) goto L1;
    if ( ! WSGetReal32List ( mlp , &_tp3 , &_tp3l) ) goto L2;
    if ( ! WSGetReal32List ( mlp , &_tp4 , &_tp4l) ) goto L3;
    if ( ! WSGetReal32 ( mlp , &_tp5) ) goto L4;
    if ( ! WSGetInteger( mlp , &_tp6) ) goto L5;
    if ( ! WSGetInteger( mlp , &_tp7) ) goto L6;
    if ( ! WSGetInteger( mlp , &_tp8) ) goto L7;
    if ( ! WSGetInteger( mlp , &_tp9) ) goto L8;
    if ( ! WSGetInteger( mlp , &_tp10) ) goto L9;
    if ( ! WSNNewPacket(mlp) ) goto L10;

    wstpthreadstrimer(_tp1 , _tp1l , _tp2 , _tp2l , _tp3 , _tp3l , _tp4 , _tp4l , _tp5 , _tp6 , _tp7 , _tp8 , _tp9 , _tp10);
    res = 1;
L10: L9: L8: L7: L6: L5: L4:    WSRReleaseReal32List(mlp , _tp4 , _tp14);
L3:    WSRReleaseReal32List(mlp , _tp3 , _tp13);
L2:    WSRReleaseReal32List(mlp , _tp2 , _tp12);
L1:    WSRReleaseReal32List(mlp , _tp1 , _tp11);
L0:    return res;
} /* _tr0 */

static struct func {
    int    f_nargs;
    int    manual;
    int  (*f_func)P((WSLINK));
    const char  *f_name;
};
```c
#define CARDOF_EVALSTRS 0

static int _definepattern P(( WSLINK, char*, char*, int));
int _WSDoCallPacket P(( WSLINK, struct func[], int));

#if WSPROTOTYPES
int WSIInstall( WSLINK mlp)
#else
int WSIInstall(mlp) WSLINK mlp;
#endif
{
  int _res;
  _res = WSConnect(mlp);
  if (_res) _res = _definepattern(mlp, (char *))
 wstpthreadhstrimer[a1List, a2List, a3List, p1List, bReal, cInteger, dInteger, eInteger, fInteger, gInteger], (char *)"{a1, a2, a3, p1, b, c, d, e, f, g}"
  ));
  if (_res) _res = WSPutSymbol( mlp, "End")
  if (_res) _res = WSFlush( mlp);
  return _res;
} /* WSIInstall */

#if WSPROTOTYPES
int WSDoCallPacket( WSLINK mlp)
#else
int WSDoCallPacket( mlp) WSLINK mlp;
#endif
{
  return _WSDoCallPacket( mlp, _tramps, 1);
} /* WSDoCallPacket */

/**************************** begin trailer
 ****************************/

#ifndef EVALSTRS_AS_BYTESTRINGS
#define EVALSTRS_AS_BYTESTRINGS 1
#endif
```
#if CARDOF_EVALSTRS
#if WSPROTOTYPES
static int _doevalstr( WSLINK mlp, int n)
#else
static int _doevalstr( mlp, n)
    WSLINK mlp; int n;
#endif
{
    long bytesleft, charsleft, bytesnow;
#if ! EVALSTRS_AS_BYTESTRINGS
    long charsnow;
#endif
    char **s, **p;
    char *t;

    s = (char **)evalstrs;
    while( n-- > 0){
        if( *s == 0) break;
        while( *s++ != 0){}
    }
    if( *s == 0) return 0;
    bytesleft = 0;
    charsleft = 0;
    p = s;
    while( *p){
        t = *p; while( *t ) ++t;
        bytesnow = t - *p;
        bytesleft += bytesnow;
        charsleft += bytesnow;
    }
    if( !EVALSTRS_AS_BYTESTRINGS
        t = *p;
        charsleft -= WSCharacterOffset( &t, t + bytesnow, bytesnow);
        /* assert( t == *p + bytesnow); */
    }
    ++p;
}

WSPutNext( mlp, WSTKSTR);
#if EVALSTRS_AS_BYTESTRINGS
p = s;
while( *p){
    t = *p; while( *t ) ++t;
    bytesnow = t - *p;
    bytesleft -= bytesnow;
}
#endif
WSPut8BitCharacters( mlp, bytesleft, (unsigned char*)p, bytesnow);
++p;
}
#else
WSPut7BitCount( mlp, charsleft, bytesleft);
p = s;
while(*p){
    t = *p; while(*t) ++t;
    bytesnow = t - *p;
    bytesleft -= bytesnow;
    t = *p;
    charsnow = bytesnow - WSCharacterOffset( &t, t + bytesnow, bytesnow);
    /* assert( t == *p + bytesnow); */
    charsleft -= charsnow;
    WSPut7BitCharacters( mlp, charsleft, *p, bytesnow, charsnow);
    ++p;
}
#endif
return WSError( mlp) == WSEOK;
#endif /* CARDOF_EVALSTRS */

#if WSPROTOTYPES
static int _definepattern( WSLINK mlp, char *patt, char *args, int func_n)
#else
static int _definepattern( mlp, patt, args, func_n)
#endif
{
    WSPutFunction( mlp, ”DefineExternal”, (long)3);
    WSPutString( mlp, patt);
    WSPutString( mlp, args);
    WSPutInteger( mlp, func_n);
    return !WSError(mlp);
} /* _definepattern */

#if WSPROTOTYPES
int _WSDoCallPacket( WSLINK mlp, struct func functable [], int nfuncs)
#else
int _WSDoCallPacket( mlp, functable, nfuncs)
    WSLINK mlp;
    struct func functable[];
    int nfuncs;
#endif
{
#if WSINTERFACE >= 4
    int len;
#else
    long len;
#endif
    int n, res = 0;
    struct func* funcp;
    if( ! WSGetInteger( mlp, &n) || n < 0 || n >= nfuncs) goto L0 ;
    funcp = &functable[n];
    if( funcp->f_nargs >= 0
#if WSINTERFACE >= 4
    && ( ! WSTestHead(mlp, "List", &len)
#else
    && ( ! WSCheckFunction(mlp, "List", &len)
#endif
    || ( !funcp->manual && ( len != funcp->f_nargs))
    || ( funcp->manual && ( len < funcp->f_nargs))
    )
    ) goto L0;

    stdlink = mlp;
    res = (*funcp->f_func)( mlp);
L0: if( res == 0)
    res = WSClearError( mlp) && WSPutSymbol( mlp, "$Failed"
    );
    return res && WSEndPacket( mlp) && WSNewPacket( mlp);} /* _WSDoCallPacket */

#if WSPROTOTYPES
wsapi_packet WSAnswer( WSLINK mlp)
#else
wsapi_packet WSAnswer( mlp)
    WSLINK mlp;
#endif
{
    wsapi_packet pkt = 0;
# if WSINTERFACE >= 4
  int waitResult;

  while ( ! WSDone && ! WSError(mlp) 
      && (waitResult = WSWaitForLinkActivity(mlp), waitResult) 
      && waitResult == WSWAITSUCCESS && (pkt = WSNextPacket(mlp) 
                                        , pkt) && 
      pkt == CALLPKT)
  {
    WSAabort = 0;
    if (! WSDoCallPacket(mlp))
      pkt = 0;
  }

# else
  while ( ! WSDone && ! WSError(mlp) && (pkt = WSNextPacket(mlp) , 
                                        pkt) && pkt == CALLPKT){
    WSAabort = 0;
    if ( ! WSDoCallPacket(mlp)) pkt = 0;
  }
# endif

  WSAabort = 0;
  return pkt;
} /* WSAnswer */

/*
Module [    { me = "$ParentLink"},
            $ParentLink = contents of RESUMEPKT;
Message[ MessageName["notfe"], me];
               me]
*/

#else

static int refuse_to_be_a_frontend( WSLINK mlp)
#endif

{  
  int pkt;

  WSPutFunction( mlp, "EvaluatePacket", 1);
  WSPutFunction( mlp, "Module", 2);
  WSPutFunction( mlp, "List", 1);
  WSPutFunction( mlp, "Set", 2);

  #endif
```c
WSPutSymbol( mlp, "me" );
WSPutSymbol( mlp, "$ParentLink" );
WSPutFunction( mlp, "CompoundExpression", 3 );
WSPutFunction( mlp, "Set", 2 );
WSPutSymbol( mlp, "$ParentLink" );
WSTransferExpression( mlp, mlp );
WSPutFunction( mlp, "Message", 2 );
WSPutFunction( mlp, "MessageName", 2 );
WSPutSymbol( mlp, "$ParentLink" );
WSPutString( mlp, "notfe" );
WSPutSymbol( mlp, "me" );
WSPutSymbol( mlp, "me" );
WSEndPacket( mlp );

while( (pkt = WSNextPacket( mlp), pkt) && pkt != SUSPENDPKT)
    WSNNewPacket( mlp );
WSNewPacket( mlp );
return WSError( mlp ) == WSEOK;
}

#if WSPROTOTYPES
#if WSINTERFACE >= 3
int WSEvaluate( WSLINK mlp, char *s)
#else
int WSEvaluate( WSLINK mlp, charp_ct s)
#endif /* WSINTERFACE >= 3 */
#else
int WSEvaluate( mlp, s)
    WSLINK mlp;
#if WSINTERFACE >= 3
    char *s;
#else
    charp_ct s;
#endif /* WSINTERFACE >= 3 */
#endif
{
    if( WSAbort) return 0;
    return WSPutFunction( mlp, "EvaluatePacket", 1L)
        && WSPutFunction( mlp, "ToExpression", 1L)
        && WSPutString( mlp, s)
        && WSEndPacket( mlp );
} /* WSEvaluate */

#if WSPROTOTYPES
#if WSINTERFACE >= 3

181
```
int WSEvaluateString( WSLINK mlp, char *s)
#else
int WSEvaluateString( WSLINK mlp, char_p ct s)
#endif /* WSINTERFACE >= 3 */
#else
int WSEvaluateString( mlp, s)
WLINK mlp;
#if WSINTERFACE >= 3
    char *s;
#else
    char_p ct s;
#endif /* WSINTERFACE >= 3 */
#endif
{
    int pkt;
    if( WSAbort) return 0;
    if( WSEvaluate( mlp, s)){
        while( (pkt = WSAnswer( mlp), pkt) && pkt != RETURNPKT)
            WSNewPacket( mlp);
    }
    return WSError( mlp) == WSEOK;
} /* WSEvaluateString */

#if WSINTERFACE >= 3
#if WSPROTOTYPES
void WSDefaultHandler( WSLINK mlp, int message, int n)
#else
void WSDefaultHandler( mlp, message, n)
    WSLINK mlp;
    int message, n;
#endif
#else
#if WSPROTOTYPES
void WSDefaultHandler( WSLINK mlp, unsigned long message, unsigned long n)
#else
void WSDefaultHandler( mlp, message, n)
    WSLINK mlp;
    unsigned long message, n;
#endif
#endif /* WSINTERFACE >= 3 */
{
    switch (message){
        case WSTerminateMessage:
            WSDone = 1;
        }
case WSIInterruptMessage:
case WSAbortMessage:
    WSAbort = 1;
default:
    return;
}

#if WSPROTOYPES
#if WSINTERFACE >= 3
static int _WSMain( char **argv, char **argv_end, char *commandline)
#else
static int _WSMain( charpp_ct argv, charpp_ct argv_end, charp_ct
    commandline)
#endif /* WSINTERFACE >= 3 */
#else
static int _WSMain( argv, argv_end, commandline)
#if WSINTERFACE >= 3
    char **argv, argv_end;
    char *commandline;
#else
    char *argv, argv_end;
    char *commandline;
#endif /* WSINTERFACE >= 3 */
#endif
{
    WSLINK mlp;
#if WSINTERFACE >= 3
    int err;
#else
    long err;
#endif /* WSINTERFACE >= 3 */
#if WSINTERFACE >= 4
    if( !stdenv)
        stdenv = WSIinitialize( (WSEnvironmentParameter)0);
#else
    if( !stdenv)
        stdenv = WSIinitialize( (WSParametersPointer)0);
#endif
    if( stdenv == (WSEnvironment)0) goto R0;
#if WSINTERFACE >= 3
    if( !stdhandler)
        stdhandler = (WSMessageHandlerObject)WSDefaultHandler;
#else
if( !stdhandler )
    stdhandler = WSCreateMessageHandler( stdenv ,
                                          WSDefaultHandler , 0 );
#endif /* WSINTERFACE >= 3 */

mlp = commandline
    ? WSOOpenString( stdenv , commandline , &err )
#if WSINTERFACE >= 3
    : WSOOpenArgcArgv( stdenv , (int ) (argv_end - argv) , argv ,
                        &err );
#else
    : WSOOpenArgv( stdenv , argv , argv_end , &err );
#endif
if( mlp == (WSLINK)0 ){
    WSAAlert( stdenv , WSErrorString( stdenv , err ) );
    goto R1;
}
if( stdyielder ) WSSetYieldFunction( mlp , stdyielder );
if( stdhandler ) WSSetMessageHandler( mlp , stdhandler );

if( WSInstall( mlp ) )
    while( WSAnswer( mlp ) == RESUMEPKT ){
        if( !refuse_to_be_a_frontend( mlp ) ) break;
    }
WSClose( mlp );
R1:  WSDeinitialize( stdenv );
    stdenv = (WSEnvironment)0;
R0:  return !WSDone;
} /* WMain */

#if WSPROTOTYPES
#if WSINTERFACE >= 3
int WMainString( char *commandline )
#else
int WMainString( charp_ct commandline )
#endif /* WSINTERFACE >= 3 */
#else
#if WSINTERFACE >= 3
int WMainString( commandline ) char *commandline ;
#else
int WMainString( commandline ) charp_ct commandline ;
#endif /* WSINTERFACE >= 3 */
#endif
{    return _WSMain( (charp_ct)0, (charp_ct)0, commandline);}

#if WSPROTOTYPES
int WSMMainArgv( char** argv, char** argv_end) /* note not FAR pointers */
#else
int WSMMainArgv( argv, argv_end) char **argv, **argv_end;
#endif
{
    static char FAR * far_argv[128];
    int count = 0;

    while(argv < argv_end)
        far_argv[count++] = *argv++;

    return _WSMain( far_argv, far_argv + count, (charp_ct)0);
}

#if WSPROTOTYPES
#if WSINTERFACE >= 3
int WSMMain( int argc, char **argv)
#else
int WSMMain( int argc, charpp_ct argv)
#endif /* WSINTERFACE >= 3 */
#else
#if WSINTERFACE >= 3
int WSMMain( argc, argv) int argc; char **argv;
#else
int WSMMain( argc, argv) int argc; charpp_ct argv;
#endif /* WSINTERFACE >= 3 */
#endif
{
#if WSINTERFACE >= 3
    return _WSMain( argv, argv + argc, (char *)0);
#else
    return _WSMain( argv, argv + argc, (charp_ct)0);
#endif /* WSINTERFACE >= 3 */
}

C.2 Hard sphere trimer RISM/PRISM code

Here is the code for the athermal trimer. This is code specifically for the linear trimer version.
Listing C.7: Mathematica Linear Trimer code

(* tol is set for determination of completion of self consistent portion *)

tol = 10^-5;

(* Grid Spacing and number of points for real and k space *)

(* Linear system requires higher precision because of \(k^{-8}\) divergence conditions *)

ngrid = 2048;
delr = SetPrecision[0.1, precision];
delk = N[[Pi]/(delr*ngrid), precision];
rvvalues = Table[i*delr, {i, 1, ngrid}];
kvalues = Table[i*delk, {i, 1, ngrid}];

(* Dimer Density. Also for Hetero dimer equal to the site densities *)

rho = SetPrecision[0.002092, precision];

(* Sizes of the respective sites in \(\text{Angstrom}\) *)

sigmaa = SetPrecision[3.93, precision];
sigmab = SetPrecision[3.93, precision];

(* \(\text{Epsilon} \) for each respective site for LJ. This is not needed for Hard Sphere *)

epsilon = 0.15535;
epsilonb = 0.046;

(* Bond length *)
l = SetPrecision[3.93, precision];

(* Distance between End Sites *)
aa = 2*l;

(* Angle for the trimer. This is needed as input for the C++ code when generating the molecules *)

angle = 180.00;

(* kb is Boltzmann’s constant in kcal/K *)

kb = 0.0019833794749;

(* Temperature in Kelvin *)
t = 41.26;

(* Here we set simulation to zero after a certain point so as not to have to sample over all the grid points. Here we are using 6.5 *)

Subscript[X\{Sigma, A\}]
simcutoff = Round[sigmab*6.5];
simcutoffindex = Round[simcutoff/delr]

(* Total number of MC steps *)

mcsteps = 1024*256;

(* Respective Site Densities *)
rhoa = rho;
rhob = rho*2;
nrho = 0.38;

(* Tells which solvent potential to use in the simulation from the RISM/PRISM results *)

Model = ”HNC";
(*water stuff not needed here*)

(*qoxygen = -0.8476;*)

qhrogen = Abs[qoxygen]/2*

(*plist is the parameter list sent to the MathLink C++ code*)

plist = {sigmar, sigmar1, t, angle}

(*density matrix*)

rhom ={rhao, 0.0}, {0.0, rhob};

(*Analytical form of Oevescript \[ Omega \], \/(k) for dimers*)

omega = Table[{1, (2*Sin[dek*i*1])/(dek*i*1)}, {2*(Sin[dek*i*1]/(dek*i*1)), raa, 2*(1+Sin[dek*i*aa]/(dek*i*aa))}, {i, 1, ngrid}];

(*PRISM equations and solution potential solving*)

solvefor delta [chatk_, omegahat_] := Table[Inverse[IdentityMatrix[2] - chatk[i].(rhoa*omegahat[[i]])].chatk[i], {i, 1, ngrid}]

solveforcsc [chatk_, delta, _] := Table[chatk[i] - delta[i], {i, 1, ngrid}]

solveforhprism [omegahat, deltat] := Module[{}, Table[omegahat[i].chatk[i].deltak[i], {i, 1, ngrid}]]

wkhnc [chatk_, shatk_] := Table[chatk[i].chatk[i].chatk[i], {i, 1, ngrid}]

(*Fourier Transform functions for matrices*)

DFFSTForward[data_, delr_, kvalues_, rvalues_, transformtype_] := Module[
{datar = data, dr = delr, k=kvalues, r=rvalues, ngrid = Length[kvalues], type = transformtype, returntemp = Table[{0, 0}, {0, 0}], {i, 1, Length[kvalues]}],
Table[returntemp[[All, i, j]] = FourierDST[data[[All, i, j]]*rvalues, type]/(2 Pi) dr Sqrt[2*ngrid]
{All, i, j} = FourierDST[data[[All, i, j]]*rvalues, type]/(2 Pi) dr Sqrt[2*ngrid]*rvalues, {i, 1, 2}, {j, 1, 2};
returntemp
]

DFFSTBackward[data_, delr_, kvalues_, rvalues_, transformtype_] := Module[
{datar = data, dr = delr, k=kvalues, r=rvalues, ngrid = Length[kvalues], type = transformtype, returntemp = Table[{0, 0}, {0, 0}], {i, 1, Length[kvalues]}],
Table[returntemp[[All, i, j]] = FourierDST[data[[All, i, j]]*kvalues, type]/(2 Pi) dr Sqrt[2*ngrid]*rvalues, {i, 1, 2}, {j, 1, 2};
returntemp
]

(*numerical integration function using trapezoid method*)

trap[data_, h_] := Module[{}, h/2 (data[[1]] + Sum[2*data[[i]], {i, 2, Length[data] - 1}]) + data[[Length[data]]]]

(*performs numerical integration to solve for the zero wave vector of a function from either FFT functions or from the real space functions*)

solvefzerok [fk_, delr_] := Module[{}, trap[DFFSTBackward[fk, delr, kvalues, rvalues, 1]*rvalues^2*4*Pi], delr]]

solvefzerokp [fk_, delr_] := Module[{}, trap[DFFSTBackward[fk, delr, kvalues, rvalues, 1]*rvalues^2*4*Pi], delr]]

solvefzerokr [fr_, delr_] := Module[{}, trap[fr*rvalues^2*4*Pi], delr]]

solvefzerokpr [fr_, delr_] := Module[{}, trap[fr*rvalues^2*4*Pi], delr]]

solvefzero4 [fr_, delr_] := Module[{}, trap[fr*1/5 rvalues^6*4*Pi], delr]]

187
solvefzero6r[fr_, delr_] := Module[{r}, trap[fr*1/-7 rvalues `8*(4\[Pi]), delr]]
solvefzero8r[fr_, delr_] := Module[{r}, trap[fr*1/9 rvalues `10*(4\[Pi]), delr]]
(* functions to go from a 1x3 to a 2x2 representation*)
to2b2from3[data_] := Module[{list = data, ngrid = Length[data], temp = Table[{{0, 0}, {0, 0}}, {i, 1, Length[data]}]}, temp[All, 1] = list[All, 1]; temp[All, 2] = list[All, 2]; temp[All, 2 - 1/2] = list[All, 2];
to3b1from2b2[data_] := Module[{}, Transpose[{data[[All, 1, 1]], data[[All, 1, 2]], data[[All, 2 - 1, 2]]}]; (* Installing MathLink C++ Monte Carlo 2 chain code*)
SetDirectory[NotebookDirectory[]]
mclink = Install["WSTP_Thread_HS_Trimer"]
(* writes directories if not only present. These are where intermediate results are saved*)
If[DirectoryQ["wr"] == False, CreateDirectory["wr"], Print["wr already exists"]];
If[DirectoryQ["cr"] == False, CreateDirectory["cr"], Print["cr already exists"]];
If[DirectoryQ["hr_prism"] == False, CreateDirectory["hr_prism"], Print["hr_prism already exists"]];
If[DirectoryQ["hr_simulation"] == False, CreateDirectory["hr_simulation"], Print["hr_simulation already exists"]];
If[DirectoryQ["hr_fixed"] == False, CreateDirectory["hr_fixed"], Print["hr_fixed already exists"]];
If[DirectoryQ["ck"] == False, CreateDirectory["ck"], Print["ck already exists"]];
If[DirectoryQ["delta_ck"] == False, CreateDirectory["delta_ck"], Print["delta_ck already exists"]];
If[DirectoryQ["diff_hk"] == False, CreateDirectory["diff_hk"], Print["diff_hk already exists"]];
If[DirectoryQ["results"] == False, CreateDirectory["results"], Print["results already exists"]];
If[DirectoryQ["restart_files"] == False, CreateDirectory["restart_files"], Print["restart_files already exists"]];

(* These functions are used to check the various k^−n conditional expressions. This is used to ensure the solver is working correctly *)
k8thaa[haa_, hab_, hbb_, hba_, h2aa_, h2ab_, h2ba_, h2bb_, h4aa_, h4ab_, h4ba_, h4bb_, h6aa_, h6ab_, h6ba_, h6bb_, h8aa_, h8ab_, h8ba_, h8bb_, l_, ksmall_] := (2025 (haa−hab−hba+hbb))/(l ksmall^8)
k8thab[haa_, hab_, hbb_, hba_, h2aa_, h2ab_, h2ba_, h2bb_, h4aa_, h4ab_, h4ba_, h4bb_, h6aa_, h6ab_, h6ba_, h6bb_, h8aa_, h8ab_, h8ba_, h8bb_, l_, ksmall_] := −((2025 (haa−hba−hbb))/(2 l ksmall^8))
k8thba \[ haa, hab, hbb, hba, h2aa, h2ab, h2ba, h2bb, h4aa, h4ab, h4ba, \\
h4bb, h6aa, h6ab, h6ba, h6bb, h8aa, h8ab, h8ba, h8bb, l, ksmall \\]= -((2025 ( haa-hab-hba+hbb ))/(2 \ 1^8 \ ksmall \ 8))
k8thbb \[ haa, hab, hbb, hba, h2aa, h2ab, h2ba, h2bb, h4aa, h4ab, h4ba, \\
h4bb, h6aa, h6ab, h6ba, h6bb, h8aa, h8ab, h8ba, h8bb, l, ksmall \\]=(2025 ( haa-hab-hba+hbb ))/(4 \ 1^8 \ ksmall \ 8)
k6thaa \[ haa, hab, hbb, hba, h2aa, h2ab, h2ba, h2bb, h4aa, h4ab, h4ba, \\
h4bb, h6aa, h6ab, h6ba, h6bb, h8aa, h8ab, h8ba, h8bb, l, ksmall \\]= -((675 (-21 h2aa+21 h2ab+21 h2ba-21 h2bb+16 haa l^2-9 hab l^2-9 hba l^2+2 hbb l^2))/14 \ 1^8 \ ksmall \ 6)
k6thab \[ haa, hab, hbb, hba, h2aa, h2ab, h2ba, h2bb, h4aa, h4ab, h4ba, \\
h4bb, h6aa, h6ab, h6ba, h6bb, h8aa, h8ab, h8ba, h8bb, l, ksmall \\]=(675 (-21 h2aa+21 h2ab+21 h2ba-21 h2bb+9 haa l^2-2 hba l^2-5 hbb l^2))/28 \ 1^8 \ ksmall \ 6)
k6thba \[ haa, hab, hbb, hba, h2aa, h2ab, h2ba, h2bb, h4aa, h4ab, h4ba, \\
h4bb, h6aa, h6ab, h6ba, h6bb, h8aa, h8ab, h8ba, h8bb, l, ksmall \\]=(675 (-21 h2aa+21 h2ab+21 h2ba-21 h2bb+9 haa l^2-2 hba l^2-5 hbb l^2))/28 \ 1^8 \ ksmall \ 6)
k6thbb \[ haa, hab, hbb, hba, h2aa, h2ab, h2ba, h2bb, h4aa, h4ab, h4ba, \\
h4bb, h6aa, h6ab, h6ba, h6bb, h8aa, h8ab, h8ba, h8bb, l, ksmall \\]=-(675 (-21 h2aa+21 h2ab+21 h2ba-21 h2bb+2 haa l^2+5 hab l^2+5 hba l^2-12 hbb l^2))/56 \ 1^8 \ ksmall \ 6)
k4thaa \[ haa, hab, hbb, hba, h2aa, h2ab, h2ba, h2bb, h4aa, h4ab, h4ba, \\
h4bb, h6aa, h6ab, h6ba, h6bb, h8aa, h8ab, h8ba, h8bb, l, ksmall \\]=1/(392 \ 1^8 \ ksmall \ 4) 45 (735 h4aa-735 h4ab-735 h4ba+3360 h2aa l^2+1890 h2ab l^2+1890 h2ba l^2-420 h2bb l^2+1696 haa l^4-527 hba l^4-152 hbb l^4)
k4thab \[ haa, hab, hbb, hba, h2aa, h2ab, h2ba, h2bb, h4aa, h4ab, h4ba, \\
h4bb, h6aa, h6ab, h6ba, h6bb, h8aa, h8ab, h8ba, h8bb, l, ksmall \\]=1/(784 \ 1^8 \ ksmall \ 4) 45 (735 h4aa-735 h4ab-735 h4ba+3360 h2aa l^2+1050 h2ba l^2+1050 h2bb l^2+527 haa l^4-240 hba l^4+51 hbb l^4)
k4thba \[ haa, hab, hbb, hba, h2aa, h2ab, h2ba, h2bb, h4aa, h4ab, h4ba, \\
h4bb, h6aa, h6ab, h6ba, h6bb, h8aa, h8ab, h8ba, h8bb, l, ksmall \\]=-1/(784 \ 1^8 \ ksmall \ 4) 45 (735 h4aa-735 h4ab-735 h4ba+3360 h2aa l^2+2 hba l^2+1050 h2ba l^2+2+1050 h2bb l^2+527 haa l^4+240 hba l^4+51 hbb l^4)
k4thbb \[ haa, hab, hbb, hba, h2aa, h2ab, h2ba, h2bb, h4aa, h4ab, h4ba, \\
h4bb, h6aa, h6ab, h6ba, h6bb, h8aa, h8ab, h8ba, h8bb, l, ksmall \\]=-(1/(1568 \ 1^8 \ ksmall \ 4)) 45 (-735 h4aa+735 h4ab+735 h4ba-735 h4bb +420 h2aa l^2+1050 h2ab l^2+1050 h2ba l^2-2520 h2bb l^2+152 haa l^4+51 hba l^4-744 hbb l^4)
k2thaa \[ haa, hab, hbb, hba, h2aa, h2ab, h2ba, h2bb, h4aa, h4ab, h4ba, \\
h4bb, h6aa, h6ab, h6ba, h6bb, h8aa, h8ab, h8ba, h8bb, l, ksmall \\]=
\[-(1/(5488 \ l^8 \ ksmall^2)) \times 45 \ \times (-343 \ h6aa+343 \ h6ab+343 \ h6ba-343 \ h6bb+3920 \ h4aa \ l^2+343 \ h6ab+343 \ h6ba-343 \ h6bb+3920 \ h4aa \ l^2+4+3689 \ h2ab \ l^4+1064 \ h2bb \ l^4+2816 \ haa \ l^6+40 \ hba \ l^6-54 \ hbb \ l^6)\]

\[k2thab[haa_., hab_., hbb_., hba_., h2aa_., h2ab_., h2ba_., h2bb_., h4aa_., h4ab_., h4ba_., h4bb_., h6aa_., h6ab_., h6ba_., h6bb_., h8aa_., h8ab_., h8ba_., h8bb_., l_., ksmall_ .]:=1/(10976 \ l^8 \ ksmall^2) \times 45 \ (343 \ h6aa-343 \ h6ab-343 \ h6ba+343 \ h6bb-2205 \ h4aa \ l^2+2490 \ h4ab \ l^2+1225 \ h4bb \ l^2+3689 \ h2aa \ l^4-1680 \ h2ab \ l^4+1064 \ h2ba \ l^4+357 \ h2bb \ l^4+40 \ haa \ l^6-544 \ hba \ l^6-54 \ hbb \ l^6)\]

\[k2thba[haa_., hab_., hbb_., hba_., h2aa_., h2ab_., h2ba_., h2bb_., h4aa_., h4ab_., h4ba_., h4bb_., h6aa_., h6ab_., h6ba_., h6bb_., h8aa_., h8ab_., h8ba_., h8bb_., l_., ksmall_ .]:=-(1/(10976 \ l^8 \ ksmall^2)) \times 45 \ (343 \ h6aa-343 \ h6ab-343 \ h6ba+343 \ h6bb-2205 \ h4aa \ l^2+2490 \ h4ab \ l^2+1225 \ h4bb \ l^2+3689 \ h2aa \ l^4-490 \ h4aa \ l^2-1225 \ h4ab \ l^2+2940 \ h4bb \ l^2-1064 \ h2aa \ l^4-357 \ h2ab \ l^4+5208 \ h2bb \ l^4+54 \ haa \ l^6+226 \ hba \ l^6+768 \ hbb \ l^6)\]

\(*below uses the simulation cut off as well as the site size to calculate how large the correcting vector should be for each. This changes the size of the correcting matrix below*)

\[\begin{align*}
begaa & = \text{Round}[\sigmaaa / \text{delr}] ; \\
begab & = \text{Round}[1/2 (\sigmaaaaa-\sigmaab) / \text{delr}] ; \\
begbb & = \text{Round}[\sigmaab / \text{delr}] ; \\
\end{align*}\]

\[\text{end} = \text{simcutoffindex} ;\]

\[\begin{align*}
\text{lenaa} & = \text{Length}[\text{rvalues}[[\text{begaa};;\text{end}]]] ; \\
\text{lenab} & = \text{Length}[\text{rvalues}[[\text{begab};;\text{end}]]] ; \\
\text{lenbb} & = \text{Length}[\text{rvalues}[[\text{begbb};;\text{end}]]] ;
\end{align*}\]

\(*\text{Second Moment zero wavevectors of Overscript[\omega, ^/]}(k)*)\]

\[\begin{align*}
w2aa & = 0 ; \\
w2ab & = -(2 \times 1^2) / 3 ; \\
w2ba & = -(1^2 / 3) ; \\
w2bb & = -(\text{aa}^2 / 3) ;
\end{align*}\]

\(*h \ \text{blanks}\)*

\[\begin{align*}
hblankaam & = \text{Table}[\text{SetPrecision}[0.0, \text{precision}],[i, \text{begaa}, \text{end}]] ; \\
hblankabm & = \text{Table}[\text{SetPrecision}[0.0, \text{precision}],[i, \text{begab}, \text{end}]] ; \\
hblankbam & = \text{Table}[\text{SetPrecision}[0.0, \text{precision}],[i, \text{begab}, \text{end}]] ; \\
hblankbbm & = \text{Table}[\text{SetPrecision}[0.0, \text{precision}],[i, \text{begbb}, \text{end}]] ;
\end{align*}\]

\(*\text{zero moment}*\)

\[\begin{align*}
\text{haam} & = 4 \times [\Pi] \times \text{delr} \times \text{Table}[\text{rvalues}[[i]]^2,[i, \text{begaa}, \text{end}]] ; \\
\text{habm} & = 4 \times [\Pi] \times \text{delr} \times \text{Table}[\text{rvalues}[[i]]^2,[i, \text{begab}, \text{end}]] ;
\end{align*}\]
hbbm = 4*\[\Pi_i \ast \text{delr} \ast \text{Table} [rvalues[i] \ast 2, \{i, \text{begab}, \text{end}\}];

hbam = 4*\[\Pi_i \ast \text{delr} \ast \text{Table} [rvalues[i] \ast 2, \{i, \text{begbb}, \text{end}\}];

(\ast \text{second moment}\ast)

h2aam = 4*\[\Pi_i \ast \text{delr} \ast \text{Table} [rvalues[i] \ast 4/3, \{i, \text{begaa}, \text{end}\}];

h2abm = 4*\[\Pi_i \ast \text{delr} \ast \text{Table} [rvalues[i] \ast 4/3, \{i, \text{begab}, \text{end}\}];

h2bam = 4*\[\Pi_i \ast \text{delr} \ast \text{Table} [rvalues[i] \ast 4/3, \{i, \text{begab}, \text{end}\}];

(\ast \text{fourth moment}\ast)

h4aam = 4*\[\Pi_i \ast \text{delr} \ast \text{Table} [rvalues[i] \ast 6/5, \{i, \text{begaa}, \text{end}\}];

h4abm = 4*\[\Pi_i \ast \text{delr} \ast \text{Table} [rvalues[i] \ast 6/5, \{i, \text{begab}, \text{end}\}];

h4bam = 4*\[\Pi_i \ast \text{delr} \ast \text{Table} [rvalues[i] \ast 6/5, \{i, \text{begab}, \text{end}\}];

(\ast \text{sixth moment}\ast)

h6aam = 4*\[\Pi_i \ast \text{delr} \ast \text{Table} [rvalues[i] \ast 8/7, \{i, \text{begaa}, \text{end}\}];

h6abm = 4*\[\Pi_i \ast \text{delr} \ast \text{Table} [rvalues[i] \ast 8/7, \{i, \text{begab}, \text{end}\}];

h6bam = 4*\[\Pi_i \ast \text{delr} \ast \text{Table} [rvalues[i] \ast 8/7, \{i, \text{begbb}, \text{end}\}];

(\ast \text{eighth moment}\ast)

h8aam = 4*\[\Pi_i \ast \text{delr} \ast \text{Table} [rvalues[i] \ast 10/9, \{i, \text{begaa}, \text{end}\}];

h8abm = 4*\[\Pi_i \ast \text{delr} \ast \text{Table} [rvalues[i] \ast 10/9, \{i, \text{begab}, \text{end}\}];

h8bam = 4*\[\Pi_i \ast \text{delr} \ast \text{Table} [rvalues[i] \ast 10/9, \{i, \text{begab}, \text{end}\}];

\text{first = \text{Join} [\text{haam, \text{habm, \text{habm, \text{bbm}}}] ];

\text{second = \text{Join} [16 \text{ haaam, \text{h2aam, \text{habm}, \text{h2bbm}}];

\text{third = \text{Join} [1696 \text{ haaam, \text{h2aam, \text{h2abm, \text{h2bam, \text{h4bam}}}] ;

\text{fourth = \text{Join} [2816 \text{ haaam, \text{h2aam, \text{h2abm, \text{h2bam, \text{h4bam}}}] ;

\text{fifth = \text{Join} [9 \text{ haaam, \text{h2aam, \text{h2abm, \text{h2bam}}];

\text{sixth = \text{Join} [527 \text{ haaam, \text{h2aam, \text{h2abm, \text{h2bam, \text{h4bam}}}] ;

\text{seventh = \text{Join} [40 \text{ haaam, \text{h2aam, \text{h2abm, \text{h2bam, \text{h4bam}}}] ;

\text{eighth = \text{Join} [9 \text{ haaam, \text{h2aam, \text{h2abm, \text{h2bam}}];

\text{ninth = \text{Join} [527 \text{ haaam, \text{h2aam, \text{h2abm, \text{h2bam, \text{h4bam}}}] ;

191
tenth = Join[40 haam 1^6+3689 h2aam 1^4−2205 h4aam 1^2+343 h6aam,−54 hamb l^6+2205 h2aam 1^2−343 h4aam 1^2−1680 h2aam 1^4+490 h4aam 1^2−234 h6aam,−226 hbbm l^6+343 h2aam 1^2−343 h4aam 1^2+343 h6aam,−544 hbbm l^6+1064 h2aam 1^4+490 h4aam 1^2−343 h6aam,−343 hbbm l^6−226 hbbm l^2+357 h2aam 1^4+1225 h4aam 1^2−343 h6aam,−1680 h2aam 1^4+490 h4aam 1^2−343 h6aam,−544 hbbm l^6−226 hbbm l^2+357 h2aam 1^4+1225 h4aam 1^2−343 h6aam,−226 hbbm l^6+343 h2aam 1^2−343 h4aam 1^2+343 h6aam];
eleventh = Join[2 haam l^2−21 h2aam, 5 habm l^2+5 habm l^2+21 h2aam, 5 hbam l^2+21 h2aam,−12 hbbm l^2−21 h2bmm];
twelfth = Join[152 haam 1^4+420 h2aam 1^2−735 h4aam, 51 hamb l^4+1050 h2aam 1^2+735 h4aam,−490 h2bmm l^4−2520 h2bmm 1^2−735 h4bmm];
thirteenth = Join[54 haam 1^6−6064 h2aam 1^4−490 h4aam 1^2+343 h6aam, 226 habm 1^6−357 h2abm 1^4−1225 h4abm 1^2−343 h6abm, 226 habm 1^6−357 h2bam l^4−1225 h4bam 1^2−343 h6bam, 768 hbbm l^6+5208 h2bbm l^4+2940 h4bbm l^2+343 h6bbm];

(* the vectors above are not linearly independent below are selected the terms the give the matrix the maximum rank *)

qmatrix = {first, second, third, fourth, fifth, sixth, seventh, eighth, ninth, tenth, eleventh, twelfth, thirteenth};

(* Random Seeds given to MathLink C++... this will likely be removed to have the C++ code do this automatically using the time as a seed. This is no longer needed but included in the off chance an older version of the simulation code is used *)

randlist = RandomInteger[10000, 5000];

(* using c(r) = −1 inside the hard sphere as the first guess. This is determined for each one based on the respective sigma. Sometimes this results in problems and having c(r)=0 is a better initial guess. That is the case here. This is still included *)

craa = Join[Table[−1.0, {i, 1, Round[1/2 (sigmaa + sigmaa)/delr]}], Table[0.0, {i, 1, ngrid - Round[1/2 (sigmaa + sigmaa)/delr]}]];
crab = Join[Table[−1.0, {i, 1, Round[1/2 (sigmaa + sigmab)/delr]}], Table[0.0, {i, 1, ngrid - Round[1/2 (sigmaa + sigmab)/delr]}]];
crbb = Join[Table[−1.0, {i, 1, Round[1/2 (sigmab + sigmab)/delr]}], Table[0.0, {i, 1, ngrid - Round[1/2 (sigmab + sigmab)/delr]}]];
crhs = Table[{{craa[[i]], crab[[i]]}, {crab[[i]], crbb[[i]]}}], {i, 1, ngrid}];
crini = crhs;

(* imports the tail portion ot be pieced on from the single chain results *)

hrinputtail = SetPrecision[Import["hrfinal2.wdx"], precision];
crini = SetPrecision[Table[{{0.0, 0.0}, {0.0, 0.0}}, {ngrid}], precision];

(* sets up the lists used to save intermediate results. it also keeps track of the error to check tol for convergence. mix is important as this tells deltaC how strong to be. savestep says how often to save the results and output to the respective folders. n is used just for making the files names long enough to save their respective run number. j just keeps track of how many iterations have happened *)

192
cr = crini;
check = {};
error = {};
error2 = {};
hrfixedlist = {};
hrlist = {};
wrlist = {};
deltalist = {};
eclist = {};
debuglist = {};
mix = SetPrecision[.025, precision];
hrprismlist = {};
dhkl = {};
dhklfirstlist = {};
save = 20;
n = 10000;
err = 20;
j = 0;
(* Intermediate results are saved every step so it can be restart. Below
used to import these in the case the code needs to be restarted*)
(*cr=ToExpression[Import[Directory[]<>"/restartfiles/cr.tsv","Data"];
dhklfirstlist=ToExpression[Import[Directory[]<>"/restartfiles/
dhklfirstlist.tsv","Data"];
error=Flatten[ToExpression[Import[Directory[]<>"/restartfiles/error.tsv
","Data"]];
j = ToExpression[Import[Directory[]<>"/restartfiles/n.tsv","Data
"]]/[1,1]−1;*)

ckmix = DFFSTForward[cr, delr, kvalues, rvalues, 1];
(* start of the loop*)

While[err > tol, ++j;
ck = ckmix;
ckold = ck;
crold = cr;
delta = solvefor delta [ck, omega];
hkprism = solveforhkprism [omega, delta];
If[IntegerQ[j / save], hrprism = DFFSTBackward[hkprism, delr, kvalues,
rvalues, 1]; Export[Directory[]<>"/hr_prism/hrprism"<>IntegerString[j
, 10, IntegerLength[n]]<>".tsv", hrprism]]; Which[Model == "PY", wr =−kb*t*Log[1.0 + DFFSTBackward[solveforcsc [ck, delta]
delr, kvalues, rvalues, 1]]; , Model == "HNC", wr = kb*t*DFFSTBackward[
solveforcsc [ck, delta], delr, kvalues, rvalues, 1]; , Model == "MS", wr = kb*t*
−Sqrt[1 + 2*DFFSTBackward[solveforcsc [ck, delta], delr, kvalues, rvalues, 1];  ,
Model == "HY", wr = kb*t*DFFSTBackward[solveforcsc [ck, delta], delr, kvalues, rvalues, 1];]
If [IntegerQ[j/savestep], Export[Directory[] <> "\/wr/wr" <> IntegerString[j, 10, IntegerLength[n]] <> ".tsv", wr]]; (*grun is the actual simulation and will give an output list that is then used by gr1, gr2 etc.*)
grun = SetPrecision[wst pthreadstrimer[wr[[All, 1, 1]], wr[[All, 1, 2]], wr[[All, 2, 2]], plist, 1.020000, 20000, msteps, 17, 320], precision]; (*the tail is pieced on here*)
gr1 = Join[Table[grun[[i + 1, 1]] / (msteps*(i)^2), {i, 1, simcutoffindex}], Table[hrinputtail[[i, 1, 1]] + 1, {i, simcutoffindex + 1, ngrid}]];
gr2 = Join[Table[grun[[i + 1, 2]] / (4*msteps*i^2), {i, 1, simcutoffindex}], Table[hrinputtail[[i, 1, 2]] + 1, {i, simcutoffindex + 1, ngrid}]];
gr3 = Join[Table[grun[[i + 1, 3]] / (4*msteps*i^2), {i, 1, simcutoffindex}], Table[hrinputtail[[i, 2, 2]] + 1, {i, simcutoffindex + 1, ngrid}]];
hr1 = gr1 - 1;
hr2 = gr2 - 1;
hr3 = gr3 - 1;
hr1new = hr1;
hr2new = hr2;
hr3new = hr3;
hr4new = hr4;
hrnew = Table[{{0, 0}, {0, 0}, {i, 1, ngrid}}];
hrnew[[All, 1, 1]] = hr1;
hrnew[[All, 1, 2]] = hr2;
hrnew[[All, 2, 1]] = hr3;
hrnew[[All, 2, 2]] = hr4;
If [IntegerQ[j/savestep], Export[Directory[] <> "/hr_simulation/hrsim" <> IntegerString[j, 10, IntegerLength[n]] <> ".tsv", hrnew]];
hktemp = hrnew;
hrnew = hktemp;
hksim = DFFSTForward[hrnew, delr, kvalues, rvalues, 1]; (*calculates hat_h(0), hat_h'(0) etc by integrating*)
{h4aa, h4ab, h6aa, h6ab} = solvefzero[hrnew, delr];
{h2aa, h2ab} = solvefzeroppr[hrnew, delr];
\[ \begin{align*}
&\hat{\mathbf{4}} - 2205 \ h_{4ba} \ l^2 + 343 \ h_{6ba} - 54 \ h_{6bb} + 1^6 + 1064 \ h_{2bb} + l^4 + 490 \ h_{4bb} + l^2 - 343 \\
&\mathbf{h}_{6bb} ) ; \\
rhs_5 &= -(9 \ h_{aa} \ l^2 - 21 \ h_{2aa} - 2 \ h_{ab} - 2 \ h_{ba} - 2 \ h_{2ba} - 5 \ h_{bb} + 1 \\
&l^2 - 21 \ h_{2bb} ) ; \\
rhs_6 &= -(527 \ h_{aa} \ l^2 - 1890 \ h_{2aa} \ l^2 + 735 \ h_{4aa} - 240 \ h_{ab} \ l^2 - 1^4 + 420 \ h_{2ab} \ l^2 - 735 \\
&h_{4ab} + 152 \ h_{ba} \ l^2 + 735 \ h_{4ba} + 51 \ h_{bb} + 1^4 + 1050 \ h_{2bb} \ l^2 + 735 \\
&h_{4bb} ) ; \\
rhs_7 &= -(40 \ h_{aa} \ l^6 + 3689 \ h_{2aa} \ l^2 + 4 - 2205 \ h_{4aa} \ l^2 + 343 \ h_{6aa} - 544 \ h_{ab} \ l^6 - 1^6 - 1680 \\
&h_{2ab} \ l^4 + 490 \ h_{4ab} \ l^2 - 343 \ h_{6ab} - 544 \ h_{ba} \ l^6 + 1064 \ h_{2ba} \ l^4 + 490 \ h_{4ba} \ l^2 \\
&l^2 - 343 \ h_{6ba} - 226 \ h_{2bb} \ l^4 + 343 \ h_{6bb} + 1225 \ h_{4bb} \ l^2 ) ; \\
rhs_8 &= -(9 \ h_{aa} \ l^2 - 21 \ h_{2aa} - 2 \ h_{ab} - 2 \ h_{ba} - 2 \ h_{2ba} - 5 \ h_{bb} + 1 \\
&l^2 - 21 \ h_{2bb} ) ; \\
rhs_9 &= -(527 \ h_{aa} \ l^2 - 1890 \ h_{2aa} \ l^2 + 735 \ h_{4aa} + 152 \ h_{ab} \ l^4 + 420 \ h_{2ab} \ l^2 \\
&- 735 \ h_{4ab} - 240 \ h_{ba} \ l^4 + 420 \ h_{2ba} \ l^2 - 735 \ h_{4ba} + 51 \ h_{bb} \ l^4 + 1050 \ h_{2bb} \ l^2 \\
&+ 735 \ h_{4bb} ) ; \\
rhs_{10} &= -(40 \ h_{aa} \ l^6 + 3689 \ h_{2aa} \ l^2 - 2205 \ h_{4aa} \ l^2 + 343 \ h_{6aa} - 544 \ h_{ab} \ l^6 - 1^6 - 1680 \\
&h_{2ab} \ l^4 + 490 \ h_{4ab} \ l^2 - 343 \ h_{6ab} - 544 \ h_{ba} \ l^6 + 1064 \ h_{2ba} \ l^4 + 490 \ h_{4ba} \ l^2 \\
&l^2 - 343 \ h_{6ba} - 226 \ h_{2bb} \ l^4 + 343 \ h_{6bb} + 1225 \ h_{4bb} \ l^2 ) ; \\
rhs_{11} &= -(2 \ h_{aa} \ l^2 - 21 \ h_{2aa} + 5 \ h_{ab} \ l^4 + 420 \ h_{2ab} + 5 \ h_{ba} \ l^2 + 21 \ h_{2ba} - 12 \ h_{bb} + 1 \\
&l^2 - 21 \ h_{2bb} ) ; \\
rhs_{12} &= -(152 \ h_{aa} \ l^2 - 4 + 420 \ h_{2aa} \ l^2 - 21 \ h_{2ba} - 735 \ h_{4aa} + 51 \ h_{ab} \ l^4 + 4 + 1050 \ h_{2ab} \ l^2 + 735 \\
&h_{4ab} + 51 \ h_{ba} \ l^4 + 4 + 1050 \ h_{2ba} \ l^2 - 735 \ h_{4ba} - 744 \ h_{bb} \ l^4 - 2520 \ h_{2bb} \ l^2 \\
&l^2 - 735 \ h_{4bb} ) ; \\
rhs_{13} &= -(54 \ h_{aa} \ l^6 - 1064 \ h_{2aa} \ l^2 - 4 + 490 \ h_{4aa} \ l^2 + 735 \ h_{6aa} + 226 \ h_{ab} \ l^6 - 357 \\
&h_{2ab} \ l^4 + 1225 \ h_{4ab} \ l^2 - 343 \ h_{6ab} + 226 \ h_{ba} \ l^6 - 357 \ h_{2ba} \ l^4 + 1225 \ h_{4ba} \ l^2 \\
&l^2 - 343 \ h_{6ba} + 768 \ h_{2bb} \ l^4 + 5208 \ h_{2bb} \ l^4 + 2940 \ h_{4bb} \ l^2 + 343 \ h_{6bb} ) ; \\
\end{align*}
\]

\[(\text{use the respective RHS terms that correspond with the qmatrix from above})\]

cvector = \{\text{rhs1, rhs3, rhs4, rhs7, rhs9, rhs11, rhs12, rhs13}\};
thing = \text{PseudoInverse}[\text{qmatrix}].cvector;

\[(\text{The following will apply the tweaking vector})*\]

hr1fixed = hr1;
hr1fixed[[begaa ;; end]] = hr1fixed[[begaa ;; end]] + thing[[1 ;; lenaa]];
hr2fixed = hr2;
hr2fixed[[begab ;; end]] = hr2fixed[[begab ;; end]] + thing[[1 + lenaa ;; lenaa+lenab]];
hr3fixed = hr3;
hr3fixed[[begab ;; end]] = hr3fixed[[begab ;; end]] + thing[[1 + lenaa+lenab ;; lenaa+lenab+lenab]];
hr4fixed = hr4;
hr4fixed[[begbb ;; end]] = hr4fixed[[begbb ;; end]] + thing[[1 + lenaa+lenab+lenab+lenab+lenab+lenbb]]; 
hrfixed = Table[{{0, 0}, {0, 0}}, \{i, 1, ngrid\}];
hrfixed[[\text{All}, 1, 1]] = hr1fixed;
hrfixed[[\text{All}, 1, 2]] = hr2fixed;
hrfixed[[\text{All}, 2, 1]] = hr3fixed;
hrfixed[[\text{All}, 2, 2]] = hr4fixed;
hr=hrfixed;
If [IntegerQ[j/savestep], Export[Directory[]<>"/hr_fixed/hrfixed"<>
  IntegerString[j,10,IntegerLength[n]]<>".tsv",hr]];

hknw = DFFSTForward[hr, delr, kvalues, rvalues, 1];
dhk = Table[hksim[[i]]−hknw[[i]],{i,1,ngrid}];
AppendTo[dhkfistlist,dhk[[1]]];
deltakh = Table[{{1,2},{2,4}}*hknw[[i]]−{{1,2},{2,4}}*hksim[[i]],{i,
  1,ngrid}];
deltack = Table[Inverse[omega[[i]]].{{1,2},{2,4}}*hknw[[i]].Inverse[omega[[i]]]−
  (−deltakh[[i]],{i,1,ngrid}];
If [IntegerQ[j/savestep], Export[Directory[]<>"/diff_hk/diffhk"<>
  IntegerString[j,10,IntegerLength[n]]<>".tsv",dhk]];
If [IntegerQ[j/savestep], Export[Directory[]<>"/delta_ck/deltack"<>
  IntegerString[j,10,IntegerLength[n]]<>".tsv",deltack]];
(* calculate new C*)
ckmix = ckold +mix*deltack;
cr=DFFSTBackward[ckmix, delr, kvalues, rvalues, 1]; If [IntegerQ[j/savestep],
  Export[Directory[]<>"/ck/cknew"<>IntegerString[j,10,IntegerLength[n]]<>".tsv",
  cmix]]; If [IntegerQ[j/savestep], Export[Directory[]<>"/cr/cr"<>IntegerString[j,
  10,IntegerLength[n]]<>".tsv",cr]]; (* check for convergence*)
err=Total[{|Sum[Flatten[(cr[[i]]−ckold[[i]])^2]],{i,1,ngrid}}]];
AppendTo[error, err];

{[h aa,h ab},{h ba,h bb}]=solvefzeror[hrfixed,delr];
{h2aa,h2ab},{h2ba,h2bb}=solvefzeroppr[hrfixed,delr];
{h4aa,h4ab},{h4ba,h4bb}=solvefzero4r[hrfixed,delr];
{h6aa,h6ab},{h6ba,h6bb}=solvefzero6r[hrfixed,delr];
{h8aa,h8ab},{h8ba,h8bb}=solvefzero8r[hrfixed,delr];

(*hk test = DFFSTForward[hrnew, delr, kvalues, rvalues, 1]*) res = {
  Iteration,=ω<>toString[j],"1/k^8. term","1/k^6. term","1/k^4. term","1/k^2. term","1/k^2. term","1/k^8+1/k^6+1/k^4+1/k^2"},"Subscript[ Oberscript[[
  Delta][C, ω], AA](Subscript[k, 1])"], a=k8thaa[h aa,h ab,h bb,h ba,h2aa,
  h2ab,h2ba,h2bb,h4aa,h4ab,h4ba,h4bb,h6aa,h6ab,h6ba,h6bb,h8aa,h8ab,
  h8ba,h8bb,1,kvalues[[1]]], b=k6thaa[h aa,h ab,h bb,h ba,h2aa,h2ab,h2ba,
  h2bb,h4aa,h4ab,h4ba,h4bb,h6aa,h6ab,h6ba,h6bb,h8aa,h8ab,h8ba,h8bb,1,
  kvalues[[1]]], c=k4thaa[h aa,h ab,h bb,h ba,h2aa,h2ab,h2ba,h2bb,h4aa,h4ab,
  h4ba,h4bb,h6aa,h6ab,h6ba,h6bb,h8aa,h8ab,h8ba,h8bb,1,kvalues[[1]]], d=k2thaa[h
  aa,h ab,h bb,h ba,h2aa,h2ab,h2ba,h2bb,h4aa,h4ab,h4ba,h4bb,h6aa,h6ab,
  h6ba,h6bb,h8aa,h8ab,h8ba,h8bb,1,kvalues[[1]]], a+b+c+d, {"Subscript[ Oberscript[[
  Delta][C, ω], AB](Subscript[k, 1])"], a=k8thab[h
  aa,h ab,h bb,h ba,h2aa,h2ab,h2ba,h2bb,h4aa,h4ab,h4ba,h4bb,h6aa,h6ab,
  h6ba,h6bb,h8aa,h8ab,h8ba,h8bb,1,kvalues[[1]]], b=k6thab[h aa,h ab,h bb,
  h ba,h2aa,h2ab,h2ba,h2bb,h4aa,h4ab,h4ba,h4bb,h6aa,h6ab,h6ba,h6bb,h8aa,
  h8ab,h8ba,h8bb,1,kvalues[[1]]], c=k4thab[h aa,h ab,h bb,h ba,h2aa,h2ab,
  h2ba,h2bb,h4aa,h4ab,h4ba,h4bb,h6aa,h6ab,h6ba,h6bb,h8aa,h8ab,h8ba,
  h8bb,1,kvalues[[1]]], d=k2thab[h aa,h ab,h bb,h ba,h2aa,h2ab,h2ba,h2bb,
C.3 Water RISM/PRISM code

The following code is for the Two-Molecule RISM theory for water. This includes the modifications allowing for the $k^{-2}$ divergence that is controlled by the dielectric with the tail fit.

Listing C.8: Mathematica Linear Trimer code

tol = 10^-5;
SetDirectory[NotebookDirectory[]];
(* Grid Spacing and number of points for real and k space *)
ngrid = 2048;
delr = 0.1;
delk = N[Pi]/(delr*ngrid); rvalues = Table[i*delr, {i, 1, ngrid}]; kvalues = Table[i*delk, {i, 1, ngrid}];
(* Dimer Density. Also for Hetero dimer equal to the site densities *)
rho = .033329;
(* Sizes of the respective sites in \[\text{Angstrom}\] *)
sigmaa = 3.166;
sigmab = .4;
(* \[\text{Epsilon} \] for each respective site for LJ *)
epsilon_a = 0.15535;
epsilon_b = 0.046;
(* Bond length *)
l = 1.00;
(* distance between hydrogens *)
aa = 2*0.816493;
(* kb is Boltzmann's constant in Cal/K *)
kb = 0.0019833794749;
(* temperature in Kelvin *)
t = 298;
(* Here we set simulation to zero after a certain point so as not to have to sample over all the grid points. Here we are using 4 Subscript[X\[\text{Sigma}, A\]] *)
simcutoff = sigmaa*5;
simcutoffindex = Round[simcutoff/delr]
(* total number of MC steps *)
mcsteps = 1024*256;
(* Respective Site Densities, *)
rhoa = rho;
rhob = rho*2;
rho = 0.9;
Model = "HNC";
qoxygen = -0.82;
qhydrogen = Abs[qoxygen]/2
electron = 4.803*^-10;
zo = qoxygen*(1^12)*electron;
zh = qhydrogen*(1^12)*electron;
epslondielectric = 80.4;
betafit = 1/(t*1.381);
(* plist is the parameter list sent to the MathLink C++ code the final term tells how strong to allow the coulomb to be *)
plist = {sigmaa, sigmab, l, t, epsilon_a, epsilon_b, qoxygen, qhydrogen, 1.0}
(* density matrix *)
rhom = (ρa₀, 0.0), (0.0, ρb₀);
(* Analytical form of Overscript \([\Omega], \^ \(k\) for dimers*)
omega = Table[{{1, 2*Sin[delk*i]}, {2*Sin[delk*i]}, {i, 1, ngird}};
(*PRISM equations and solution potential solving*)
solvefordelta[chatk_, omegahat_] := Table[Inverse[IdentityMatrix[2] -
chatk[[i]]* (rhoa*omegahat[[i]])] * chatk[[i]], {i, 1, ngird}]
solveforcsc[chatk_, delta_] := Table[-chatk[[i]] - delta[[i]], {i, 1, ngird}]
solveforhprism[omegak_, deltk_] := Module[{}, Table[omegak[[i]], {i, 1, ngird}]
wkhnc[chatk_, shatk_] := Table[chatk[[i]] * shatk[[i]] * chatk[[i]], {i, 1, ngird}]

(* Fourier Transform functions for matrices*)
DFFSTForward[data_, delr_, kvalues_, rvalues_, transformtype_] := Module[
{data = data, dr = delr, k = kvalues, r = rvalues, ngrid = Length[kvalues],

type = transformtype, returntemp = Table[{{0, 0}, {0, 0}}, {i, 1, Length[kvalues]}]},
Table[returntemp[[All, i, j]] = 2*i*Pi*dr*Sqrt[2*ngrid]
FourierDST[data[[All, i, j]]*rvalues, type]/kvalues, {i, 1, 2}, {j, 1, 2}];

DFFSTBackward[data_, delr_, kvalues_, rvalues_, transformtype_] := Module[
{data = data, dr = delr, k = kvalues, r = rvalues, ngrid = Length[kvalues],

type = transformtype, returntemp = Table[{{0, 0}, {0, 0}}, {i, 1, Length[kvalues]}]},
Table[returntemp[[All, i, j]] = FourierDST[data[[All, i, j]]*rvalues, type]/2*i*Pi*dr*Sqrt[2*ngrid]*rvalues, {i, 1, 2}, {j, 1, 2}];

(*numerical integration function using trapezoid method trap 2 is if*
the tail is being added because you won’t use all the simulation
points.*)
trap[data_, h_, start_, end_] := Module[{}, h/2 (data[[start]] + Sum[2*data[[i]], {i, start + 1, end - 1}] + data[[end]])

trap2[data_, h_, start_, end_] := Module[{}, h/2 (data[[start]] + Sum[2*data[[i]], {i, start + 1, end}])

traptail[data_, h_, start_, end_] := Module[{}, h/2 (Sum[2*data[[i]], {i, start, end - 1}] + data[[end]])]

(*performs numerical integration to solve for the zero wave vector of a
function from either FFT functions or from the real space functions*)
solvefzerok[fk_, delr_, start_, end_] := Module[{r}, trap2[DFFSTBackward[fk, delr, kvalues, rvalues, 1]*rvalues^2*4*i*Pi, delr, start, end]]
solvefzeroopk[fk_, delr_, start_, end_] := Module[{r}, trap2[DFFSTBackward[fk, delr, kvalues, rvalues, 1]*i - 3 rvalues^2*4*Pi, delr, start, end]]
solvefzerofull[fr_, delr_, start_, end_] := Module[{r}, trap2[fr*rvalues^2*4*i*Pi, delr, start, end]]
solvefzeror[fr_, delr_, start_, end_] := Module[{r}, trap2[fr*rvalues^2*4*i*Pi, delr, start, end]]
solvezeroppr[fr_, delr_, start_, end_] := Module[{r},
  trap2[fr*1/3, rvalues^4*(4\Pi), delr, start, end]]
solvezeropprefull[fr_, delr_, start_, end_] := Module[{r},
  trap[fr*1/3, rvalues^4*(4\Pi), delr, start, end]]
solvezerortail[fr_, delr_, start_, end_] := Module[{r},
  traptail[fr*rvalues^2*4\Pi, delr, start, end]]

(*functions to go from a 1x3 to a 2x2 representation*)
to2b2from3[data_] := Module[{list = data, ngrid = Length[data], temp = Table[{{0, 0}, {0, 0}}, {i, 1, Length[data]}],
  temp[[All, 1]] = list[[All, 1]]; temp[[All, 2]] = list[[All, 2]]; temp[[All, 3]] = list[[All, 3]]; temp],
  Transpose[{data[[All, 1]], data[[All, 2]]}]);

(*Installing MathLink C++ Monte Carlo 2 chain code*)
SetDirectory[NotebookDirectory[]];
mclink = Install["WSTP_Thread_Water"];
"results already exists"

k4thab [haa_, hab_, hbb_, hba_, h2aa_, h2ab_, h2ba_, h2bb_, w2aa_, w2ab_, w2ba_,
w2bb_, gaa_, gab_, gba_, gbb_, g2aa_, g2ab_, g2ba_, g2bb_, ksmall_] := (16 (haa
-hab-hba+hbb))/(2 w2aa-2 w2ab-2 w2ba+2 w2bb) 2 ksmall 4

k4thbb [haa_, hab_, hbb_, hba_, h2aa_, h2ab_, h2ba_, h2bb_, w2aa_, w2ab_, w2ba_,
w2bb_, gaa_, gab_, gba_, gbb_, g2aa_, g2ab_, g2ba_, g2bb_, ksmall_] := -(8 (haa
-hab-hba+hbb))/(2 w2aa-2 w2ab-2 w2ba+2 w2bb) 2 ksmall 4

k4thba [haa_, hab_, hbb_, hba_, h2aa_, h2ab_, h2ba_, h2bb_, w2aa_, w2ab_, w2ba_,
w2bb_, gaa_, gab_, gba_, gbb_, g2aa_, g2ab_, g2ba_, g2bb_, ksmall_] := -(8 (haa
-hab-hba+hbb))/(2 w2aa-2 w2ab-2 w2ba+2 w2bb) 2 ksmall 4

k2tha[ haa_, hab_, hbb_, hba_, h2aa_, h2ab_, h2ba_, h2bb_, w2aa_, w2ab_, w2ba_,
w2bb_, gaa_, gab_, gba_, gbb_, g2aa_, g2ab_, g2ba_, g2bb_, ksmall_] := (1/(ksmall 2) 2 ((16 h2aa-16 h2ab-16 h2ba+16 h2bb-8 hba w2ab+8 hbb w2ab
-16 hbb w2ba+16 hbb w2ba+16 haa w2bb-8 hba w2bb-8 hbb w2bb)/(-4 w2aa
+2 w2ab-2 w2ba-2 w2bb)) 2 -(64 (haa-hab-hba+hbb) (w2ab w2ba-w2aa w2bb)
)/(-4 w2aa+2 w2ab-2 w2ba-2 w2bb) 3)

k2thab[ haa_, hab_, hbb_, hba_, h2aa_, h2ab_, h2ba_, h2bb_, w2aa_, w2ab_, w2ba_,
w2bb_, gaa_, gab_, gba_, gbb_, g2aa_, g2ab_, g2ba_, g2bb_, ksmall_] := (8 (h2aa
-2 h2ab-2 h2ba+2 h2bb-2 hab w2aa+2 hbb w2ab-2 hab w2ab+wbb-2 hab w2bb-2 hab w2ba+hbb w2bb)/(4 w2aa+2 w2ab+2 w2ba-2 w2bb) 2 + (64 (haa-hab-hba+hbb) (w2ab w2ba-w2aa w2bb))/(4 w2aa+2 w2ab+4 w2ba-2 w2bb) 3)/ksmall 2

k2thba[ haa_, hab_, hbb_, hba_, h2aa_, h2ab_, h2ba_, h2bb_, w2aa_, w2ab_, w2ba_,
w2bb_, gaa_, gab_, gba_, gbb_, g2aa_, g2ab_, g2ba_, g2bb_, ksmall_] := -(8 (h2aa
-8 h2ab-8 h2ba+8 h2bb-8 hba w2ba+8 hbb w2ba+8 haa w2ba)/(4 w2aa+4 w2ab
+4 w2ba-2 w2bb) 2 -(32 (haa-hab-hba+hbb) (w2ab w2ba-w2aa w2bb))/(4 w2aa+2 w2ab+4 w2ba-2 w2bb) 3)/ksmall 2

k2thbb[ haa_, hab_, hbb_, hba_, h2aa_, h2ab_, h2ba_, h2bb_, w2aa_, w2ab_, w2ba_,
w2bb_, gaa_, gab_, gba_, gbb_, g2aa_, g2ab_, g2ba_, g2bb_, ksmall_] := (8 (h2aa
-8 h2ab-8 h2ba+8 h2bb-8 hba w2aa-8 hba w2ba+16 hbb w2aa+16 hba w2ba-4
hba w2ab-8 hba w2ba-8 hbb w2ba)/(4 w2aa+2 w2ab+2 w2ba-2 w2bb) 2 -(32 (haa-hab-hba+hbb) (w2ab w2ba-w2aa w2bb))/(4 w2aa+2 w2ab+2 w2ba-2 w2bb) 3)/ksmall 2

k0thab [haa_, hab_, hbb_, hba_, h2aa_, h2ab_, h2ba_, h2bb_, w2aa_, w2ab_, w2ba_,
w2bb_, gaa_, gab_, gba_, gbb_, g2aa_, g2ab_, g2ba_, g2bb_, ksmall_] := 2 (-((2
(16 h2aa-16 h2ab-16 h2ba+16 h2bb-8 hba w2ab+8 hbb w2ab-16 hab w2ab
+16 hbb w2ba-16 haa w2bb-8 hab w2bb-8 hba w2bb) (w2ab w2ba-w2aa w2bb)
)/(-4 w2aa+2 w2ab+4 w2ba-2 w2bb) 3) + (96 (haa-hab-hba+hbb) (w2ab
w2ba-w2aa w2bb) 2 ) /(-4 w2aa+2 w2ab+4 w2ba-2 w2bb) 4 + (4 h2ba w2ab+4
h2bb w2ab-8 h2ab w2ab+8 h2bb w2ba+4 hbb w2ab w2ba+8 h2aa w2bb-4 h2ab
w2bb-4 h2ba w2bb-2 hba w2ab w2bb-4 hab w2ba w2bb+2 haa w2bb 2 ) /(-4
w2aa+2 w2ab+4 w2ba-2 w2bb) 2)}
\[ k_{0thab}[h_{aa}, h_{ab}, h_{bb}, h_{ba}, h_{2aa}, h_{2ab}, h_{2bb}, w_{2aa}, w_{2ab}, w_{2ba}, w_{2bb}, g_{aa}, g_{ab}, g_{ba}, g_{bb}, g_{2aa}, g_{2ab}, g_{2ba}, g_{2bb}, k_{small}]:=(16(2h_{2aa}-2h_{2ab}+2h_{2bb}-2hab+2h_{2ab}+h_{bb}w_{2ab}+h_{2ba}w_{2aa}w_{2bb}))/(4w_{2aa}+2w_{2ab}+4w_{2ba}+w_{2bb})^3-(96(h_{aa}+h_{bb}+h_{ab}+h_{ba})(w_{2ab}w_{2ba}+w_{2aa}w_{2bb}+h_{aa}w_{2ab}+h_{bb}w_{2ab}+h_{ba}w_{2ba}-w_{2aa}w_{2bb})^2)/(4w_{2aa}+2w_{2ab}+4w_{2ba}+w_{2bb})^4+(8(h_{aa}+h_{ab}+h_{ba}+h_{bb})(w_{2ab}w_{2ba}+w_{2aa}w_{2bb})^2)/(4w_{2aa}+2w_{2ab}+4w_{2ba}+w_{2bb})^4 ) \\
\]

\[ k_{0thba}[h_{aa}, h_{ab}, h_{bb}, h_{ba}, h_{2aa}, h_{2ab}, h_{2bb}, w_{2aa}, w_{2ab}, w_{2ba}, w_{2bb}, g_{aa}, g_{ab}, g_{ba}, g_{bb}, g_{2aa}, g_{2ab}, g_{2ba}, g_{2bb}, k_{small}]:=-2((-2(8h_{2aa}-8h_{2ab}-8h_{2bb}+8h_{ba}w_{2aa}-8h_{bb}w_{2aa}+16h_{ba}w_{2ba}+8h_{bb}w_{2bb})))/(4w_{2aa}+2w_{2ab}+4w_{2ba}+w_{2bb})^3+(48(h_{aa}+h_{ab}+h_{ba}+h_{bb})(w_{2ab}w_{2ba}+w_{2aa}w_{2bb})^2)/(4w_{2aa}+2w_{2ab}+4w_{2ba}+w_{2bb})^4 ) \\
\]

\[ k_{0thbb}[h_{aa}, h_{ab}, h_{bb}, h_{ba}, h_{2aa}, h_{2ab}, h_{2bb}, w_{2aa}, w_{2ab}, w_{2ba}, w_{2bb}, g_{aa}, g_{ab}, g_{ba}, g_{bb}, g_{2aa}, g_{2ab}, g_{2ba}, g_{2bb}, k_{small}]:=(16(2h_{2aa}-2h_{2ab}-2h_{2bb}+2hab+2h_{2ab}+h_{bb}w_{2ab}+h_{2ba}w_{2aa}w_{2bb}))/(4w_{2aa}+2w_{2ab}+4w_{2ba}+w_{2bb})^2 ) \\
\]

(*below uses the simulation cut off as well as the site size to calculate how large the correcting vector should be for each. This changes the size of the correcting matrix below *)

\text{begaa} = \text{Round}[(\sigma_{aa}/deltar) \]
\text{begab} = \text{Round}[(1/2(\sigma_{aa}+\sigma_{ab})/deltar) \]
\text{begbb} = \text{Round}[(\sigma_{ab}/deltar) \]
\text{end} = simcutoffindex;  
\text{lenaa}=\text{Length}[rvalues[[\text{begaa};;\text{end}]]]; 
\text{lenab}=\text{Length}[rvalues[[\text{begab};;\text{end}]]]; 
\text{lenbb}=\text{Length}[rvalues[[\text{begbb};;\text{end}]]];  
(* tfp is the number of tail fitting points 
 tfs is the tail fit start index 
 tfsp is the tail fit stop index*) 
\text{tfp} = 50; 
\text{tfs} = \text{end}-\text{tfp}; 
\text{tfsp} = \text{end};  
(* tail output...here is where we start applying the tail 
 t0s = \text{end}+1; 
 t0sp = ngrid;  
(* Second Moment zero wavevectors of Overscript [\Omega, \hata](k)*)
\[ h_{2ab} = -\frac{(2^*1^*2)}{3}; \]
\[ w_{2bb} = -\frac{(aa^*2)}{3}; \]
\[ w_{2aa} = 0; \]
\[ h_{2am} = 4 \times h_{2bbm} = 4 \times h_{2bam}; \]
\[ h_{2aam} = 4 \times h_{2abm}; \]
\[ \pi = h_{2tail} = 4; \]
\[ w_{2ab} = w_{2ba} = w_{2aa} = 0; \]
\[ \left\{ \begin{align*}
(\text{second} = 1/2 \times \text{craa} = \text{Join}[
\text{fourth} = 1/2 \times \text{craa} = \text{Join}[
\text{third} = \text{Join}[
\text{first} = \text{Join}[
\text{fifth} = \text{Join}[
\text{sixth} = \text{Join}[
\end{align*} \right. \]
\[ \text{qmatrix} = \{ \text{first} , \text{second} , \text{third} , \text{fourth} , \text{fifth} , \text{sixth} \}; \]

\textit{(Random Seeds given to MathLink C++... this will likely be removed to have the C++ code do this automatically using the time as a seed*)}
\textit{randlist = RandomInteger[10000,5000]; (*) using } c(r) = -1 \text{ inside the hard sphere as the first guess. This is determined for each one based on the respective } \sigma_{i} \text{ *)}
crab = Join[Table[-1.0, {i, 1, Round[1/2 (sigmaa+sigmab)/delr]}], Table[0.0, {i, 1, ngrid-Round[1/2 (sigmaa-sigmab)/delr]}]]
crbb = Join[Table[-1.0, {i, 1, Round[1/2 (sigmab-sigmab)/delr]}], Table[0.0, {i, 1, ngrid-Round[1/2 (sigmab+sigmab)/delr]}]]

crhs = Table[{{craa[[i]], crab[[i]]}, {crab[[i]], crbb[[i]]}}, {i, 1, ngrid}];

 crini = crhs;

crini = Table[{{0.0, 0.0}, {0.0, 0.0}}, {ngrid}];

hrinputtail = crini;

cr = crini;

check = {};

error = {};

error2 = {};

hrfixedlist = {};

hrlist = {};

wrlist = {};

deltalist = {};

eklist = {};

debuglist = {};

mix = .005;

hrprismlist = {};

dhklist = {};

dhkfirstlist = {};

savestep = 20;

n = 10000;

err = 20;

j = 0;

While[err > tol, ++j;

cr = DFFSTForward[cr, delr, kvalues, rvalues, 1];

crold = cr;

delta = solvefordelta[ck, omega];

hkprism = solveforhkprism[omega, delta];

hrprism = DFFSTBackward[hkprism, delr, kvalues, rvalues, 1];

If[IntegerQ[j/savestep], Export[Directory[] <> "/hr_prism/hrprism" <> IntegerString[j, 10, IntegerLength[n]] <> ".tsv", hrprism]];

Which[Model \[Equal\] "PY", wr = kb*t*Log[1.0 + DFFSTBackward[solveforcsc[ck, delta], delr, kvalues, rvalues, 1]]; Model \[Equal\] "HNC", wr = kb*t*

DFFSTBackward[solveforcsc[ck, delta], delr, kvalues, rvalues, 1]; Model \[Equal\] "MS", wr = kb*t*(-Sqrt[1 + 2*DFFSTBackward[solveforcsc[ck, delta], delr, kvalues, rvalues, 1]] + 1);]

If[IntegerQ[j/savestep], Export[Directory[] <> "/wr/wr" <> IntegerString[j, 10, IntegerLength[n]] <> ".tsv", wr]];

grun = wstpthreadwater[wr[All, 1, 1], wr[All, 1, 2], wr[All, 2, 2], plist, 1.0, 20000, 20000, mcsteps, 17, 320];

204
gr1 = Join[Table[grun[[i + 1, 1]]/(msteps*(i)^2), {i, 1, simcutoffindex}], Table[1.0, {i, simcutoffindex + 1, ngrid}]];
gr2 = Join[Table[grun[[i + 1, 2]]/(4*msteps*i^2), {i, 1, simcutoffindex}], Table[1.0, {i, simcutoffindex + 1, ngrid}]];
gr3 = Join[Table[grun[[i + 1, 3]]/(4*msteps*i^2), {i, 1, simcutoffindex}], Table[1.0, {i, simcutoffindex + 1, ngrid}]];
hr1 = gr1 - 1;
hr2 = gr2 - 1;
hr3 = gr3 - 1;
hr4 = gr4 - 1;
hrnew = hr1;
hrnew = hr2;
hrnew3 = hr3;
hr4new = hr4;
hrnew = Table[{{0, 0}, {0, 0}, {i, 1, ngrid}}];
hrnew[{{All, 1, 1}}] = hr1;
hrnew[{{All, 1, 2}}] = hr2;
hrnew[{{All, 2, 1}}] = hr3;
hrnew[{{All, 2, 2}}] = hr4;
If[IntegerQ[j/savestep], Export[Directory[] <> "/hr_simulation/hrsim" <> " .hne", hrnew]];
hksim = DFFSTForward[hrnew, delr, kvalues, rvalues, 1];
{{gaa, gab}, {gba, gbb}} = solvefzerok[hkprism, delr, 1, end];
{{g2aa, g2ab}, {g2ba, g2bb}} = solvefzeroppk[hkprism, delr, 1, end];
{{h2aa, h2ab}, {h2ba, h2bb}} = solvefzeroppr[hrnew, delr, 1, end];
(* transition points are used that is why the integral is from 1 to tn 
instead of end*)
(*{{h2aa, h2ab}, {hba, hbb}} = solvefzerortail[hrnew, delr, tos 
*tos, tosp];
{{h2taillaa, h2tailab}, {h2tailba, h2tailbb}} = solvefzeropprtail[hrnew, delr, 
tos, tosp];*)
(*{{h2aa, h2ab}, {h2ba, h2bb}} = solvefzerortail[hrnew, delr, 
tos, tosp];*)
(*we keep the rhs1 because this is the 1/k^4 correction term and half 
of the 1/k^2 we expect an additional 1/k^2 divergence that is to be 
controlled by the dielectric h(r) sums*)
rhs1 = -(h2aa - h2ab - hba + hbb);
rhs2 = -1/2(zo*zo*rhoa*rhoa+h2aa +zo*zh*rhoa*rhob*h2ab+zh*zo*rhob*rhoa+ 
h2ba+zh*zh*rhob*rhob+h2ba) -1/2(zo*zo*rhoa*w2aa+zo*zh*rhoa*w2ab+zh*zo 
rhob*w2ba+zh*zh*rhob*w2bb)-(1/epsilonondielectric -1)(1/(4*Pi) 
*betafit));
rhs3 = -2*Sum[hr1[[i]], {i, tsf, tsfp}];
rhs4 = -2*Sum[hr2[[i]], {i, tsf, tsfp}];
rhs5 = -2*Sum[hr3[[i]], {i, tsf, tsfp}];
rhs6 = -2*Sum[hr4[[i]],[i,tfs,tfsp]];
cvector = {rhs1,rhs2,rhs3,rhs4,rhs5,rhs6};
thing = PseudoInverse[qmatrix].cvector;
coefficients = Take[thing,-4];
(* generate tails *)
tailaa = coefficients[[1]]*Table[1/rvalues[[i]]^6,{i,tos,tosp}];
tailba = coefficients[[2]]*Table[1/rvalues[[i]]^6,{i,tos,tosp}];
tailbb = coefficients[[3]]*Table[1/rvalues[[i]]^6,{i,tos,tosp}];
hr1fixed = Join[hr1[[1;;end]],tailaa];
hr1fixed[[begaa;;end]] = hr1fixed[[begaa;;end]] + thing[[1;;lenaa]];
hr2fixed = Join[hr2[[1;;end]],tailab];
hr2fixed[[begab;;end]] = hr2fixed[[begab;;end]] + thing[[1+lenaa;;lenaa+lenab]];
hr3fixed = Join[hr3[[1;;end]],tailba];
hr3fixed[[begab;;end]] = hr3fixed[[begab;;end]] + thing[[1+lenaa+lenab;;lenaa+lenab+lenab]];
hr4fixed = Join[hr4[[1;;end]],tailbb];
hr4fixed[[begab;;end]] = hr4fixed[[begab;;end]] + thing[[1+lenaa+lenab+lenab+lenab+lenab+lenab]];
hrfixed = Table[{{0,0},{0,0}},{i,1,ngrid}];
hrfixed[[All,1,1]] = hr1fixed;
hrfixed[[All,1,2]] = hr2fixed;
hrfixed[[All,2,1]] = hr3fixed;
hrfixed[[All,2,2]] = hr4fixed;
hr = hrfixed;

If[IntegerQ[j/savestep],Export[Directory[]<>"/hr_fixed/hrfixed"<>
    IntegerString[j,10,IntegerLength[n]]<>".tsv",hr]];   
hknew = DFFSTForward[hr,delr,kvalues,rvalues,1];
dhk = Table[hksim[[i]] - hknew[[i]],[i,1,ngrid]];
AppendTo[dhkfistlist,dhk[[1]]];
deltahk = Table[Inverse[omega[[i]]].(hknew[[i]] - hkprism[[i]]),
    {i,1,ngrid}];
deltahk = Table[Inverse[omega[[i]]].(hknew[[{1,2},{2,4}]] -
    hkprism[[{1,2},{2,4}]]),
    {i,1,ngrid}];
If[IntegerQ[j/savestep],Export[Directory[]<>"/diff_hk/diffhk"<>
    IntegerString[j,10,IntegerLength[n]]<>".tsv",dhk]];
If[IntegerQ[j/savestep],Export[Directory[]<>"/delta_ck/deltack"<>
    IntegerString[j,10,IntegerLength[n]]<>".tsv",deltack]];   
hkunmod = DFFSTForward[hrnew,delr,kvalues,rvalues,1];
deltackunmod = Table[Inverse[omega[[i]]].(hkunmod[[i]] -
    hkprism[[i]]),
    {i,1,ngrid}];
ckmix = ckmix + mix*deltack;
If[IntegerQ[j/savestep],Export[Directory[]<>"/ck/cknew"<>
    IntegerString[j,10,IntegerLength[n]]<>".tsv",ckmix]];   
cr = DFFSTBackward[ckmix,delr,kvalues,rvalues,1];}
If [IntegerQ[j/savestep], Export[Directory[] <> "\cr/\cr" <> IntegerString[j, 10, IntegerLength[n]] <> "_\cr.tsv", cr]]; 
err=Total[ςm[Flatten[ςm[i]−clold[i]]^2],{i,1,ngrid}]; 
AppendTo[error, err]; 
{haa, hab}, {hba, hbb} = solvezeroorfull[hrfixed, delr, 1, ngrid]; 
{h2aa, h2ab}, {h2ba, h2bb} = solvezeroopprfull[hrfixed, delr, 1, ngrid]; 
htest = DFFSTForward[hrnew, delr, kvalues, rvalues, 1]; res = {{"Iteration =\cr", "1/k^4\cr-term", "1/k^2\cr-term", "1/k^0\cr-term", "1/k^4+1/k^2+1/k^0"}, {"Overscript[C, \cr]AA\cr\cr(Subscript[k, 1])", a = k4thaa[haa, hab, hbb, hba, h2aa, h2ab, h2ba, h2bb, w2aa, w2ab, w2ba, w2bb, gaa, gab, gba, gbb, g2aa, g2ab, g2ba, g2bb, kvalues[[1]]], b = k2thaa[haa, hab, hbb, hba, h2aa, h2ab, h2ba, h2bb, w2aa, w2ab, w2ba, w2bb, gaa, gab, gba, gbb, g2aa, g2ab, g2ba, g2bb, kvalues[[1]]], c = k0thaa[haa, hab, hbb, hba, h2aa, h2ab, h2ba, h2bb, w2aa, w2ab, w2ba, w2bb, gaa, gab, gba, gbb, g2aa, g2ab, g2ba, g2bb, kvalues[[1]]], a+b+c}, {"Overscript[C, \cr]AB\cr\cr(Subscript[k, 1])", a = k4thab[haa, hab, hbb, hba, h2aa, h2ab, h2ba, h2bb, w2aa, w2ab, w2ba, w2bb, gaa, gab, gba, gbb, g2aa, g2ab, g2ba, g2bb, kvalues[[1]]], b = k2thab[haa, hab, hbb, hba, h2aa, h2ab, h2ba, h2bb, w2aa, w2ab, w2ba, w2bb, gaa, gab, gba, gbb, g2aa, g2ab, g2ba, g2bb, kvalues[[1]]], c = k0thab[haa, hab, hbb, hba, h2aa, h2ab, h2ba, h2bb, w2aa, w2ab, w2ba, w2bb, gaa, gab, gba, gbb, g2aa, g2ab, g2ba, g2bb, kvalues[[1]]], a+b+c}, {"Overscript[C, \cr]BA\cr\cr(Subscript[k, 1])", a = k4thba[haa, hab, hbb, hba, h2aa, h2ab, h2ba, h2bb, w2aa, w2ab, w2ba, w2bb, gaa, gab, gba, gbb, g2aa, g2ab, g2ba, g2bb, kvalues[[1]]], b = k2thba[haa, hab, hbb, hba, h2aa, h2ab, h2ba, h2bb, w2aa, w2ab, w2ba, w2bb, gaa, gab, gba, gbb, g2aa, g2ab, g2ba, g2bb, kvalues[[1]]], c = k0thba[haa, hab, hbb, hba, h2aa, h2ab, h2ba, h2bb, w2aa, w2ab, w2ba, w2bb, gaa, gab, gba, gbb, g2aa, g2ab, g2ba, g2bb, kvalues[[1]]], a+b+c}, {"Overscript[C, \cr]BB\cr\cr(Subscript[k, 1])", a = k4thbb[haa, hab, hbb, hba, h2aa, h2ab, h2ba, h2bb, w2aa, w2ab, w2ba, w2bb, gaa, gab, gba, gbb, g2aa, g2ab, g2ba, g2bb, kvalues[[1]]], b = k2thbb[haa, hab, hbb, hba, h2aa, h2ab, h2ba, h2bb, w2aa, w2ab, w2ba, w2bb, gaa, gab, gba, gbb, g2aa, g2ab, g2ba, g2bb, kvalues[[1]]], c = k0thbb[haa, hab, hbb, hba, h2aa, h2ab, h2ba, h2bb, w2aa, w2ab, w2ba, w2bb, gaa, gab, gba, gbb, g2aa, g2ab, g2ba, g2bb, kvalues[[1]]], a+b+c}}; 
AppendTo[check, res]; Export[Directory[] <> "\textbf{\textit{restart files/\cr.tsv"}, clold]; 
Export[Directory[] <> "\textbf{\textit{restart files/n.tsv"}, j]; 
Export[Directory[] <> "\textbf{\textit{restart files/error.tsv"}, error]; 
hrfinal prism = hrprism; 
hrfinal sim = hrfixed; 
cr final = clold; 
wr final = wr; 
ck final = ckold; 
Export[Directory[] <> "\textbf{\textit{results/hrfinal pr\textit{ism.tsv"}, hrfinal prism]; 
Export[Directory[] <> "\textbf{\textit{results/hrfinal sim.tsv"}, hrfinal sim]; 
Export[Directory[] <> "\textbf{\textit{results/cr\textit{final tsv"}, cr final]; 
Export[Directory[] <> "\textbf{\textit{results/wr\textit{final tsv"}, wr final];
C.4 $k^{-n}$ divergence equations

Below are the detailed equations used to project out the unphysical $k^{-n}$ divergences in the calculation of $\hat{C}_{New}(k)$ in the Two-Chain SC-PRISM hybrid theory/simulation method for the trimer molecules. The Dimer molecule equations are written out explicitly in Chapter 5. For the trimer molecules below, $L$ is the bond length between the two sites and $y$ is used to indicate how far from linear the molecule is. Linear molecules correspond to $y = 0$. Of important note is when comparing the software code presented earlier in the Appendix, the following derivations already have included the analytical values for $\hat{\omega}_{\alpha\gamma}(0)$ and $\hat{\omega}_{\alpha\gamma}^{(2)}(0)$. While they say “full expansions” I have only included the terms from the expansion with $k$ terms in them. The “+ ...” refers to terms that are independent of $k$.

C.4.1 Dimer

The $\delta\hat{C}_{AA}(k)$ full expansion is defined below:

\[
\frac{9(\hat{h}_{AA}(0) - \hat{h}_{AB}(0) - \hat{h}_{BA}(0) + \hat{h}_{BB}(0))}{k^4L^4} + \frac{3}{10k^2L^4}(8L^2\hat{h}_{AA}(0) - 3L^2\hat{h}_{AB}(0) - 3L^2\hat{h}_{BA}(0) - 2L^2\hat{h}_{BB}(0) + 15\hat{h}_{AA}^{(2)}(0) - 15\hat{h}_{AB}^{(2)}(0) + 15\hat{h}_{BB}^{(2)}(0))
\]

+ ...

The $\delta\hat{C}_{AB}(k)$ full expansion is defined below:
\[-\frac{9(\hat{h}_{AA}(0) - \hat{h}_{AB}(0) - \hat{h}_{BA}(0) + \hat{h}_{BB}(0))}{k^4L^4}\]
\[-\frac{3}{10k^2L^4}(3L^2\hat{h}_{AA}(0) - 8L^2\hat{h}_{AB}(0) + 2L^2\hat{h}_{BA}(0) + 3L^2\hat{h}_{BB}(0) + 15\hat{h}_{AA}^{(2)}(0)\]
\[-15\hat{h}_{AB}^{(2)}(0) - 15\hat{h}_{BA}^{(2)}(0) + 15\hat{h}_{BB}^{(2)}(0))\]
+ ...

The $\delta\hat{C}_{BA}(k)$ full expansion is defined below:

\[-\frac{9(\hat{h}_{AA}(0) - \hat{h}_{AB}(0) - \hat{h}_{BA}(0) + \hat{h}_{BB}(0))}{k^4L^4}\]
\[-\frac{3}{10k^2L^4}(3L^2\hat{h}_{AA}(0) - 8L^2\hat{h}_{AB}(0) + 2L^2\hat{h}_{BA}(0) + 3L^2\hat{h}_{BB}(0) + 15\hat{h}_{AA}^{(2)}(0)\]
\[-15\hat{h}_{AB}^{(2)}(0) - 15\hat{h}_{BA}^{(2)}(0) + 15\hat{h}_{BB}^{(2)}(0))\]
+ ...

The $\delta\hat{C}_{BB}(k)$ full expansion is defined below:

\[\frac{9(\hat{h}_{AA}(0) - \hat{h}_{AB}(0) - \hat{h}_{BA}(0) + \hat{h}_{BB}(0))}{k^4L^4}\]
\[+ \frac{3}{10k^2L^4}(-2L^2\hat{h}_{AA}(0) - 3L^2\hat{h}_{AB}(0) - 3L^2\hat{h}_{BA}(0) + 8L^2\hat{h}_{BB}(0) + 15\hat{h}_{AA}^{(2)}(0)\]
\[-15\hat{h}_{AB}^{(2)}(0) - 15\hat{h}_{BA}^{(2)}(0) + 15\hat{h}_{BB}^{(2)}(0)\]
+ ...

**C.4.2 Bent Trimer**

The $\delta\hat{C}_{AA}(k)$ full expansion is defined below:
\[
\frac{144(\hat{h}_{AA}(0) - \hat{h}_{AB}(0) - 2\hat{h}_{BA}(0) + 2\hat{h}_{BB}(0))}{k^4 y^2(y - 4L)^2} \\
- \frac{12}{5k^2 (y^3(y - 4L)^3)}(-32L^4\hat{h}_{AA}(0) + 32L^4\hat{h}_{AB}(0) + 64L^4\hat{h}_{BA}(0) - 64L^4\hat{h}_{BB}(0) + 32L^3\hat{h}_{AA}(0)y \\
- 72L^3\hat{h}_{AB}(0)y - 144L^3\hat{h}_{BA}(0)y + 224L^3\hat{h}_{BB}(0)y + 56L^2\hat{h}_{AA}(0)y^2 + 34L^2\hat{h}_{AB}(0)y^2 \\
+ 68L^2\hat{h}_{BA}(0)y^2 - 248L^2\hat{h}_{BB}(0)y^2 - 32L\hat{h}_{AA}(0)y^3 - 8L\hat{h}_{AB}(0)y^3 - 16L\hat{h}_{BA}(0)y^3 \\
+ 96L\hat{h}_{BB}(0)y^3 + 120L\hat{h}_{AA}^{(2)}(0)y - 120L\hat{h}_{AB}^{(2)}(0)y - 240\hat{h}_{BA}^{(2)}(0)y + 240\hat{h}_{BB}^{(2)}(0)y \\
+ 4\hat{h}_{AA}(0)y^4 + \hat{h}_{AB}(0)y^4 + 2\hat{h}_{BA}(0)y^4 - 12\hat{h}_{BB}(0)y^4 - 30\hat{h}_{AA}^{(2)}(0)y^2 + 30\hat{h}_{AB}^{(2)}(0)y^2 \\
+ 60\hat{h}_{BA}^{(2)}(0)y^2 - 60\hat{h}_{BB}^{(2)}(0)y^2) \\
+ ...
\]

The \( \delta \hat{C}_{AB}(k) \) full expansion is defined below:

\[
\frac{144(\hat{h}_{AA}(0) - \hat{h}_{AB}(0) - 2\hat{h}_{BA}(0) + 2\hat{h}_{BB}(0))}{k^4 (y^2(y - 4L)^2)} \\
- \frac{12}{5k^2 (y^3(y - 4L)^3)}(32L^4\hat{h}_{AA}(0) - 32L^4\hat{h}_{AB}(0) - 64L^4\hat{h}_{BA}(0) + 64L^4\hat{h}_{BB}(0) - 72L^3\hat{h}_{AA}(0)y \\
+ 112L^3\hat{h}_{AB}(0)y + 224L^3\hat{h}_{BA}(0)y - 304L^3\hat{h}_{BB}(0)y + 34L^2\hat{h}_{AA}(0)y^2 - 44L^2\hat{h}_{AB}(0)y^2 \\
- 248L^2\hat{h}_{BA}(0)y^2 + 268L^2\hat{h}_{BB}(0)y^2 - 8L\hat{h}_{AA}(0)y^3 + 8L\hat{h}_{AB}(0)y^3 + 96L\hat{h}_{BA}(0)y^3 \\
- 96L\hat{h}_{BB}(0)y^3 + 120L\hat{h}_{AA}^{(2)}(0)y + 120L\hat{h}_{AB}^{(2)}(0)y + 240L\hat{h}_{BA}^{(2)}(0)y - 240L\hat{h}_{BB}^{(2)}(0)y \\
+ \hat{h}_{AA}(0)y^4 - \hat{h}_{AB}(0)y^4 - 12\hat{h}_{BA}(0)y^4 + 12\hat{h}_{BB}(0)y^4 + 30\hat{h}_{AA}^{(2)}(0)y^2 - 30\hat{h}_{AB}^{(2)}(0)y^2 \\
- 60\hat{h}_{BA}^{(2)}(0)y^2 + 60\hat{h}_{BB}^{(2)}(0)y^2) \\
+ ...
\]

The \( \delta \hat{C}_{BA}(k) \) full expansion is defined below:
\[
\frac{72(\hat{h}_{AA}(0) - \hat{h}_{AB}(0) - 2\hat{h}_{BA}(0) + 2\hat{h}_{BB}(0))}{k^4(y^2(y - 4L)^2)}
\] 
\[
- \frac{6}{5k^2(y^3(y - 4L)^3)}(32L^4\hat{h}_{AA}(0) - 32L^4\hat{h}_{AB}(0) - 64L^4\hat{h}_{BA}(0) + 64L^4\hat{h}_{BB}(0) - 72L^3\hat{h}_{AA}(0)y
\]
\[
+ 112L^3\hat{h}_{AB}(0)y + 224L^3\hat{h}_{BA}(0)y - 304L^3\hat{h}_{BB}(0)y + 34L^2\hat{h}_{AA}(0)y^2 - 124L^2\hat{h}_{AB}(0)y^2
\]
\[
- 88L^2\hat{h}_{BA}(0)y^2 + 268L^2\hat{h}_{BB}(0)y^2 - 8L\hat{h}_{AA}(0)y^3 + 48L\hat{h}_{AB}(0)y^3 + 16L\hat{h}_{BA}(0)y^3
\]
\[
- 96L\hat{h}_{BB}(0)y^3 - 120\hat{h}_{AA}(0)y^4 + 120\hat{h}_{AB}(0)y^4 + 240L\hat{h}_{BA}(0)y^4 - 240L\hat{h}_{BB}(0)y^4
\]
\[
+ \hat{h}_{AA}(0)y^4 - 6\hat{h}_{AB}(0)y^4 - 2\hat{h}_{BA}(0)y^4 + 12\hat{h}_{BB}(0)y^4 + 30\hat{h}_{AA}(0)y^2 - 30\hat{h}_{AB}(0)y^2
\]
\[
- 60\hat{h}_{BA}(0)y^2 + 60\hat{h}_{BB}(0)y^2)
\]

+ ...

The \(\delta \hat{C}_{BB}(k)\) full expansion is defined below:

\[
\frac{72(\hat{h}_{AA}(0) - \hat{h}_{AB}(0) - 2\hat{h}_{BA}(0) + 2\hat{h}_{BB}(0))}{k^4(y^2(y - 4L)^2)}
\] 
\[
+ \frac{12}{5k^2y^2(y - 4L)^2}(16L^4\hat{h}_{AA}(0) - 16L^4\hat{h}_{AB}(0) - 32L^4\hat{h}_{BA}(0) + 32L^4\hat{h}_{BB}(0) - 56L^3\hat{h}_{AA}(0)y
\]
\[
+ 76L^3\hat{h}_{AB}(0)y + 152L^3\hat{h}_{BA}(0)y - 192L^3\hat{h}_{BB}(0)y + 62L^2\hat{h}_{AA}(0)y^2 - 67L^2\hat{h}_{AB}(0)y^2
\]
\[
- 134L^2\hat{h}_{BA}(0)y^2 + 144L^2\hat{h}_{BB}(0)y^2 - 24L\hat{h}_{AA}(0)y^3 + 24L\hat{h}_{AB}(0)y^3 + 48L\hat{h}_{BA}(0)y^3
\]
\[
- 48L\hat{h}_{BB}(0)y^3 - 60\hat{h}_{AA}(0)y^4 + 60\hat{h}_{AB}(0)y^4 + 120\hat{h}_{BA}(0)y^4 - 120L\hat{h}_{BB}(0)y^4
\]
\[
+ 3\hat{h}_{AA}(0)y^4 - 3\hat{h}_{AB}(0)y^4 - 6\hat{h}_{BA}(0)y^4 + 6\hat{h}_{BB}(0)y^4 + 15\hat{h}_{AA}(0)y^2
\]
\[
- 15\hat{h}_{AB}(0)y^2 - 30\hat{h}_{BA}(0)y^2 + 30\hat{h}_{BB}(0)y^2)
\]

+ ...

C.4.3  Linear Trimer

The \(\delta \hat{C}_{AA}(k)\) full expansion is defined below:
\[
\frac{2025(\hat{h}_{AA}(0) - \hat{h}_{AB}(0) - \hat{h}_{BA}(0) + \hat{h}_{BB}(0))}{k^8 L^8}
\]
\[
- \frac{675}{14k^6 L^8} (16L^2 \hat{h}_{AA}(0) - 9L^2 \hat{h}_{AB}(0) - 9L^2 \hat{h}_{BA}(0)
\]
\[
+ 2L^2 \hat{h}_{BB}(0) - 21\hat{h}_{AA}^{(2)}(0) + 21\hat{h}_{AB}^{(2)}(0) + 21\hat{h}_{BA}^{(2)}(0) - 21\hat{h}_{BB}^{(2)}(0))
\]
\[
+ \frac{45}{392k^4 L^8} (735\hat{h}_{AA}^{(4)}(0) + 1696L^4 \hat{h}_{AA}(0) - 527L^4 \hat{h}_{AB}(0) - 527L^4 \hat{h}_{BA}(0) - 152L^4 \hat{h}_{BB}(0)
\]
\[
- 3360L^2 \hat{h}_{AA}^{(2)}(0) + 1890L^2 \hat{h}_{AB}^{(2)}(0) + 1890L^2 \hat{h}_{BA}^{(2)}(0) - 420L^2 \hat{h}_{BB}^{(2)}(0) - 735\hat{h}_{AB}^{(4)}(0) - 735\hat{h}_{BA}^{(4)}(0)
\]
\[
+ 735\hat{h}_{BB}^{(4)}(0))
\]
\[
- \frac{45}{5488k^2 L^8} (3920\hat{h}_{AA}^{(4)}(0)L^2 + 2816L^6 \hat{h}_{AA}(0) + 40L^6 \hat{h}_{AB}(0) + 40L^6 \hat{h}_{BA}(0) - 54L^6 \hat{h}_{BB}(0)
\]
\[
- 11872L^4 \hat{h}_{AA}^{(2)}(0) + 3689L^4 \hat{h}_{AB}^{(2)}(0) + 3689L^4 \hat{h}_{BA}^{(2)}(0) + 1064L^4 \hat{h}_{BB}^{(2)}(0) - 2205L^2 \hat{h}_{AB}^{(4)}(0)
\]
\[
- 2205L^2 \hat{h}_{BA}^{(4)}(0) + 490L^2 \hat{h}_{BB}^{(4)}(0) - 343\hat{h}_{AA}^{(6)}(0) + 343\hat{h}_{AB}^{(6)}(0) + 343\hat{h}_{BA}^{(6)}(0) - 343\hat{h}_{BB}^{(6)}(0))
\]
\[
+ \ldots
\]

The $\delta \hat{C}_{AB}(k)$ full expansion is defined below:

\[
- \frac{2025(\hat{h}_{AA}(0) - \hat{h}_{AB}(0) - \hat{h}_{BA}(0) + \hat{h}_{BB}(0))}{2k^8 L^8}
\]
\[
+ \frac{675}{28k^6 L^8} (9L^2 \hat{h}_{AA}(0) - 2L^2 \hat{h}_{AB}(0) - 2L^2 \hat{h}_{BA}(0)
\]
\[
- 5L^2 \hat{h}_{BB}(0) - 21\hat{h}_{AA}^{(2)}(0) + 21\hat{h}_{AB}^{(2)}(0) + 21\hat{h}_{BA}^{(2)}(0) - 21\hat{h}_{BB}^{(2)}(0))
\]
\[
- \frac{45}{784k^4 L^8} (527L^4 \hat{h}_{AA}(0) - 240L^4 \hat{h}_{AB}(0) + 152L^4 \hat{h}_{BA}(0) + 51L^4 \hat{h}_{BB}(0) - 1890L^2 \hat{h}_{AA}^{(2)}(0)
\]
\[
+ 420L^2 \hat{h}_{AB}^{(2)}(0) + 420L^2 \hat{h}_{BA}^{(2)}(0) + 1050L^2 \hat{h}_{BB}^{(2)}(0) + 735\hat{h}_{AA}^{(4)}(0) - 735\hat{h}_{AB}^{(4)}(0) - 735\hat{h}_{BA}^{(4)}(0)
\]
\[
+ 735\hat{h}_{BB}^{(4)}(0))
\]
\[
- \frac{45}{10976k^2 L^8} (40L^6 \hat{h}_{AA}(0) - 34L^6 \hat{h}_{AB}(0) - 54L^6 \hat{h}_{BA}(0) - 226L^6 \hat{h}_{BB}(0) + 3689L^4 \hat{h}_{AA}^{(2)}(0)
\]
\[
- 1680L^4 \hat{h}_{AB}^{(2)}(0) + 1064L^4 \hat{h}_{BA}^{(2)}(0) + 357L^4 \hat{h}_{BB}^{(2)}(0) - 2205L^2 \hat{h}_{AA}^{(4)}(0) + 490L^2 \hat{h}_{AB}^{(4)}(0)
\]
\[
+ 490L^2 \hat{h}_{BA}^{(4)}(0) + 1225L^2 \hat{h}_{BB}^{(4)}(0) + 343\hat{h}_{AA}^{(6)}(0) - 343\hat{h}_{AB}^{(6)}(0) - 343\hat{h}_{BA}^{(6)}(0) + 343\hat{h}_{BB}^{(6)}(0)
\]
\[
+ \ldots
\]

212
The \( \delta \hat{C}_{BA}(k) \) full expansion is defined below:

\[
- \frac{2025(\hat{h}_{AA}(0) - \hat{h}_{AB}(0) - \hat{h}_{BA}(0) + \hat{h}_{BB}(0))}{2k^8L^8} \\
+ \frac{675}{28k^6L^8}(9L^2\hat{h}_{AA}(0) - 2L^2\hat{h}_{AB}(0) - 2L^2\hat{h}_{BA}(0) \\
- 5L^2\hat{h}_{BB}(0) - 21\hat{h}_{AA}^{(2)}(0) + 21\hat{h}_{AB}^{(2)}(0) + 21\hat{h}_{BA}^{(2)}(0) - 21\hat{h}_{BB}^{(2)}(0)) \\
- \frac{45}{784k^4L^8}(527L^4\hat{h}_{AA}(0) + 152L^4\hat{h}_{AB}(0) - 240L^4\hat{h}_{BA}(0) + 51L^4\hat{h}_{BB}(0) - 1890L^2\hat{h}_{AA}^{(2)}(0) \\
+ 420L^2\hat{h}_{AB}^{(2)}(0) + 420L^2\hat{h}_{BA}^{(2)}(0) + 1050L^2\hat{h}_{BB}^{(2)}(0) + 735\hat{h}_{AA}^{(4)}(0) - 735\hat{h}_{AB}^{(4)}(0) - 735\hat{h}_{BA}^{(4)}(0) \\
+ 735\hat{h}_{BB}^{(4)}(0)) \\
\frac{45}{10976k^2L^8}(40L^6\hat{h}_{AA}(0) - 54L^6\hat{h}_{AB}(0) - 544L^6\hat{h}_{BA}(0) - 226L^6\hat{h}_{BB}(0) + 3689L^4\hat{h}_{AA}^{(2)}(0) \\
+ 1064L^4\hat{h}_{AB}^{(2)}(0) - 1680L^4\hat{h}_{BA}^{(2)}(0) + 357L^4\hat{h}_{BB}^{(2)}(0) - 2205L^2\hat{h}_{AA}^{(4)}(0) + 490L^2\hat{h}_{AB}^{(4)}(0) \\
+ 490L^2\hat{h}_{BA}^{(4)}(0) + 1225L^2\hat{h}_{BB}^{(4)}(0) + 343\hat{h}_{AA}^{(6)}(0) - 343\hat{h}_{AB}^{(6)}(0) - 343\hat{h}_{BA}^{(6)}(0) + 343\hat{h}_{BB}^{(6)}(0)) \\
+ ... 
\]

The \( \delta \hat{C}_{BB}(k) \) full expansion is defined below:
\[
\frac{2025(\hat{h}_{AA}(0) - \hat{h}_{AB}(0) - \hat{h}_{BA}(0) + \hat{h}_{BB}(0))}{4k^8L^8}
\]
\[
- \frac{675}{56k^6L^8}(2L^2\hat{h}_{AA}(0) + 5L^2\hat{h}_{AB}(0) + 5L^2\hat{h}_{BA}(0)
- 12L^2\hat{h}_{BB}(0) - 21\hat{h}_{AA}^{(2)}(0) + 21\hat{h}_{AB}^{(2)}(0) + 21\hat{h}_{BA}^{(2)}(0) - 21\hat{h}_{BB}^{(2)}(0))
\]
\[
- \frac{45}{1568k^4L^8}(152L^4\hat{h}_{AA}(0) + 51L^4\hat{h}_{AB}(0) + 51L^4\hat{h}_{BA}(0) - 744L^4\hat{h}_{BB}(0) + 420L^2\hat{h}_{AA}^{(2)}(0)
+ 1050L^2\hat{h}_{AB}^{(2)}(0) + 1050L^2\hat{h}_{BA}^{(2)}(0) - 735\hat{h}_{AA}^{(4)}(0) + 735\hat{h}_{AB}^{(4)}(0) + 735\hat{h}_{BA}^{(4)}(0)
- 735\hat{h}_{BB}^{(4)}(0))
\]
\[
+ \frac{45}{21952k^2L^8}(54L^6\hat{h}_{AA}(0) + 226L^6\hat{h}_{AB}(0) + 226L^6\hat{h}_{BA}(0) + 768L^6\hat{h}_{BB}(0) - 1064L^4\hat{h}_{AA}^{(2)}(0)
- 357L^4\hat{h}_{AB}^{(2)}(0) - 357L^4\hat{h}_{BA}^{(2)}(0) + 5208L^4\hat{h}_{BB}^{(2)}(0) - 490L^2\hat{h}_{AA}^{(4)}(0) - 1225L^2\hat{h}_{AB}^{(4)}(0) - 1225L^2\hat{h}_{BA}^{(4)}(0)
+ 2940L^2\hat{h}_{BB}^{(4)}(0) + 343\hat{h}_{AA}^{(6)}(0) - 343\hat{h}_{AB}^{(6)}(0) - 343\hat{h}_{BA}^{(6)}(0) + 343\hat{h}_{BB}^{(6)}(0))
\]
\[+ ...\]