Structure Of Polarisable Ionic Fluids

Nick Burch
Magdalen College
Oxford University

A thesis submitted for the
Honour School of Natural Science.
(Chemistry Part II)
June 2003

The work in this thesis is entirely my own, except where I have *either* acknowledged help from a named person *or* given a reference to a published source or a thesis. Text taken from another source will be enclosed in quotation marks and a reference given.

Date **Signature**

Summary

In this thesis the structure and dynamical properties of two ionic fluids are studied by computer simulation. This proceeded by the calculation of radial distribution functions, structure factors, neutron scattering patterns, diffusion coefficients and viscosities. This data are then compared to experimental and theoretical results where appropriate.

Before looking at the properties of these systems, the run length and system size effects are investigated. For the run length study, we look at how averaging may be used to improve the quality of results. As part of this, we investigate the appropriate run lengths and the suitable degree of averaging required to calculate the properties of interest. The problem of stable values against long time values is highlighted and discussed. We introduce a new method for using computing clusters for Molecular Dynamics simulations, which is well suited to generating data for averaging and scanning a large number of state points. For the system size investigation, we study how representative our results are of large systems, by testing a range of system sizes around our chosen values.

A theoretical model for the structure of polarisable ionic fluids is tested against Molecular Dynamics simulations, and its predictions are verified. This model is then investigated further, with predictions made on its dynamic properties, and two candidates for a possible reduced parameter for dynamics are investigated. The effect of polarisation on the properties of the system is examined, with the aim of discovering under what circumstances polarisation plays a large role. The effects of polarisation and temperature on the cavitation of the system are also investigated.

Finally, the dynamic and structural properties of a complex system, $ScCl_3$, are investigated. $ScCl_3$ is chosen as it is one of the tricker MX_3 systems to simulate, owing to its long range chains. Simulation of MX_3 systems is important in helping to understand electrodeposition of refactory metals and the electrolytic separation of nuclear waste, which typically occur with physical conditions too intense for experimental study. The dynamic and structural properties of $ScCl_3$ are investigated.

Contents

List of Figures

Acknowledgements

First and foremost I would like to thank my supervisor, Prof. Paul Madden, for his help in getting me started at the beginning of the year, and for his insightful input throughout the rest of the time. Secondly, I'd like to thank Gus Gray-Weale for his help, advice and collaboration throughout the year.

Thanks also to the other members of the group, for making it a year to remember. Ollie and Ben, without your patient replies to "um, how do I do this?", I'd never have made it. Steve and Will, thanks for reminding me of all the things I should've remembered but hadn't, and for convincing me it was fine to skip work and go to the pub..... Sandro, thanks for keeping me on my toes by coming up with all sorts of computing problems that I couldn't solve! Overall, thanks everyone in the group, it was a pleasure to work with you.

To my friends - thanks for being there all, and for frequently dragging me to the pub / party / competition / event etc, and generally helping me have a good time this year.

To the Oxford Supercomputer Centre - thanks for giving me an excessive quantity of computing time to use, without which this project wouldn't have been possible.

I would like to thank my parents, for their support both this year and for the rest of the degree, and being there when I needed their help.

Finally, I'd like to thank my girlfriend, Hilary.

Oh, and to everyone who proof read this thesis - thank you, and I hope it wasn't *too* painful a process...

Glossary

MD	Molecular Dynamics. A computer simulation of a system of particles in which the system's forces are calculated from the potentials between the particles, and then Newton's laws of motion are used to calculate their trajectories.
RIM	Rigid Ion Model. This is a simple ionic model, where the ions are treated as electronically rigid closed shell species, with their formal valence charges.
PIM	Polarisable Ion Model. An extension of the RIM, where the ions are no longer treated as electronically rigid species, but have induced multipoles.
SMSM	Simple Molten Salt Model. A simple theoretical, rigid-ion model for ionic fluids.
PSMSM	Polarisable Simple Molten Salt Model. An extention to the SMSM, which allows for the anions to be polarisable.
RPM	Restrictive Primitive Model. An equimolar mixture of oppositely charged hard spheres. Generally similar to the SMS.
HNCM	Hypernetted Chain Model. A theoretical approximation for fluids, which is not thermodynamically consistent.
RDF	Radial Distribution Function. A measure of how dense the other particles around any given particle are, relative to the ideal gas.
MSD	Mean Squared Displacement. A function which gives a measure of how far on average the particles have moved in a given time.
k-vectors	When performing calculations in reciprocal space, k-vectors are points in that reciprocal space.

Chapter 1

Introduction

In this thesis, we use Molecular Dynamics simulations to test a theoretical model [1] for the structure of polarisable ionic fluids. In order to easily test the theory at a wide range of state points, we make use of a new method of running programs on a computing cluster. Having verified the theory, we go on to investigate the dynamic properties of a system it works with, using the theory in a predictive role.

Finally, we apply these Molecular Dynamics techniques to a system of practical interest, molten $ScCl_3$. MX_3 systems are of technological importance due to their use in electrodeposition of refactory metals [3] and electolytic separation of nuclear waste [4]. $ScCl_3$ was selected as it displays many of the more tricky "covalent" characteristics, which require a polarisable ionic system to be modelled correctly [2].

1.1 Molecular Dynamics Simulation

Molecular Dynamics is the technique of simulating a system by solving the classical equations of motion for the particles. Using the Born-Oppenheimer approximation, it is possible to describe the system in terms of the positions and momenta of its atoms and ions. MD simulations work by calculating the force on all the ions, and then solving Newton's laws of motion, lead to predictions of the velocities and positions of the particles at some later time. These are calculated for one *timestep* later. By repeating this process for a succession of timesteps, we produce the trajectory of the system for the timescale of interest. We can use the trajectory to calculate many measurable properties of the system, as well as to examine the microscopic processess involved in these properties.

The length of the timestep depends on the system under investigation. We must ensure that it is not too long, otherwise the system won't display the correct classical trajectories. Using a timestep that is too short will result in unnecessary calculations being performed. The timestep for any system must be tuned in advance of the data collection simulations. For the $ScCl_3$ systems, a timestep of 15 au was used (1 au = 2.418 x 10^{-17}s). The theoretical system makes use of scaled units (see section 1.5) to maximise its applicability. Its timestep was 0.005 scaled time units. Both of these timescales provided smooth, well behaved trajectories in the regions of interest, where the equations of motion could be properly integrated.

The number of timesteps to run for (run length) is affected by the available computing resources, and the properties under investigation. Details of the criteria for the selection of the run lengths can be found in chapter 2.

In order to correctly simulate a chemical system by MD, one must have a realistic interaction potential to describe the forces between the particles. For this thesis, this was relatively straightforward. For the theoretical model, the potential was already specified. For $ScCl_3$, previous work done in the group had produced a potential which was quite accurate.

When looking at the properties of a bulk liquid (as we are), it is important to remove any surface effects that might be introduced by the simulation. This is achieved by using *periodic boundary conditions* [5], where the simulation cell is repeated through space, creating an infinite lattice. Whenever a particle moves out of the simulation cell, it is replaced by one of its periodic images entering the simulation cell from the opposite face. This can be seen in figure 1.1.

Each ion then has an infinite number of images of itself. However, when calculating short range interaction terms, we only need to consider the ion once. As such, the short range interaction between two ions i and j is calculated between i and the

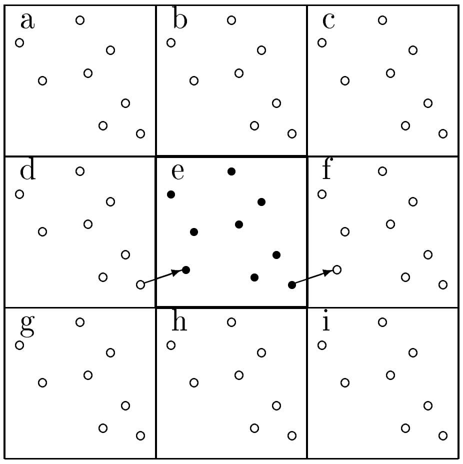

Figure 1.1: The application of Periodic Boundary Conditions.

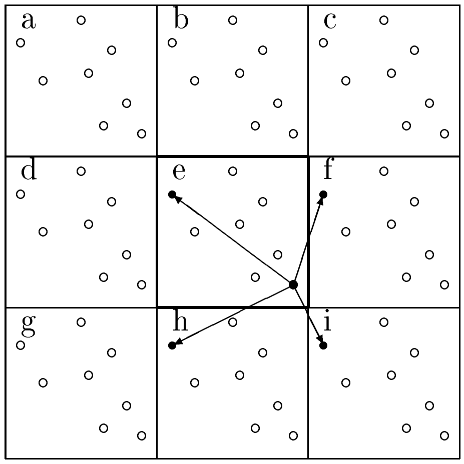

Figure 1.2: The calculation of the positions of the Nearest Neighbours

nearest periodic image of j. This is known as the *mimimum image convention* [6], and can be seen in figure 1.2.

For longer range interaction terms, such as the Coulombic charge-charge interaction (given by $\frac{Q_iQ_j}{r_{ij}}$), the terms don't become negligibly small on the scale of the simulation cell. We need to include the periodic images of the ions. This procedure is handled in reciprocal space by the use of *Ewald Summations* [7].

1.2 The Rigid Interaction Model

In a simple ionic model, with ions treated as rigid spheres unaffected by their surroundings, the system is best represented by a pair potential of Born-Mayer form:

$$U_{ij} = \frac{Q_iQ_j}{r_{ij}} + B_{ij}e^{-\alpha_{ij}r_{ij}} - \sum_{n=6,8} \frac{C_{ij}^n}{r_{ij}^n} f_{ij}^n \tag{1.1}$$

This model is also known as the Rigid Ion Model (RIM). The $\frac{Q_iQ_j}{r_{ij}}$ term describes the coulombic interactions between two ions, i and j, with formal charges Q_i and Q_j at a separation of r_{ij}. The $B_{ij}e^{-\alpha_{ij}r_{ij}}$ term is the short range repulsion term due to the Pauli Exclusion Principle, which prevents overlap of the closed shell electron wavefunctions. Finally, the $\sum_{n=6,8} \frac{C_{ij}^n}{r_{ij}^n} f_{ij}^n$ term is the dispersion energy. This is caused by attractive perturbation effects of the electron distribution on one ion on the distribution on a neighbouring ion, such as van der Waals forces.

All compressibility and polarisability effects are ignored by this model. It doesn't allow non-spherical electron distributions, as caused by such things as being in an asymmetrical coordination site. Because of this, we will normally use a model which allows non-spherical distributions via the effect of polarisation, the PIM.

1.3 Polarisable Ion Model (PIM)

This model is based on RIM, but adds polarisability to the ions [8], most commonly (but not always) the anions. These dipoles interact both with themselves, and with the ionic charges present.

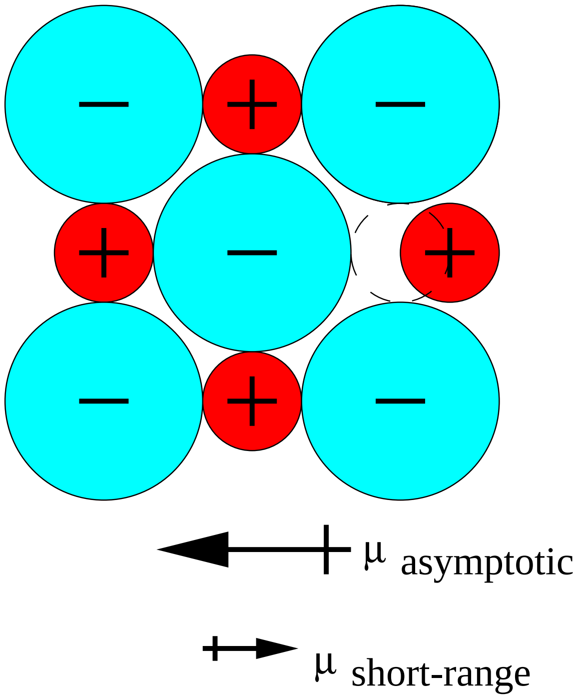

Figure 1.3: Origin of the "short range" and "asymptotic" contributions to the dipole that is induced by cation motion in a condensed phase.

In the case where the ions and the dipoles are far enough away that their electrons don't overlap, then the resultant dipole is due only to coulombic effects. It is given by equation 1.2:

$$\vec{\mu}^i = \alpha_i \left(\sum_j \frac{Q_j \vec{r}_{ij}}{r_{ij}^3} - \sum_j \left[\frac{\vec{\mu}^j}{r_{ij}^3} - \frac{3\vec{r}_{ij}\left(\vec{r}_{ij}\cdot\vec{\mu}^j\right)}{r_{ij}^5} \right] \right) \tag{1.2}$$

In the condensed phase, the dipole induction is modified by the overlap of the electron clouds. We can see this by considering an anion, surrounded by cations, one of which is off its lattice site. This is depicted in figure 1.3. As one cation moves away from its lattice site, then it will induce a dipole on the polarisable anion (the asymptotic dipole). This induced dipole will be such that the negative end points towards the cation that is *opposite* the one which moved away. The overall dipole will be reduced in magnitude by a second dipole (the short-range dipole), in the opposite direction. This is caused by electron cloud overlap - the anion's electrons will move into the gap left by the cation which moved. Thus, the asymptotic (coulombic) component is opposed by the short range interaction. The short range component is given by

$$\vec{\mu}^i_{\text{short-range}} = \sum_{j\neq i} \alpha_i f\left(r_{ij}\right) \frac{Q\vec{r}_{ij}}{r_{ij}^3} \tag{1.3}$$

where the f_{ij} term is the short range damping parameter. Electronic structure calculations have shown that this is best described by a modified Tang-Toennies function [9]:

$$f_{ij}^n = -\mathrm{e}^{-br_{ij}} \sum_{k=0}^{n} \frac{\left(br_{ij}\right)^k}{k!} \tag{1.4}$$

This tends to -1 at $r = 0$, and to 0 at large r. The b term is the short range induction damping parameter, which scales with the sum of the ionic radii:

$$b = \frac{c}{\sigma_i + \sigma_j} \tag{1.5}$$

The c term is a constant for the given stoichiometry. For anions, it can reduce the resultant dipole by up to 50%, an effect known as induction damping.

Previous studies have shown how polarisation affects the structure of solid and liquid ionic systems [10, 11]. Some investigation has been done in how polarisation affects the dynamics in real MX systems [12], real MX_3 systems [13], the liquid-vapour interface [14], and a few other systems.

1.4 Theoretical Models for Ionic Systems

With MD simulations, we use an ionic model which describes the individual ions, and then calculate forces and trajectories for each ion. With most theory calculations, we produce an equation for the property of interest, based on an ionic model, and solve this directly. This makes MD simulations more computationally expensive to perform.

Theories of fluid structure are faster to compute results from, and can quickly provide a fluid's RDF for many densities and temperatures. They allow you to trace behaviours over a wide range of thermodynamic states, and for some properties (eg the low k limit of the structure factor) they can give a result with much less computation that MD can. However, in order to make them solvable, it is necessary to introduce simplifications. These lead to *systematic errors*, particularly thermodynamic inconsistencies. As such, theories complement MD, and neither can really replace the other.

At low density (just above the ideal gas region), any fluid only has weak interactions. In this region, we can express g(r) (the RDF) as $g(r) = \exp(-\frac{u(r)}{kT})$.

At higher density, multi body effects come into play. The presence of a third body will restrict the available conformations for a pair of ions. As such, we need to alter the g(r) expression to include many body effects. This is normally done via an extra term:

$$g(r) = \exp\left(-\frac{u(r)}{kT} + W(r)\right) \tag{1.6}$$

The trick is then in picking a function for W(r). Many choices for W(r) have been explored, but a discussion of these is beyond the scope of the thesis. For this thesis, I have been involved in testing a theory whose $W(r)$ is based on the Hypernetted

Chain Approximation (HNC). The theory has been developed for polarisable ionic fluids by Dr A. Gray-Weale in this group. A full account of this theory (which is beyond the scope of the thesis) is given in reference [1].

The use of equation 1.6 normally requires a pair potential, as is the case where the dipoles behave as classical oscillators in thermal equilibrium with the ionic coordinates, or for the rigid ion model (where polarisation effects are ignored). However, the situation we'd really like is to have the system in its ground electronic state, as this is more physically correct, since $kT \gg \hbar\omega$. This can be represented by minimising the energy across the whole system, solving $\frac{\partial V}{\partial \mu_i} = 0$, for the dipoles with the ions fixed. In this situation, our system is no longer governed by pair potentials. In reference [1], a way to calculate the RDF for a system with $\frac{\partial V}{\partial \mu_i} = 0$ and with the usual fluid structure theory (HNC) is proposed. In section 4.1, we will test how well this theory works.

Theoretical models deliver a "broad brush" approach. As such, it is normal to apply them to a simple ionic model. A common approach is to pick the Simple Molten Salt Model (SMSM) [15]. With a SMS, there are a number of simplifications to the ionic model. Firstly, the short range attractive part is given by $U_{SR}(r) = \frac{1}{9}(\frac{\sigma}{r})^9$. The system has ions of equal and opposite charge (MX stoichiometry). For the theory we're examining, the model has been extended, and the anions are polarisable.

1.5 Corresponding States

When using theoretical models, it is normal to try to make them as general as possible. One common way to aid in this goal is through the use of corresponding states [16]. With corresponding states, we try to come up with a minimal set of parameters to specify the system. We then apply relationships to generate all the other parameters of the system from these. Thus, when moving from one system to another, we only need to know a few specific things to be able to come up with all the parameters our simulation or theory requires. In essence, we combine physical quantities together into a smaller number of dimensionless parameters, and use only these to represent the differences between systems.

One example of corresponding states is with Lenard-Jones systems. Here we can use the scaled temperature $T^* = \frac{kT}{\epsilon}$ (where ϵ is the depth of the potential minimum) in calculations, to encapsulate the system differences. A more common use of corresponding states is within the Restrictive Primitive Model (RPM), a simple model for ionic systems. Here, instead of $u = \frac{\beta Q^2}{\epsilon r}$, we use a general form of $u = \frac{\beta^*}{r}$ (where $\beta^* = \frac{Q^2 \beta}{\epsilon}$, ϵ is now the dielectric constant, and β is the reciprocal thermal energy $\beta = \frac{1}{kT}$). We now only need the one parameter, where previously we required three. As an aside, we also notice the use of ϵ in these equations. One of the objectives of this thesis is to see to what extent the effects of polarisation on the properties of the simulated fluid could be treated as a dielectric screening effect in the manner of the RPM. More on this can be found starting in section 4.3.

By writing our theory to take advantage of such corresponding states, we can make it far more general. As seen above, we define a scaling (or reduced) parameter in terms of our physical parameters, and then use only this. By altering our scaling parameter, we can move to the system of interest. Such changes, for example, allow the RPM to model both molten ($T^* = 0.05$ at critical point) and aqueous NaCl ($T^* = 0.15$ at 298k), simply by altering β^* or T^* accordingly.

The scaling parameters for use with the RPM on MX systems are well known [16], and have been shown to apply with only limited changes to polarisable systems [1]. In chapter 4, we will investigate a number of potential scaling parameters for describing the dynamic properties of the system.

1.6 Properties of Interest

In this section, we will look at the theoretical basis for the properties of interest in our ionic fluids. These are Pressure, Diffusion, Viscosity, Structure Factors, and Neutron Diffraction Patterns.

1.6.1 Pressure

The pressure of the system can be determined by a number of different methods, depending on which statistical mechanical properties are closest to hand during the simulation.

For the MD code used with this thesis, the pressure was calculated from the virial pressure expression, which for this material has three components. The first component is the "k_BT" term, equal to $\rho k_B T$. This is the ideal gas pressure. The second component comes from the short range potential. The final component is the coulombic pressure. A special property of coulombic fluids is that the coulombic term of their pressure is directly proportional to the coulombic energy.

Full details of calculating pressures for MD systems can be found in reference [17].

1.6.2 Structure Factors

To get the structure factors for a MD simulation, there are two main ways. We will see one of them in section 1.6.3, when we calculate Neutron Scattering Patterns. For calculations of the partial structure factors, the second method was employed. In this, we directly average over the correlation functions of the Fourier components of the ion densities:

$$S_{\alpha\beta}(k) = (N_\alpha N_\beta)^{-\frac{1}{2}} \left\langle \sum_{i\in\alpha} \sum_{j\in\beta} \exp[i\vec{k}.\vec{r^{ij}}] \right\rangle \tag{1.7}$$

Where α and β are the ion types, e.g. $S_{anion,anion}$.

In effect, this looks at all the possible k vectors in turn, and calculates their contribution. As we get to larger k values, the number of vectors for that k distance increases rapidly. As such, it doesn't scale very well out to large distances. Also, the smallest k value we can see if restricted by the periodic boundary conditions to be $\frac{2\pi}{L}$. Due to these restrictions, the structure factors haven't been calculated at out as far as those from the theory, nor at quite such low k values. It is at these low k values that the special properties of coulombic fluids manifest themselves, so this method is may be of use for investigating those properties in larger box sizes.

We use this method to calculate the three partial structure factors, anion-anion, anion-cation and cation-cation. From these, it is possible to calculate other structure factors. For MX systems, the number-number structure factor is given by

$$S_{nn} = \frac{S_{cc} + S_{aa} + 2S_{ca}}{2} \tag{1.8}$$

where cc is cation-cation, ca is cation-anion etc. Meanwhile, the charge-charge structure factor for MX systems is found from

$$S_{zz} = \frac{S_{cc} + S_{aa} - 2S_{ca}}{2} \tag{1.9}$$

1.6.3 Neutron Scattering Patterns

The main way to calculate the neutron scattering patterns for a MD simulation is to work from the structure factors, and weight them according to the neutron scattering length for each ion (found experimentally).

One way to calculate the structure factors was given in section 1.6.2. As was noted there, it is computationally expensive to use this method at large k values. A second method to calculate the structure factors is simply to calculate the Fourier transformations of the RDFs, computing:

$$S_{\alpha\beta}(k) = \delta_{\alpha\beta} + 4\pi n_0 \sqrt{c_\alpha c_\beta} \int dr (g_{\alpha\beta}(r) - 1) r \frac{\sin(kr)}{k} \tag{1.10}$$

Where c_α is the concentration of species α. This method is much faster, provided the RDFs have been computed (which they have in this case). However, this method can suffer truncation errors as k becomes commensurate with the simulation box size. Therefore, we are restricted to calculating out to values smaller than the box size. For us, this is a sufficient distance, and so this is not a problem.

In this thesis, all the neutron scattering patterns were calculated using structure factors from equation 1.10.

To go from the structure factors to the neutron scattering pattern, we use

$$NS_{tot}(k) = b_c^2 c_c \left[S_{cc}(k) - 1\right] + 2 b_a b_c \sqrt{c_a c_c} S_{ca}(k) + b_a^2 c_a \left[S_{aa}(k) - 1\right] \tag{1.11}$$

where b_x is the coherent neutron scattering length of species x, c_x is the concentration of species x, S_{xx} is the x-x partial structure factor, and $_a$ and $_c$ are anion and cation respectively.

This resultant function $NS_{tot}(k)$ is directly comparable with the experimental total neutron weighted structure factor.

1.6.4 Diffusion

The diffusion coefficient for any given species can be found from the *mean squared displacements* of the ions of that species by applying the Einstein relation

$$D_a = \lim_{t \to \infty} \frac{\left\langle \left| \delta \vec{r}_a\,(t) \right|^2 \right\rangle}{6t} \tag{1.12}$$

where $\delta \vec{r}_a\,(t)$ is the displacement of any ion of species a in time t. An alternative route to the diffusion coefficient is via the velocity auto-correlation function,

$$D_a = \frac{1}{3} \int_0^\infty \left\langle \vec{v}_a\,(t) \cdot \vec{v}_a\,(0) \right\rangle \,\mathrm{d}t. \tag{1.13}$$

One problem with the use of equation 1.13 it is that it is difficult to tell when the correlation function has died away to zero. This can lead to small, systematic, long-time correlations being neglected. Because of this, all the diffusion coefficients in this thesis were calculated with equation 1.12.

For examples of the curves produced by MSD calculation programs, and reading them, see section 2.2.2, especially figures 2.2 and 2.3. To read the graphs, we find the differential of the MSD graph, and take the value at the first proper minima, as is shown in figure 2.3. A more detailed description on using MSDs to find diffusion coefficients, including reading the graphs, can be found in chapter 2 of reference [12].

1.6.5 Viscosity

This is another dynamic property of interest, which is relatively easy to calculate. The shear viscosity η is given by the time integral of the autocorrelation function of an off-diagonal element of the stress tensor:

$$\eta = \frac{\beta}{V} \int_0^{\infty} \langle \sigma_{\alpha\beta}(t) \cdot \sigma_{\alpha\beta}(0) \rangle \, \mathrm{d}t \tag{1.14}$$

The viscosity is a multi-particle property. Being a property of the system as a whole, it is not possible to perform additional averaging over the N particles. This leads to η having much poorer statistical properties than the likes of D. It is possible to improve the precision of the data by averaging over different components, $\alpha\beta = xy$, yz, xz, x^2-y^2, $2z^2-x^2-y^2$ of $\sigma_{\alpha\beta}$.

To find the value for the viscosity, we integrate the average stress correlation function, and then read off the value at the first proper maxima, as seen in figures 2.4 and 2.5. As for diffusion, example curves can be found in section 2.2.3, and a full description is given in chapter 2 of reference [12].

Chapter 2

Run Lengths And Averaging

Most of the properties of interest are averages. We need to consider how long it will take them to reach their steady values, so we can ensure that our runs are long enough to give meaningful results. We also need to investigate if averaging over independent runs at the same state point will offer any improvements on the quality of the data. This is necessary because we want to test a theory which does not suffer from finite run lengths.

In this chapter, we look at averaging and parallelisation with computer clusters, and see how this can help us. We then consider some of the pitfalls associated with deciding on appropriate run lengths. Finally, we apply this to our systems of interest, to decide the run lengths and averaging to be used in our data gathering runs.

2.1 Parallelism and Simulations

We need to ensure that our results are reproducible. One of the things commonly done to ensure reproducability is to repeat every experiment a number of times. The repeated results can be used to verify that you have a stable value. They can also be averaged together, to improve the statistics of the results.

In computational simulations, typically the computing time is expensive and in short supply. Many people opt to run a simulation for as long as they can to get *a* result, instead of doing a number of identical shorter runs, which may not be sufficient to get *any* results from.

Over recent years, the move has started from using single, large computers to clusters of smaller computers. In many cases, a cluster of smaller computers working together offers a much better price/performance ratio than a single, large computer. However, programming for a large number of machines is significantly trickier than for one single machine, and often involves large communication overheads, which marr the performance.

2.1.1 Programming for Clusters

The Message Passing Interface (MPI) [18] is a language-independent way of programming for clusters. It defines ways to send, receive and manage data between a number of computers. It requires the programmer to explicitly code how the different *nodes* of the cluster will interact with each other, and how to parallise the task in hand. This is relatively hard, and the method carries with it a fair number of overheads, but when done well allows the job to be efficiently spread over a number of machines. There are a number of MPI implementations, the most common of which is MPICH [19].

An alternative model for clusters is to write locally multi-threaded jobs, and then use an environment like OpenMosix [20] to handle distributing threads around the cluster. This requires less programmer effort, and is suitable for different kinds of tasks (typically better for more independent, concurrent tasks) [21].

The other main model for clusters is PVM [22]. Programming this is similar to MPI, but the parallisation is handled differently, and offers some advantages and disadvantages over MPI [23].

A full description of these methods of programming clusters is well beyond the scope of this thesis. As a result of the level of support provided by the Oxford Supercomputer Center [24], the MPI environment was selected.

2.1.2 Using MPI with MD

As we have already seen, there are three problems we face - how to get repeatable data, how to easily get data from a number of nearby state points, and how to make effective use of the supercomputer clusters that we have access to.

One solution would be to write a series of MPI helper libraries, to co-ordinate the running of a number of independent jobs across a cluster. Such a scheme would be limited in run length by the speed of the individual cluster machines (it would not sub-divide the work of one job across machines). However, as we will see later, this isn't a large problem. The scheme should touch as little of the existing code base as possible, should be easy to use, and should allow one to quickly set up runs at a large number of state points. Thus, the task of collecting data for four adjacent temperatures, with four runs per state point, moves from being lengthy and problematic to fast and easy.

Such a system was writen, and more details on it are to be found in the appendix A.

2.1.3 Long Runs and Averages

In order to make use of the cluster, we need to consider a number of things. We need to find the minimum run length which will give us correct values. We need to see how averaging will affect our results. We need to consider the available resources - are lots of shorter runs better than a small number of larger runs?

One thing we must be very careful of is stable values versus long time values. Later on in this chapter, we will see several examples of this phenomena. A *stable value* is considered to be the repeatable value that is tended towards as a number of identical length runs are averaged. A *long time value* is considered to be the repeatable value that is tended towards as the run length is increased.

What we must be careful of is where the stable value at a given run length isn't the long time value. If we are not careful, we may pick a run length that gives repeatable but incorrect answers. To avoid this, we must ensure that when testing, we do runs at a number of different run lengths, and ensure that our chosen length gives the same answer as longer runs.

Using figure 2.4 as an example, we see that for 100,000 step runs, the stable value for the shear viscosity is around 1.1x10^{-13}au. For 200,000 step runs, the stable value is around 1.4x10^{-13}au. For 500,000 and 1,000,000 step runs, the stable value is the long time value of about 1.3x10^{-13}au. Were we to only do runs at 50k and 100k steps, we might erroneously decide that 100k was a long enough run, as it gave a repeatable answer. However, by doing longer runs, we see that while the 100k value is repeatable, it isn't correct!

For all the properties of interest, we must do test runs to decide on an appropriate run length and degree of averaging. We must do runs larger than our intended length, to ensure a repeatable value is the long time value. We should also check that we are applying enough averaging. This must be done for every property under investigation, as the results will differ between properties, especially between structural and dynamic ones.

2.2 $ScCl_3$ Testing

For this section, a number of runs of different length were done, and then a variety of averaging was applied to these. The aim was then to discover what the optimal run lengths and averaging would be for the properties of interest, given the computing resources available. For the $ScCl_3$ testing, the sets looked at were 100k runs at 1, 2, 4 or 8 averaged, 200k runs at 1, 2 or 4 averaged, 500k runs at 1, 2 or 4 averaged, and a single 1m run.

After looking at all the properties of interest, it was decided to do the data gathering runs at 400k steps, averaged over 2 independent runs.

2.2.1 Pressure

The first property of interest for $ScCl_3$ is the pressure. Figure 2.1 shows us the effects of the averaging and run length on the system's pressure.

Looking at 2 runs averaged, we see that the 200k and 500k runs have largely converged. The fact that at 4 runs, they are still converged together, but at a slightly higher value is of slight concern. However, the magnitude of the difference is very slight, and could possibly be considered a measure of the error.

We can see that the 500k and 1m runs are at the same value, which tells us that by 500k runs, the pressure value has converged with respect to run length. Since the 200k and 500k runs are converged at 2 and 4 runs, this tells us that by 2 runs averaged, the value has converged with respect to run averaging.

The chosen value of 400k and 2 runs averaged should be long enough and with enough averaging. It also falls in the centre of the graph, and as such should produce meaningful results.

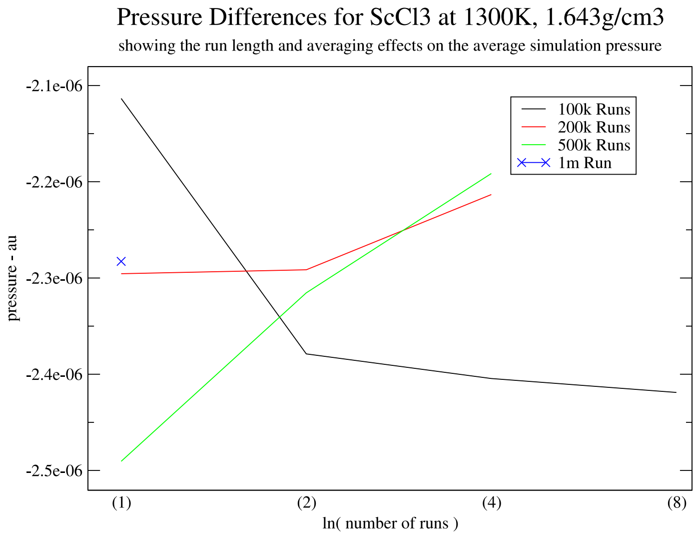

Figure 2.1: The Pressure of $ScCl_3$ as the run length is altered, and different numbers of runs are averaged over.

2.2.2 Diffusion

Another set of properties of interest are the diffusivities and conductivities. In this thesis, we have tended to focus on ion diffusivities, rather than on conductivities. In general, very similar results to the diffusivities are found for the Nernst-Einstein conductivity, since this is a weighted average of the diffusivities. Real conductivities require much longer runs, and so their values were not were not routinely calculated.

When looking at the run lengths and averaging to be used, as seen in figure 2.2, only anion diffusivities have been studied in detail. However, it was checked that the cation diffusion and the Nernst-Einstein conductivity showed identical trends, as would be expected.

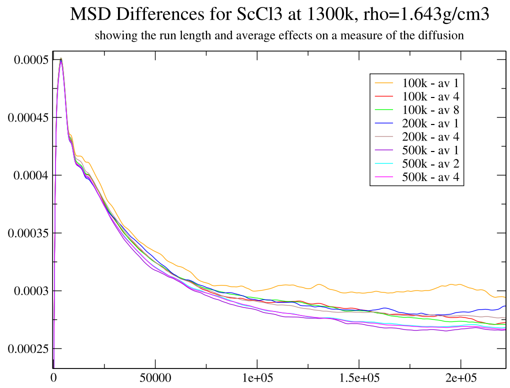

Figure 2.2: The effect of run length and averaging on the Cation MSD Difference values for $ScCl_3$.

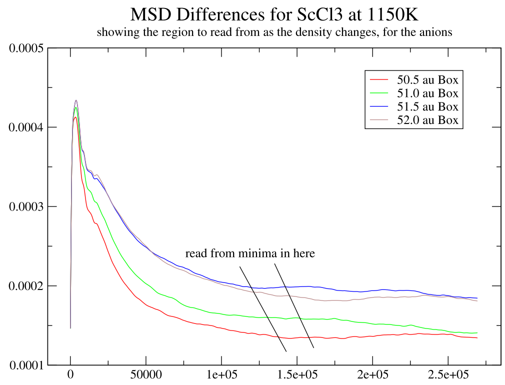

Figure 2.3: Using several similar state points graphs to aid in reading off MSD Difference values.

The low average runs at 100k and 200k have close, but slightly incorrect values. With the 100k runs, at high averages, the results tend towards the correct value, but are still very slightly off. The high average 200k runs are better than their 100k equivalents, but are still very slightly off. All the 500k runs give the correct values. However, at low averages, it is very hard to see exactly where to read off. Adding averaging doesn't alter the (correct) value, however it does make it significantly easier to identify the region of the graph to read from. The difference between an average of 2 and 4 is slight, and can be equally achieved by comparing the graphs from two similar runs (which will have almost identical read-off points).

As such, using a run length of 400k, and averaging over two runs wouldn't be quite good enough to be sure, if we were looking at a run on its own. However, as we are always looking at a number of similar runs, it is fine. This can been seen in figure 2.3.

2.2.3 Viscosity

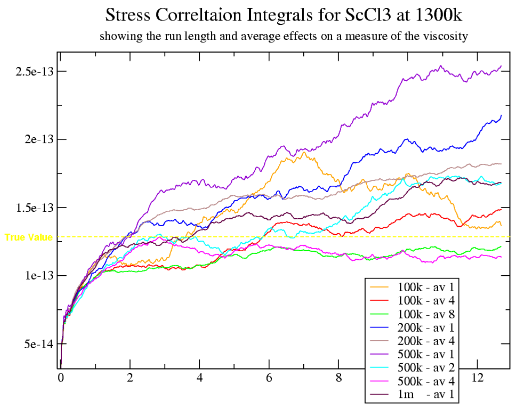

Figure 2.4: The effect of run length and averaging on the Stress Correlation Integrals for $ScCl_3$, with the long time large average value indicated.

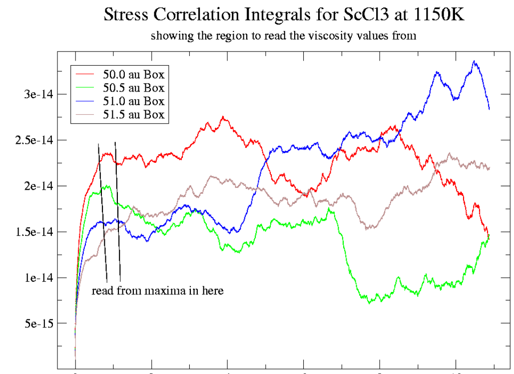

Figure 2.5: Using several similar state points graphs to aid in reading off Stress Correlation Integral graphs.

We can see that all of the graphs in figure 2.4 are very noisy. Even runs done during the initial investigation of 2 million steps showed a fairly high degree of noise. This can be traced back to the much lower degree of averaging during the calculation than in other properties, as discussed in section 1.6.

The 100k runs are quite off, coming in with a values too low. The higher average runs do have a fairly consistent value, however it's not the correct one. The 200k runs are also off, with values that are too high. For the 500k runs, the single run has almost the correct value, but one might be tempted to read off at a different (incorrect) point. The higher average runs give the same (correct) value, and the region to read from is quite easy to see. The single 1m run has the same value as the 500k runs, indicating we've reached the long time value.

With 400k runs and only 2 runs averaged, spotting the peak to read from might be slightly tricky. However, we can take advantage of the fact that runs at similar state points will have very similar regions to read from. As such, provided we plot a number of similar runs at once, we are able to reliably read off the correct values, having easily spotted the peak region by looking at the multiple curves. A 400k run is also a long enough run to get the correct value out, providing you can find the correct point to read at, as seen in figure 2.5.

2.2.4 Neutron Diffraction Pattern

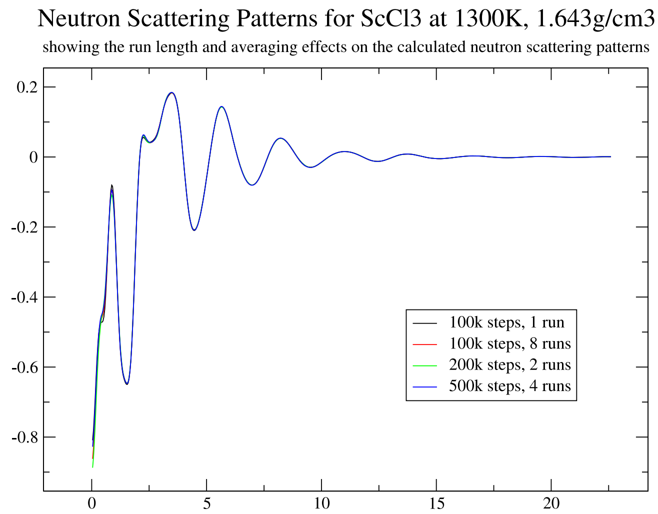

Figure 2.6: The effect of run length and averaging on the Neutron Scattering Patterns for $ScCl_3$, showing the limited effects seen.

As seen in figure 2.6, the run length has very little effect on the neutron scattering pattern. Even the shortest run with the minimal averaging gives almost the same scattering pattern as the longest run with the greatest averaging.

This highlights the big difference between the run lengths required for structural properties compared to dynamic properties. The structural patterns are given by the common (typically ground) states and positions. In dynamics, most of the processes depend on infrequently occupied rare (often excited) states and positions. The neutron pattern looks at where the ions normally are, while conversely the diffusion requires them to move through locations where they are normally not. Thus, in order to get the same amount of time in the states which have the most effect on the behavior of interest, we must run the simulation for much longer for dynamics than for structures.

If we were only looking into structural issues, we could get away with very short runs and minimal averaging (if any). As it is, the chosen run length and averaging is more than enough.

2.3 Theoretical Model Testing

For this section, again a number of different length runs were done, with a variety of averaging strategies applied to them. As with the $ScCl_3$ work in section 2.2, the aim was to discover the optimal run lengths and averaging, for the properties of interest. In the testing, the sets were 50k runs at 1, 2 or 4 averaged, 100k runs at 1, 2 or 4 averaged, and 200k runs at 1 or 2 averaged.

After analysing the results for the properties of interest, it was decided to perform the data gathering with runs of 100k steps, averaged over 2 independent runs.

2.3.1 Pressure

Figure 2.7 tells a similar story to the $ScCl_3$ work in section 2.2.1, but with slightly smaller run lengths. We can see that 200k runs averaged twice offers very little advantage over 100k runs averaged twice. As such, opting for the shorter run length should be fine. The difference between 100k at 2 and 4 runs is small in absolute terms, and so is nothing to worry about. It may be a useful indication of the errors in recording the pressures.

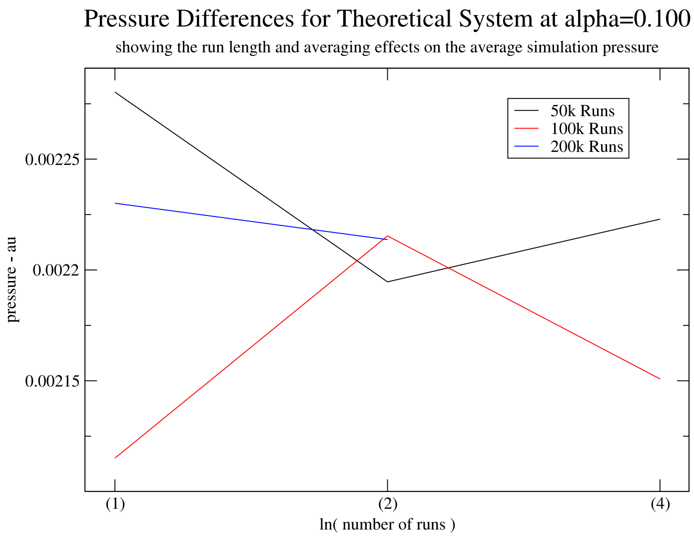

Figure 2.7: The Pressure as the run length is altered, and with different numbers of run are averaged over.

2.3.2 Diffusion

Looking at figure 2.8, we see similar things as with $ScCl_3$ (section 2.2.2). The lowest run length and averaging doesn't give the correct value, though the high averaging almost does. With the low averages at 100k and 200k, the graphs are quite noisy, so it's hard to read off the correct value. The higher averaging at 100k and 200k avoid this, so we're fine with using 2x100k for our data gathering.

2.3.3 Structure Factors

Looking at figures 2.9 and 2.10, we see only limited differences between the graphs. Apart from the very lowest run lengths and averaging, all of the runs give perfectly fine structures. As such, 2 runs at 100k is more than good enough to give us our structure data.

This results re-inforces that fact that dynamics are much harder to find than structures, and require much longer runs. Were we only studying structures, two 50k runs would have been good enough for all except the very highest resolution studies.

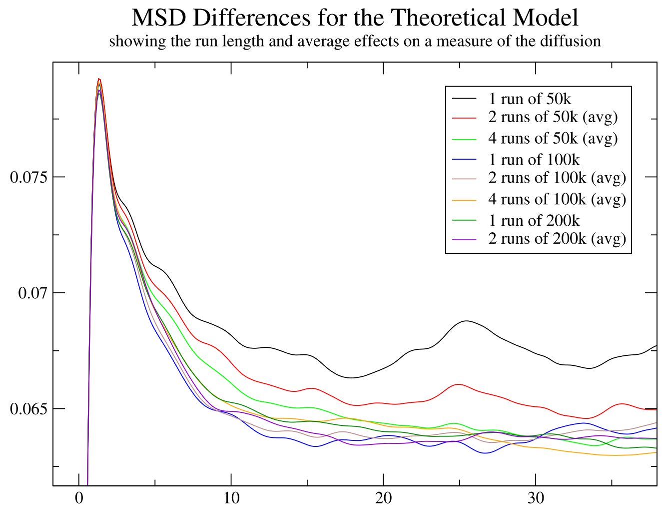

Figure 2.8: The MSD Differences for the cations as run length is altered, and different numbers of runs are averaged over.

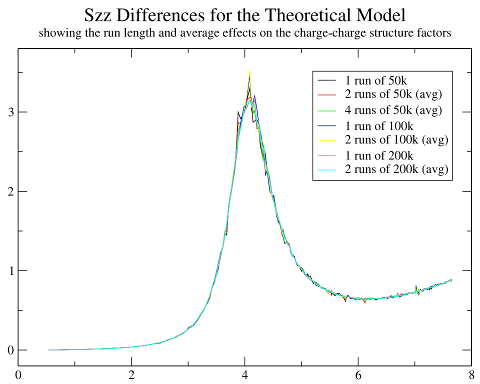

Figure 2.9: The Charge-Charge structure factors for the theoretical system as run length is altered, and different numbers of runs are averaged over.

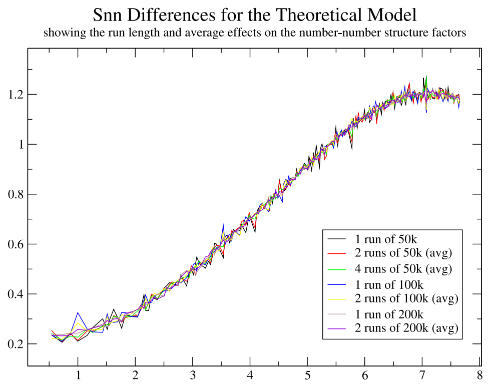

Figure 2.10: The Number-Number structure factors for the theoretical system as run length is altered, and different numbers of runs are averaged over.

Chapter 3

System Size Effects

As was seen in section 1.1, we have to apply a number of tricks to deal with the small simulation cell sizes. Some of these will affect our results. Because the objective is to test a theory, we need to find the system size at which results will be close to those of an infinite system.

In most cases, the system size (box size and number of particles) is constrained by the available computing resources. The larger the number of particles in the system, the longer the runs will take. We therefore tend to pick the largest system size that we can, which still gives manageable run times for the run lengths we've decided are needed.

While we may have only limited control over what system size we pick, we should still try to test what sort of an effect our chosen size will have on the results we produce.

3.1 Effects on $ScCl_3$

For this section, all runs were done for 400,000 steps, and averaged over two independent runs, as decided from chapter 2. The exceptions are the 864 ion run, where a system failure meant the runs were very slightly shorter, and for the 1372 ion run (marked with a red cross), which was done for 500,000 steps and with only one run. All our data gathering runs in chapter 5 were done with 500 particles.

Looking at figure 3.1, we see large effects on the pressure at the small system

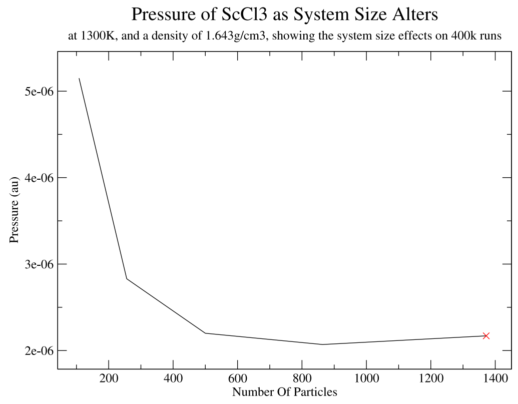

Figure 3.1: The System Size Effects on the pressure in $ScCl_3$.

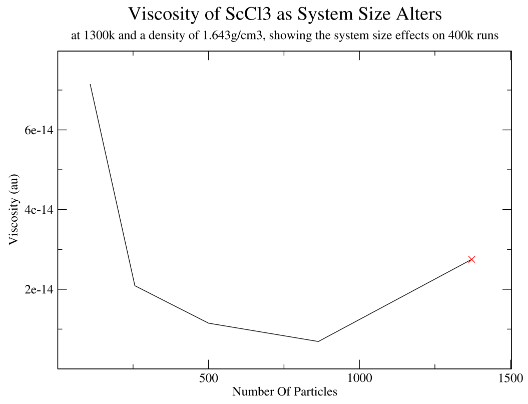

Figure 3.2: The System Size Effects on the viscosity in $ScCl_3$.

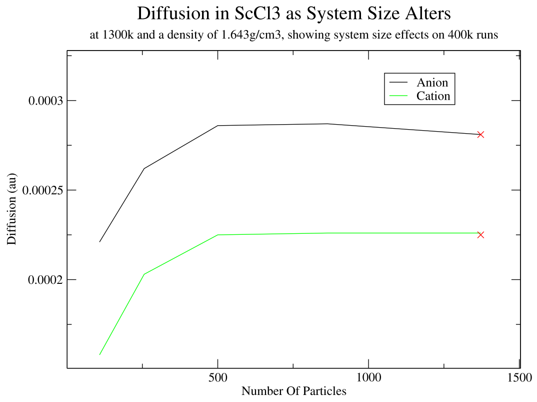

Figure 3.3: The System Size Effects on the anion and cation diffusivities in $ScCl_3$.

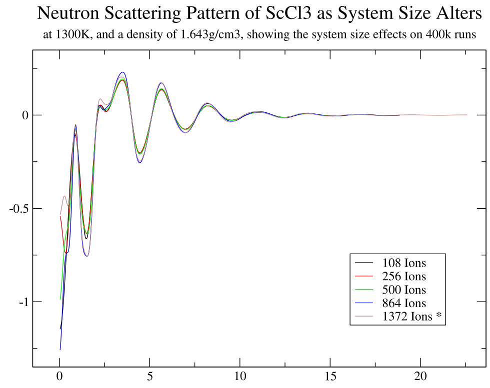

Figure 3.4: The System Size Effects on the Neutron Scattering Pattern of $ScCl_3$.

sizes. As we move into larger systems, the pressure seems to converge. It seems that our chosen size of 500 particles has quite small system size effects for pressure, so its result should be fairly representative.

In figure 3.4, we see remarkably little alteration in the neutron diffraction patterns as we increase the system size. The second peak, and the first minima show slight changes, but the pre-peak is largely unaffected. Since the pre-peak (at ≈ 1 au^{-1}) depends on intermediate range ordering (see section 5.1.1), we might have expected a bigger effect on it with the smaller box sizes. Given the importance of intermediate range ordering on the neutron pattern, it would be interesting to run a simulation with a very much larger box size, and see if anything changes. Since the neutron scattering pattern requires much smaller run lengths than the other properties in order to give good values, such a study wouldn't be too hard to run.

Figure 3.3 shows that the diffusion is affected by system size for small box sizes. However, for the larger box sizes, there is very little variation at all. It seems that our value of 500 is the first for which the diffusivities have converged, and as such its values should be fairly representative.

Figure 3.2 shows a more complex picture for viscosity. Again, small systems are insufficient for good values. However, the picture doesn't settle as much for larger systems as the other properties. Without data for even larger systems, we can't be sure what value the viscosity will eventually tend to. This means that our values from 500 ions may not be properly representative, but hopefully any trends shown will be correct. Without a run from an even larger system, we can't quantify how far our 500 ion system's values will be out.

In general, the system size effects on $ScCl_3$ at 500 particles are slight, except for the viscosity.

3.2 Effects on Theoretical System

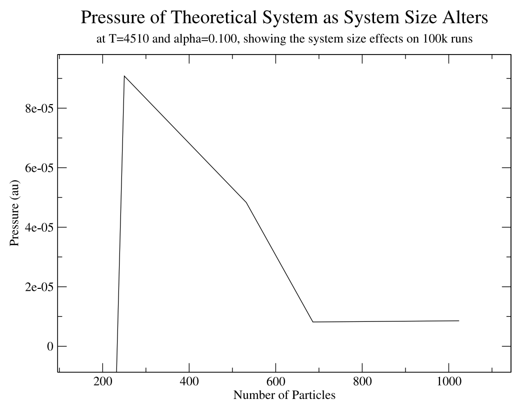

Figure 3.5: The System Size Effects on the pressure in Theoretical Model.

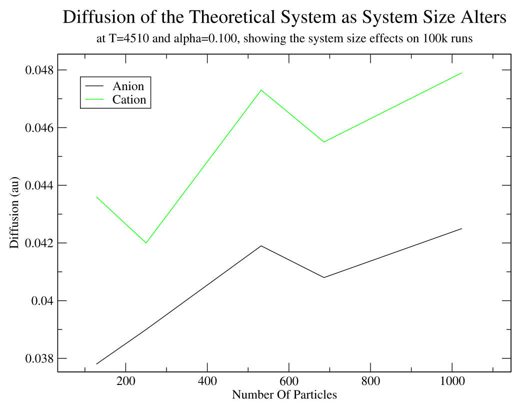

Figure 3.6: The System Size Effects on the diffusivities in the Theoretical Model.

For this section, all runs were done for 100,000 steps, and averaged over two independent runs, as decided from chapter 2. All our data gathering runs in chapter 4 were done with 686 particles.

Looking at the pressure from figure 3.5, we see a very marked effect for small box sizes, with a large variation in values. However, at larger box sizes, the value converges quite nicely. It appears that our values from 686 ion systems will be representative.

Figure 3.6 shows two things. Firstly, we can see that the diffusivities have a complex relationship with box size, but one which appears to converge for the larger box sizes. Secondly, we see that the anion-cation diffusion difference is neither constant, nor dependent on the magnitude of the values. Clearly, some complex relationship exists, which cannot be easily explained.

From figure 3.6, we can't be entirely sure how representative our diffusion coefficients from a 686 ion system will be. It looks that the values are bouncing around a consistent value in the last 3 system sizes, however without data from an even larger system, we can't be sure. We hope that our values will show all the correct trends, even if it may later transpire that their absolute values are very slightly out.

A system size of 686 ions seems like it should give fairly representative results, but data from a larger run than 1024 ions will be required to verify this fact.

Chapter 4

Theoretical Model Work

In this section, we will look at how MD studies may be used to confirm and augment the predictions of the PSMS model.

The data from every state point are in fact the average of two independent runs at that state point. This allows us to have greater confidence in our results than might otherwise be the case.

Scaled units (see section 1.5) are used with this model.

4.1 Validating The Theory

Before we can go on to make predictions on the dynamic properties of the theoretical system, we need to ensure we can generate overlapping results. This has two aims, the first of which is to ensure that we are correctly modelling the same system. The second of these is to validate the theory's results. As we saw in section 1.4, the theory makes a number of approximations, and is not thermodynamically consistent, while MD is. If we do get an agreement in results, we have shown that the results from the theory are correct.

As has already been seen, the theory entirely concentrates on structural aspects. With this in mind, we need to check the overlap between our structural data and the theory's. The structural data in question is the S_{zz} (charge-charge structure factor), the S_{nn} (number-number structure factor) and the RDFs. These should be compared for a number of different temperatures and polarisabilities, to ensure a good match.

As we can see from figures 4.1, 4.2, 4.3 and 4.4, we do have the required good agreement between the simulations and the theory.

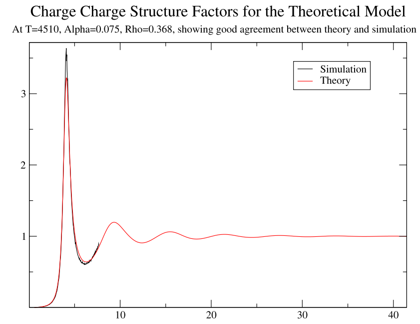

Figure 4.1: S_{zz} match at α=0.075, T=4510k

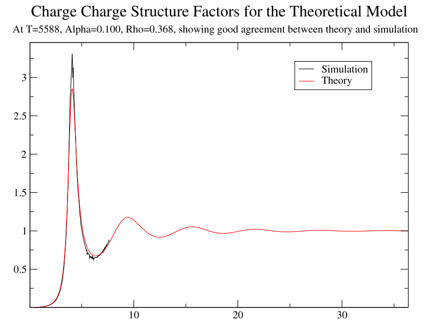

Figure 4.2: S_{zz} match at α=0.100, T=5588k

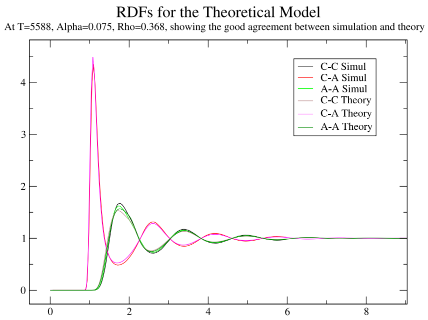

Figure 4.3: RDF match at α=0.075, T=5588K

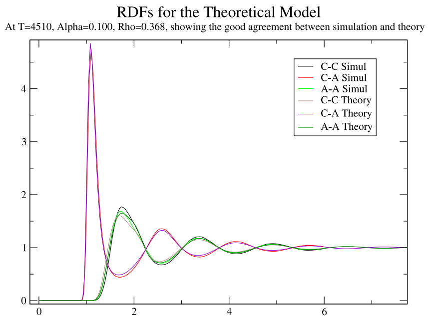

Figure 4.4: RDF match at α=0.100, T=5588K

However, we have not just checked for good agreement at only these four state points. Thanks to the use of the MPI code (see section 2.1.2), we have structural data from a very large number of state points. Correct matches were checked at a very wide range of densities and polarisabilities, as well as at three separate temperatures. This allows us to have a high degree of confidence in the accuracy of the theory's predictions, as well as in our ability to model the same system as it.

Without the MPI system, it is hard (and typically quite time consuming) to thoroughly test theories. The MPI method is not only a faster simulation method (when covering a number of state points), but through its automation of running

MD simulations at a wide range of state points, is a large help with the development of new theories.

4.2 Cavitation

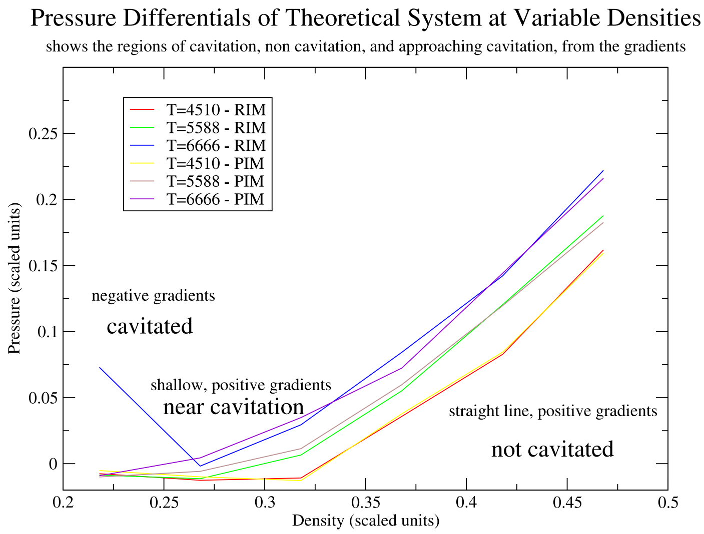

Figure 4.5: Showing the regions of cavitation and non cavitation, and how density, polarisation and temperature affect these.

One of the striking predictions of Gray-Weale's theory was a very strong effect of polarisation on the apparent phase separation of the fluid into high and low density regions. Since the critical behaviour of ionic systems has been the focus of much theoretical interest, it is important to validate this prediction.

At lower densities and temperatures, the system can become mechanically unstable [25]. The point of mechanical instability need not be near thermodynamic instability, and happens to fall within the region of interest. When a system is mechanically unstable, it is liable to cavitate. On cavitation, the system splits into two parts, a higher density region and an empty region. The spinodal curve is the boundary between the unstable region (mechanically unstable, cavitation occurs here) and metastable region (unstable, but requires a trigger to cavitate). The binodal curve is the boundary between the metastable and stable regions.

The theory predicts that polarisability will inhibit cavitation. Specifically, it predicts a strong supression of large scale number density fluctuations with polarisation. However, as the theory is thermodynamically inconsistent, these results previously couldn't be trusted. Looking at figure 4.5, we see that prediction does appear to be the correct. Figure 4.5 is a plot of $\frac{dp}{d\rho}$ for a number of temperatures in rigid and polarisable runs. These graphs validate the theory's predictions, since the polarisable run curves occur above their rigid counterparts. The plot of the pressures themselves can be seen in figure 4.10. It can be easily seen that the pressure differential graph (4.5) is much better for spotting the regions of cavitation than the pressure graph (4.10), which is why it has been used. To spot cavitation in figure 4.5, we need to look at the gradient. Straight, positive lines indicates that the pressure increases with increasing density, as would be expected. When the gradient is only very slightly positive, the system is close to cavitation. When the gradient is negative, the system has cavitated. As we decrease the density, the pressure increases, because the cavity is growing.

The figure also shows that in many cases, polarisability will prevent cavitation for one jump in density, a change of 0.05 or about 20%. The graph also highlights the regions approaching cavitation, where the dynamic properties may be affected by impending cavitation, but where cavitation hasn't occured. In these areas, the dynamic properties will may not follow the non cavitated trends, owing to the unusually large distances between the ions. As it may not be immediately obvious that the system is affected by impending cavitation, the differential graph is a good way to spot such regions. However, the effect to which the proximity to the cavitation point will affect the dynamic properties is unknown, and would require a large number of simulations around the point cavitation to study.

Unfortunately, we only have data on 3 different temperatures, which is not sufficient to quantify the temperature effect on preventing cavitation. Runs at a few more temperatures would be required to do this. In general though we can see that increasing the temperature inhibits cavitation, as would be expected to be the case.

Figure 4.5 shows us the effect that polarisation has on the occurance of cavitatition, which is not insignificant. This means that a system such as the RPM will be insufficient for properly predicting cavitation. Many studies (such as [25]) of phase separation use the RPM, to keep the systems simple enough to allow study. However, these results indicate that polarisation does have a not insignificant role to play in cavitation and possibly the coexistence curve. As such, future studies may wish to include it.

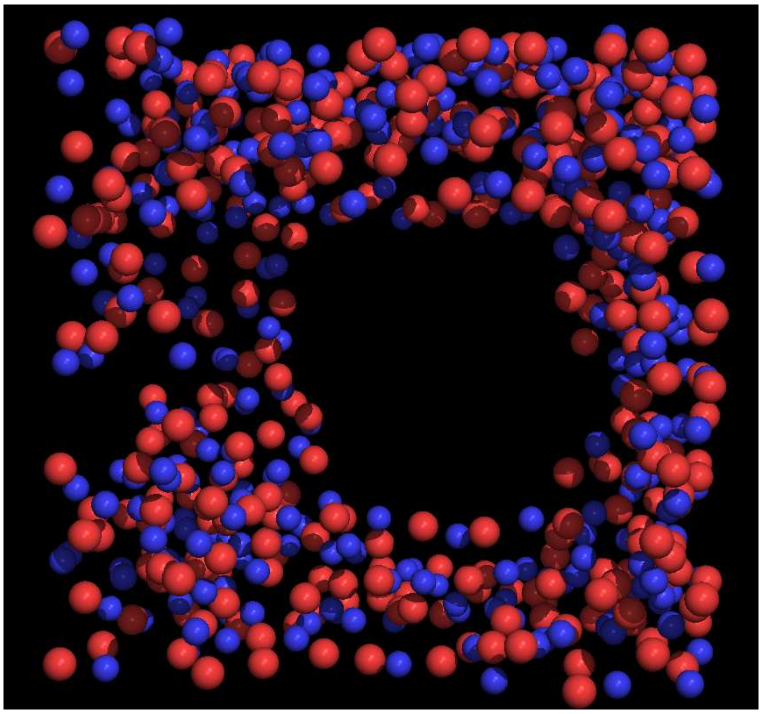

Figure 4.6: The system having cavitated. Occurs at low density, and is more likely without polarisability.

Looking at figure 4.6, we notice that there is charge neutrality at the surface of the cavity. On cavitation, S_{NN} becomes large at small k, as the cavitation causes a hole to form in the liquid. However, $\lim_{k \to 0}(S_{zz}) = 0$, i.e. charge neutrality is maintained, since anything else would incur a large energy penalty.

At the surface, the ions will be in an asymmetric environment. In any asymmetrical environment, polarisation is important. In $ScCl_3$, polarisation has a big effect to play on how the chlorides behave in their asymmetric, 2 coordinate about scandium environment, as we shall see in chapter 5. We will see the charge effects shortly.

It has been shown [14] that the presence of polarisation leads to much rougher liquid-vapour surfaces for ionic fluids.

This (several effects, including entropy, effects may oppose each other)

can be partially explained by the need to minimise the energy, which favours the asymmetric environments where the dipoles aren't cancelled. Another factor is the system entropy - the rougher surface has more potential configurations, and hence a greater entropy.

These might be expected to cause polarisation to incite cavitation. However, we find the opposite is true, and polarisation inhibits cavitation.

Clearly there are a number of factors at play here. In the rigid system, ion-ion charge interactions have a notable effect. By cavitating we increase the density in the region where the ions are; a process which is coulombically favourable, but which may or may not lead to an increase in free energy. Also, as we decrease the density, the system has fewer and fewer potential configurations it can be in, and eventually there is a large entropic gain by forming a fault. In the polarisable system, in addition to the ion-ion charge energy (which favours density increase and cavitation), we also have the dipole effects of polarisation. The polarisation shields like ion interactions, which means the system will have more configurations at a given density than a rigid system, and will be more able to respond to a stress without cavitating. Finally, we have an extra entropy term arising from the dipoles on the polarisable anions at the surface, which have a variety of possible alignments. All of these together lead to polarisation inhibiting cavitation.

4.3 $\epsilon(0)$ Results

As we saw in section 1.5, the use of corresponding states and reduced variables increases the applicability of theories. Previously, no work has been done on finding an appropriate reduced parameter for describing the dynamic properties of simple polarisable ionic systems. In the next two sections, we will consider two candidates for a dynamic properties reduced parameter, and see how well they work. We will examine the variation of the diffusion coefficients for a series of systems with different polarisabilities (and densities in section 4.5), and see if the behaviour can be accounted for by a single dimensionless quantity, which measures the strength of the polarisation effect.

Two of the obvious choices for our dimensionless quantity (which we will be plotting our data against) are $\epsilon(0)$ (the static dielectric constant) and ϵ_∞ (the high frequency dielectric constant). It should be noted that these dielectric constants are associated with the anion's dipoles, and not with charge rearrangements. As was seen in section 1.6, many of the properties of interest are themselves average properties. Since the static dielectric constant covers averages over long timescales, we postulate that the static constant could be the better of the two dielectric constants to consider.

The value for $\epsilon(0)$ can be found by fitting to the first part of S_{zz} curve. At low values of q, the structure factor may be expanded out in the form $S(q) = \Lambda_D^2 q^2 + O(q^4)$. From the Stillinger-Lovett conditions, we know that $(\Lambda_D^{(rim)})^{-2} = 4\pi\beta\rho$. We also know that if the rigid ions were embeded in a dielectric continuum, then $\Lambda_D^{-2} = \frac{4\pi\beta\rho}{\epsilon_0} = (\Lambda_D^{(rim)})^{-2}/\epsilon_0$, and hence $\Lambda_D^2 = \epsilon(0)(\Lambda_D^{RIM})^2$. In these, and future equations, $\beta = \frac{1}{k_B T}$.

By applying the above relationships to our simulation results for the polarisable fluids, we can come up with an expression for the structure factor in terms of $\epsilon(0)$, an effective dielectric constant. This allows us to find values for $\epsilon(0)$ from curve fitting to the low q structure factor. The equation is:

$$S(q) = (\Lambda_D^{RIM})^2 q^2 \epsilon + O(q^4) \tag{4.1}$$

As was seen in figures 4.2 and 4.1, there is a very good agreement between the simulation values and the theory values for S_{zz}. Equation 4.1 shows that to find our values of $\epsilon(0)$, we can fit to the low q values of S_{zz}, provided we have previously found Λ_D^{RIM}. Since q is inversely proportional to distance, the lower the q value required, the larger the simulation box size needs to be. As has already been mentioned, one of the advantages of the theory is it allows us to get information for larger length scales than the simulation would yield, for similar expenditure in computing time.

The data available from the simulation doesn't go out to low enough q to allow an accurate curve fit. The fitting is very sensitive to the number of values available, and for the simulation sizes used, it isn't possible be confident in a fitted value from the data. As such, it isn't possible to accurately determine a value for $\epsilon(0)$ from the simulation runs. Instead, the theory was used to calculate values for S_{zz} and $\epsilon(0)$. The low q values for S_{zz} were plotted against the simulation values, and it was ensured that there was a good match. Since the two agreed on the lowest q S_{zz} values accessible in the simulation, it is possible to use the theoretical values for $\epsilon(0)$.

Unfortunately, the theory runs have problems converging at large values of α, which leads to us only having S_{zz} curves for the smaller polarisabilities. This means we can only plot out the diffusivities against $\epsilon(0)$ for a limited number of points.

Looking at figure 4.7, we see that both the anions and cations show a temperature dependence in how their diffusivities change with $\epsilon(0)$. We also see that generally, the cations have higher diffusivities than the anions.

As a result of the limited number of data points (both in terms of temperatures and against $\epsilon(0)$), it is not possible to accurately quantify the relationship of the diffusivities to $\epsilon(0)$ and temperature. Studies at a number of extra temperatures (both a wider range, and with smaller jumps) are really needed. In order to get

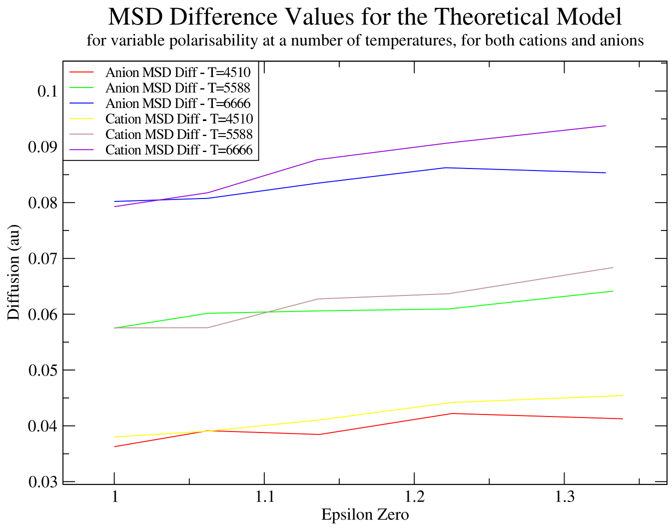

Figure 4.7: Showing the effects of polarisability (via $\epsilon(0)$) on the diffusivity. Cations show greater diffusivity, and both ions show a temperature dependence. The temperature is in scaled units.

these extra data points, either the theory will need tweaking to allow it to converge at higher polarisabilities, or the MD runs will need to be done with larger box sizes (allowing better low q fits, but with a large increase in computation time).

Looking at the graphs, it is not possible to come up with any sensible empirical fits as we do in other areas. The temperature dependence is simply too complex to be fitted to a simple form, and there is no point producing an empirical relationship with fractional temperature dependence powers based on such a small data set. However, our data is sufficient to show the general trends in behaviour - that there is a temperature dependence in the ion's diffusivities, and that the cations tend to diffuse slightly faster than the anions.

From the above, it appears that $\epsilon(0)$ is not a good choice for a reduced parameter to describe the dynamic properties.

4.4 ϵ_∞ Results

Unlike $\epsilon(0)$, ϵ_∞ has a very simple relationship with polarisability, for small values of $\alpha_0\rho$, such as those examined here. For this region, it is given by the Clausius-Mossotti equation:

$$\frac{\epsilon_\infty - 1}{\epsilon_\infty + 2} = \frac{4\pi\rho_{(anion)}\alpha_0}{3} \tag{4.2}$$

$$\epsilon_\infty = \frac{3 + 8\pi\rho_{(anion)}\alpha_0}{3 - 4\pi\rho_{(anion)}\alpha_0} \tag{4.3}$$

This makes it much easier to calculate and work with than $\epsilon(0)$, as we require no graph fittings, and can therefore get data out to much higher polarisabilities.

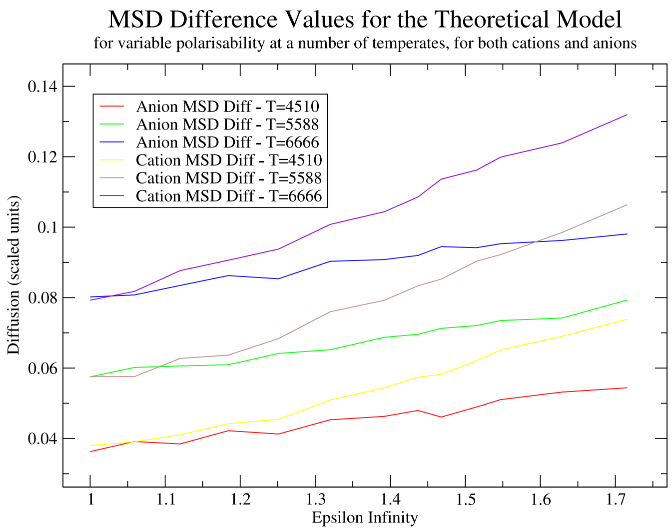

Figure 4.8: Showing the effects of polarisability (via ϵ_∞) on the diffusivity. Cations show both a temperature dependence and a greater polarisabilty effect, while Anions don't.

Looking at figure 4.8, we see that we largely have straight lines. The cations have a greater polarisability effect than the anions, and the cations also show a temperature dependence. We shall propose a possible explanation for this dependence shortly.

It is possible to come up with empirical relationships for the diffusivities in the regions under investigation. However, we do not have data from a sufficient number of temperatures to properly find the temperature dependencies of the diffusivities. As such, the following relationships should be treated merely as empirical fits, and not as indicative of some underlying theoretically justifiable relationships.

Anion (polarisable species) diffusion appears to show no temperature dependence in the region of investigation. Its diffusion alters with ϵ_∞ (alpha changes) as:

$$\text{Anion MSD Difference} = RIM + 0.027\epsilon_\infty \tag{4.4}$$

Cation (rigid species) diffusion does appear to show a simple temperature dependence. Its diffusion alters with ϵ_∞ (alpha changes) as:

$$\text{Cation MSD Difference} = RIM + \frac{3.73\epsilon_\infty}{\beta} \tag{4.5}$$

As the Nernst-Einstein conducivity is the species weighted average of the ion diffusivities, it also shows a temperature dependence. Its value alters with ϵ_∞ as:

$$\text{Nernst-Einstein Conductivity} = RIM + \frac{2.63\epsilon_\infty}{\beta} \tag{4.6}$$

In all of the above, the diffusion coefficients are realted to the MSD Difference by a constant scaling factor.

We can see that the cation's diffusivity is showing a temperature dependence with the anion's polarisation, and yet the anion's doesn't show such a dependence. To attempt an explanation of this behaviour, we need to consider the effect of polarisability on the ion's movements. In a very simple manner, the diffusion can be considered as ions moving from one site largely surrounded by opposite ions, through a small gap (typically between opposite ions), and out into a new site. The motion will most likely be between opposite ions, as the energetics of moving close to like charged ions will largely preclude this as a migration path. From figure 4.9, we can see that the cation's migration will be assisted by the dipoles induced in the surrounding anions. However, no such dipoles are found when an anion moves between cations. It seems likely that the presence of the dipoles in the cation case causes a marked difference in the activation and transition state energies between the anions and the cations. This manifests itself both in the greater diffusivity for the cations compared to anions, and in the temperature dependence that is seen.

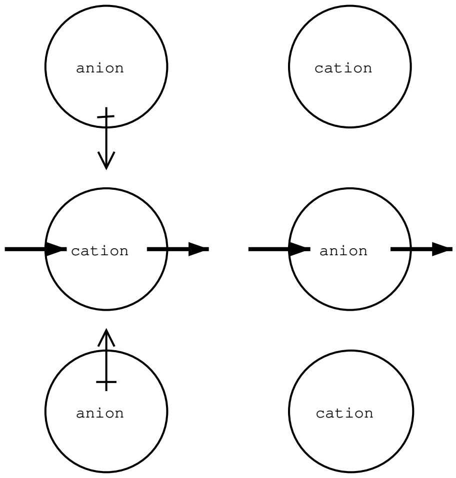

Figure 4.9: A stabilising dipole is generated during cation migration but not during anion migration. This affects the transition state energies, and may explain the diferening temperature dependencies.

The cations are more mobile than the anions. In both, we see a cation temperature dependence, albeit a more simple one for ϵ_∞. The big difference is with the anions, where there is apparently no temperature dependence with ϵ_∞, while a complex one is seen for $\epsilon(0)$. They are equally as noisy, as we are using the same data, just picking where along the x-axis to plot the data. It is simply how the x value changes with polarisation that affects what temperature dependence we will see.

From these two sections, we can see that ϵ_∞ seems to be a better reduced parameter for diffusion than $\epsilon(0)$. However, we do still have an apparently simple temperature dependence for the cation diffusion, which isn't ideal. More data is required to verify that the temperature dependencies really are as simple as they appear. What we can say is that for the time being, ϵ_∞ looks like our best bet for a reduced parameter, despite it being on a different timescale to the properties of interest.

4.5 Variable Density

In section 4.2, we looked at how polarisation and density affected the mechanical stability of the system. We were looking at the effects of density on a polarised ($\alpha_0 = 0.100$) and a rigid system ($\alpha_0 = 0.000$), with an eye to cavitation. In this section, we will be looking at the effects on diffusion, for the non-cavitated regions.

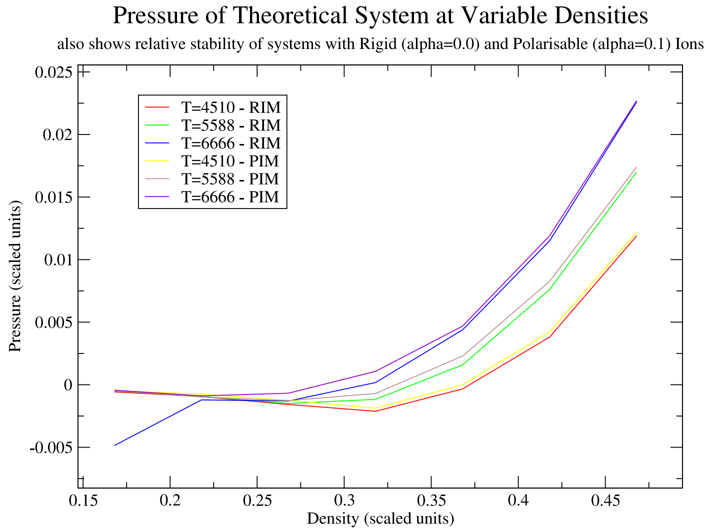

Figure 4.10: Showing the effect of density and polarisation on the Pressure. The negative gradients indicate a region where cavitation has occured.

Firstly, we need to use figures 4.5 and 4.10 to ascertain the regions of cavitation and non cavitation. These have been marked onto figures 4.11 and 4.12, so we can be sure to only include the regions of mechanical stability in our studies on diffusion.

Looking now at these non-cavitated regions in the figures 4.11 and 4.12, we see that the graphs of diffusion against density for these regions are made up of straight

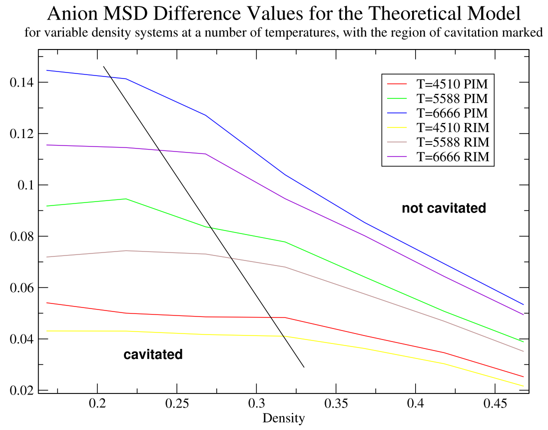

Figure 4.11: Anion diffusivity with temperatue and density. Polarisability makes the system more mobile.

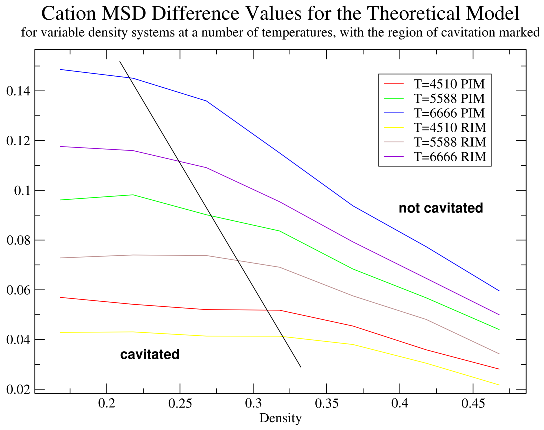

Figure 4.12: Cation diffusivity with temperature and density. Polarisability makes the system more mobile.

lines. It is possible to express the diffusivities for these regions with empirical equations specific to the appropriate values of α and β:

$$\text{for PIM at } \alpha\text{=0.100, Anion MSD Difference} = \frac{500 - 815\rho}{\beta^2} \tag{4.7}$$

$$\text{for PIM at } \alpha\text{=0.100, Cation MSD Difference} = \frac{530 - 830\rho}{\beta^2} \tag{4.8}$$

$$\text{for RIM } (\alpha\text{=0.000}), \text{ Anion MSD Difference} = \frac{440 - 700\rho}{\beta^2} \tag{4.9}$$

$$\text{for RIM } (\alpha\text{=0.000}), \text{ Cation MSD Difference} = \frac{430 - 670\rho}{\beta^2} \tag{4.10}$$

$$\text{for PIM - RIM, Anion MSD Difference} = \frac{70 - 120\rho}{\beta^2} \tag{4.11}$$

$$\text{for PIM - RIM, Cation MSD Difference} = \frac{100 - 160\rho}{\beta^2} \tag{4.12}$$

Unfortunately, with data from only two values of α, it is not possible to test the applicability of equations 4.4 and 4.5 to the above relationships. We would need to run simulations at a wide range of polarisabilities (probably a minimum of three more, but ideally a similar number to in section 4.4) to test this.

Also, with data from only three temperatures, it is not possible to be confident about the generality of the temperature dependences seen. As such, the above equations must only be regarded simply as empirical relationships for the regions under study, and not indicative of some theoretically justifiable global relationship. Only studies at a wider range of temperatures, coupled with theoretical study would be sufficient to achieve this. However, we can be relatively sure of the trends - that the cations are more mobile than the anions, polarisability increases mobility, as does temperature, while increasing density reduces the diffusion coefficients.

In this section, we have only considered the diffusivity data for the non cavitated region. In the cavitated region, the values have much less significance, as the system contains large empty spaces. It might be of interest to try to calculate a density for the ions only (i.e. without the cavitated space), possibly via a Monte-Carlo filling method on the cavity. We could then calculate the diffusion coefficients for a normal system at this density. The difference in the diffusion coefficients between these two systems will tell us something about the effect of the cavity surface on the diffusion. We will expect the cavity surface to cause there to be two different diffusion rates, one parallel to the surface and one perpendicular to it. The comparison of the diffusion coefficients may give some insights into the relative magnitudes of these, as so is probably worth of future study.

Chapter 5

Investigating $ScCl_3$

5.1 Main Results

For the investigation of $ScCl_3$ [2], runs were done for a number of densities and temperatures. The temperatures used ranged from 1150k to 1300k, which is around the experimental melting temperature. The box sizes used varied from 50.0 au to 53.5 au, which is around the experimental density for the temperature range. Also, a RIM run at 1250k was done, to look at the effect of polarisability on this more complex system.

All runs were done for 400,000 steps, and data was taken from an average of 2 independent runs at any state point (see section 2.2). The simulations were done with a system of 500 ions (125 molecular units).

5.1.1 Neutron Scattering

For this section, the experimental data comes from reference [26]. It was measured at a temperature of about 1250K, and a density of around 1.643gcm^{-3} (corresponding to a simulation box size of 50.532 au).

Looking at figure 5.1, we can see that our system has quite a good agreement with the experimental data when we use polarisability. The effects of temperature and density seem to be quite slight. They affect the pre-peak (known in experimental circles more commonly as a First Sharp Diffraction Peak or FSDP), and the first part of the main peak. Otherwise, the effect on the tail is minimal, with almost no changes in peak heights or the period of the fluctuations.

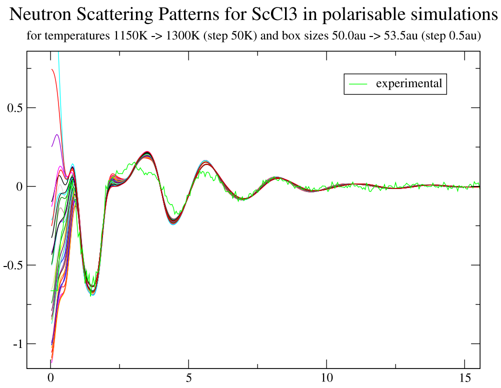

Figure 5.1: The Neutron Scattering Pattern of $ScCl_3$ with polarisability.

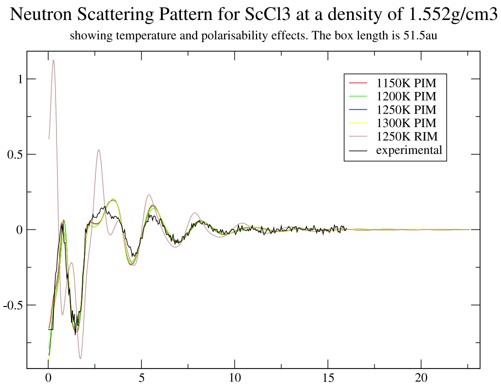

Figure 5.2: The Neutron Scattering Pattern of $ScCl_3$ at one density.

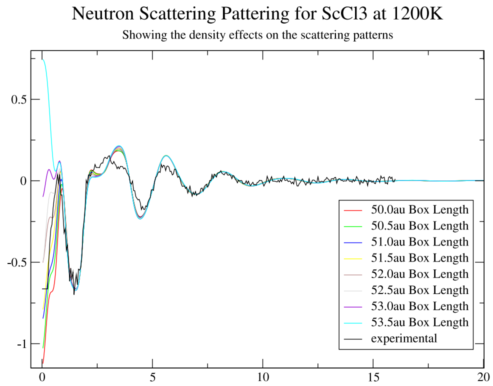

Figure 5.3: The Neutron Scattering Pattern of $ScCl_3$ for one temperature.

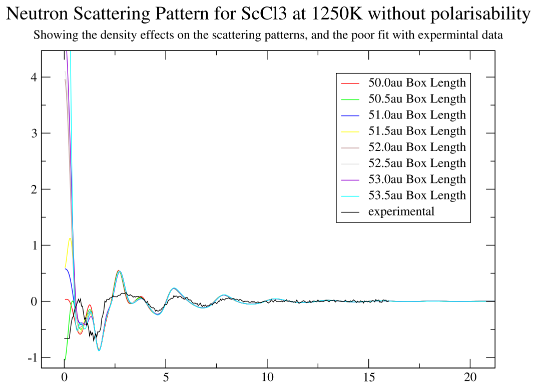

Figure 5.4: The Neutron Scattering Pattern of $ScCl_3$ without polarisability.

Also looking at figure 5.1, there are a few features of interest that should be pointed out. The first is the pre-peak, at a k value of about 1Å. This corresponds to intermediate range ordering in the system, which is somewhat unusual in molten salts. On looking at the simulation cell, we can see long chains of $ScCl_3$, with gaps between them. The pre-peak is caused by this intermediate range ordering [27]. What is also interesting is that as we've moved into a more complex system from the SMS, we're seeing features like this. It will be interesting to see if a scaled parameter model for an MX_3 system can easily predict such a thing, when the theory is extended to account for more complex stoichiometries than MX. However, it is expected that a system using scaled parameters will not get the intermediate range part correct, though it is thought that it probably will correctly model the long range components.

Figure 5.2 shows us two things, the effect of temperature and the effect of polarisability. Taking the temperature first, we see a remarkably small effect. The exact location and height of the pre-peak shows very slight shifts, indicating that the increased molecular motion with temperature isn't having much of an effect on the long range ordering. We can also see that none of the temperatures really takes the pattern any closer to the experimental pattern. Looking at the effect of polarisability shows a much more dramatic alteration. The rigid runs produce scattering patterns that are clearly very wrong. The period of the fluctuations is somewhat off, and the amplitudes are very very wrong. From this we can see that rigid runs give very poor structural agreement on complex systems.

Moving on to figure 5.3, we can look in detail at the density effects. The pre-peak is the main change, with the position and height of the maxima changing density. The second peak also shows some change - the lower densities have a larger saddle point in the main peak, while the higher density ones have this feature smaller but the main peak larger. The other peaks are largely the same.

Finally, we look at figure 5.4, showing the effect of density on the rigid runs. We can again see how poor a fit the rigid runs give for the neutron pattern, and hence the structure in general. The period of large k oscillations are wrong. At short k, the amplitudes are wrong, especially in the second peak. Finally, the pre-peak is very wrong, indicating that the rigid system is unable to generate the intermediate range ordering. Looking specifically at the density effects, we see very little. About the only noticable effects come in the pre-peak, and the exact positioning of the second and third peaks.

In general, we can see that the rigid runs very poor at getting structure correct. As such, polarisable runs must be done when trying to study systems of this complexity.

While the current potential for the polarisable system is close, it could still use some more work to further improve its fit to the experimental structure. One possible way to achieve this is via reverse Monte-Carlo fitting [28, 29].

5.1.2 Pressure

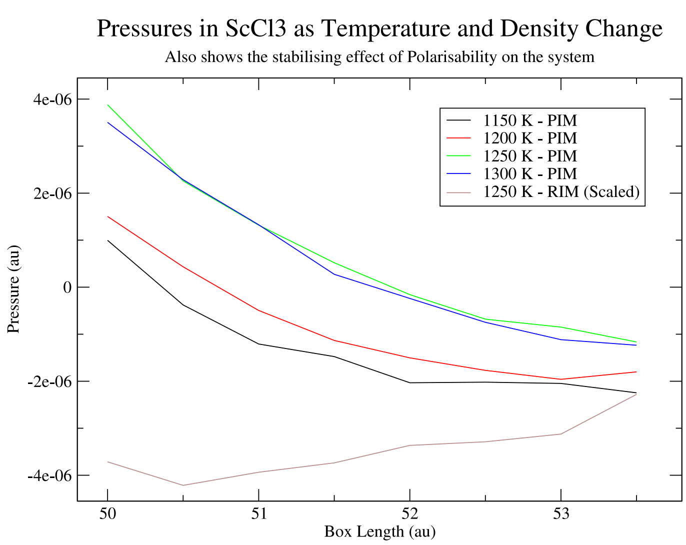

Figure 5.5: The Pressure of $ScCl_3$ as the density is altered, for various temperatures. The RIM curve is scaled by a factor of 10

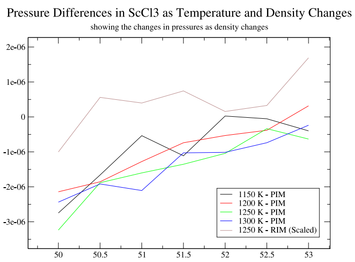

Figure 5.6: The Pressure Difference of $ScCl_3$ as the density is altered, for various temperatures. The RIM curve is scaled by a factor of 0.1

It is clear from looking at figure 5.5 that there is a complex relationship between the density, the temperature and the pressure. In order to be able to properly quantify this relationship, we would need significantly more results. These should be both over a wider range of state points, and with finer grained changes (such as box size changes of 0.25 au, temperature changes of 25k etc). Without this extra data, we can't say much about the effects on pressure.

However, one thing we can easily note is the effect of polarisation on increasing the system's stability. This effect is very much greater than was seen with the theoretical system in section 4.2, owing to the greater complexity of the system.

The rigid system shows the opposite trend with density to the polarisable ones. This is indicative that a rigid simulation just doesn't model the system properly, as the effect of polarisability is crucial in modeling the thermodynamics of a $ScCl_3$ system.

We should also note that the pressure in the rigid system appears to be behaving unphysically. The fact that the pressure is increasing as the volume increases indicates that the system is mechanically unstable, since it has a negative compressability. This indicates that perhaps it is near the point of cavitation. It is also possible that the rigid system has an equilibrium density at 1250K that is greater than the polarisable runs, which would explain the minima at low box length (high density). Further runs with both smaller and larger box sizes would be needed to confirm or deny these ideas.

Looking at the pressure differentials in figure 5.6, we can see that there isn't a very large temperature effect in how the pressures alter with density. The average of the different temperatures shows this, but also highlights that the change with density isn't linear. Clearly, the behaviour of the system is much more complex than the theoretical model in chapter 3. As mentioned above, data from more temperatures and a wider range of state points would be needed to quantify this relationship.

5.1.3 Diffusion

Looking at figures 5.7 and 5.8, we see there is a temperature dependence on the diffusivities. As we only have data from a very limited number of temperatures, we can't really quantify this relationship with any certainty.

However, looking at the scaled diffusivities in figures 5.9 and 5.10, we see the scaling has removed the temperature dependence. The scaling has been done by dividing all values by the lowest density value. This was simply done for convenience, and scaling by any of the values will give a similar pattern.

In figures 5.9 and 5.10, we note that the average over all the temperatures are

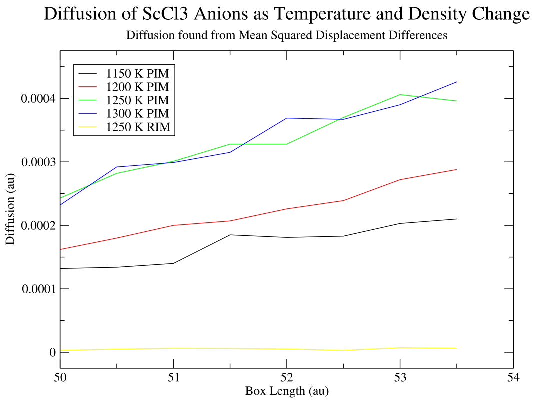

Figure 5.7: The Diffusion of the $ScCl_3$ Anion, showing temperature and density effects.

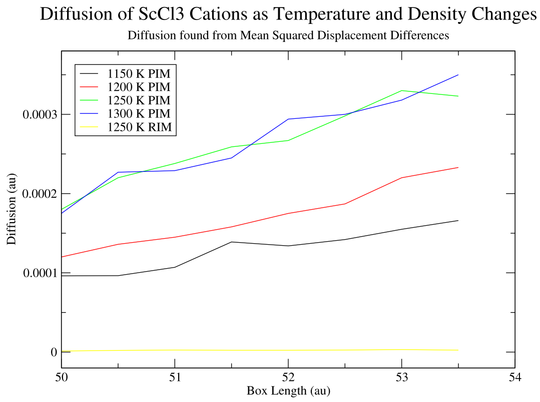

Figure 5.8: The Diffusion of the $ScCl_3$ Cation, showing temperature and density effects.

pretty much straight lines. This tells us that not only has the scaling removed the temperature dependence, but that it has left us with a simple relationship afterwards. It will be interesting to see how well this simple relationship holds as the range of temperatures studied is extended.

For the region of interest, we can devise some empirical relationships for how the scaled diffusivities vary:

$$\frac{\text{anion diffusion}}{\text{anion diffusion}_{(ref)}} = 1 + 0.20(\rho - \rho_{(ref)}) \tag{5.1}$$

$$\frac{\text{cation diffusion}}{\text{cation diffusion}_{(ref)}} = 1 + 0.25(\rho - \rho_{(ref)}) \tag{5.2}$$

From equations 5.1 and 5.2, we can see that the cations show a greater density effect. This is interesting, as the anions have higher diffusivities than the cations, as shown in figures 5.7 and 5.8. Another sign that the $ScCl_3$ system is more complex than the theoretical one is that we cannot make a prediction for the absolute value in the region of interest. Unlike equations 4.7 and 4.8, we can only list relative values, and not absolute ones. Also, as the rigid simulation is such a poor representation of the system, we can't come up with equations related to the RIM runs (such as

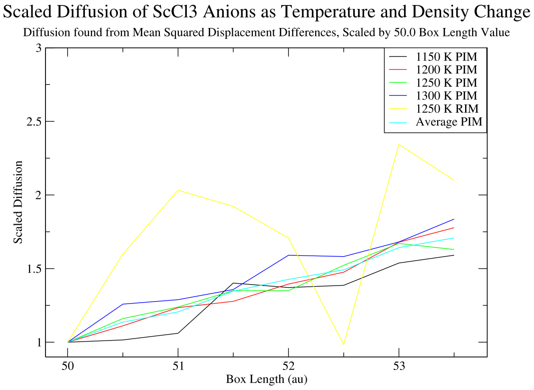

Figure 5.9: The Scaled Diffusion of the $ScCl_3$ Anion. Scaling has removed the temperature effect.

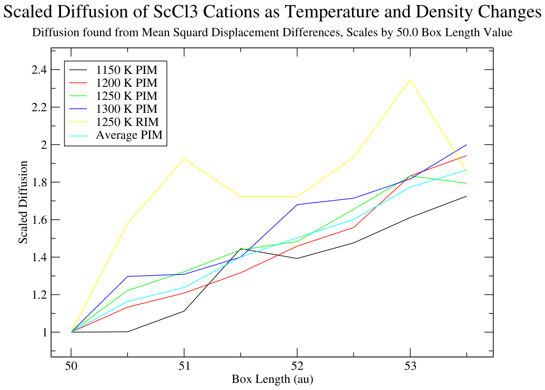

Figure 5.10: The Scaled Diffusion of the $ScCl_3$ Cation. Scaling has removed the temperature effect.

equations 4.4 and 4.5).

Clearly, the $ScCl_3$ system is much more complex than the theoretical model. Tricks such as scaling are required to simplify the system sufficiently to allow us to produce empirical relationships. However, the fact that such a trick is sucessful gives hope that a general relationship might be possible. With a wider range of densities and temperatures, the above relationships can be extended. Once theoretical backing can be found for the equations given in sections 4.4 and 4.5, perhaps it can be extended to produce proper relationships for the diffusivity in $ScCl_3$. For now, we will have to make do with MD supplied empirical relationships for the diffusion.

5.1.4 Viscosity

Looking at figure 5.11, we see that the viscosity shows a temperature dependence, and is somewhat noisy, despite the fairly long run length and the averaging. Using a similar trick to with the diffusion (section 5.1.3), we can scale the values by the lowest density value, as seen in figure 5.12. This scaling removes the temperature dependence in the viscosity, for the region of investigation, but makes the errors in the values more obvious. By averaging over all the temperatures, we simplify the picture, by averaging out much of the noise.

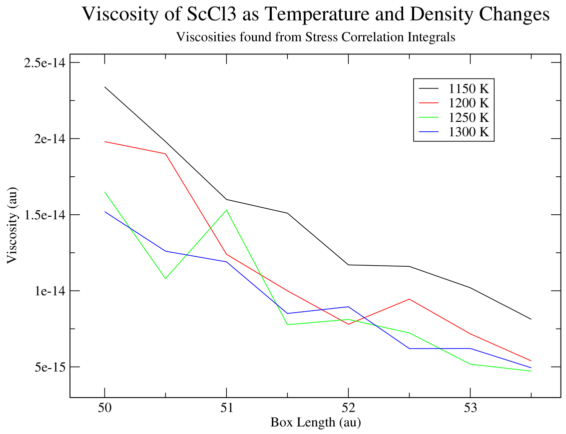

Figure 5.11: The Viscosity of $ScCl_3$, showing temperature and density effects, but without a simple temperature dependence being present.

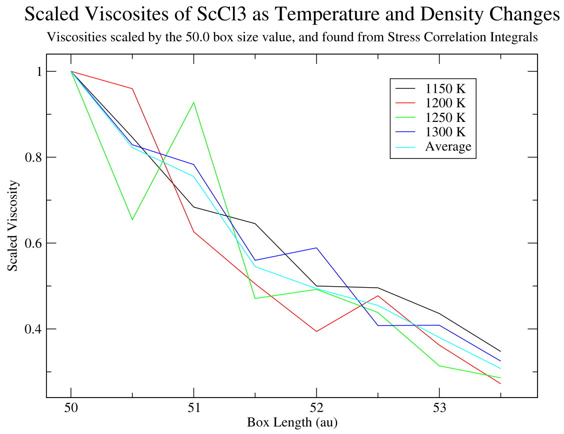

Figure 5.12: The Viscosities of $ScCl_3$ scaled by the 50.0 box size value for each temperature, showing the density effect. The scaling has removed any temperature dependency.

Looking at the average in the scaled graph, we see an almost straight line. From this, it is possible to come up with an empirical relationship for the viscosity of $ScCl_3$, for the region of investigation:

$$viscosity(au) = 1 - \frac{0.19(\rho - \rho_{(ref)})}{viscosity_{(ref)}} \tag{5.3}$$

Any value from the region can be used as the reference point. Data from many more temperatures will be required to quantify the temperature dependence, and hence come up with a general formula including density and temperature for the region of interest.

For the rigid runs, it was not possible to calculate the viscosities. The stress correlation functions did not converge in the timescale of the simulation, unlike with the polarisable runs. This indicates that transport properties in the rigid system are occuring on a different (longer) timescale to the polarisable system. Clearly, polarisability has a big role to play in facilitating the dynamics, possibly due to stabilising effects on the intermediate states, or lowering activation energies. Once more, we see an example of how the rigid system differs markedly from the polarisable system in $ScCl_3$, where only a minor alteration was seen with the theoretical model.

5.2 $ScCl_3$ vs Theoretical Model

We can see that the two systems are quite different in terms of complexity. With the PSMS, the rigid ion runs differed from the polarisable ones, but gave results in the same ball park. With $ScCl_3$, the rigid runs were very wrong, giving answers that were very different. The empirical relationships from one system is unsurprisingly untransferable to the other. Without more data, it is not possible to come up with more general relationships (ideally involving reduced variables) to test how well they might match up.

The MD method was able to easily deal with both systems. However, the theoretical model is currently completely incapable of handling an MX_3 system. That said it has recently been extended to handle MX_2 systems (including $ZnCl_2$), so it shouldn't be too long until it is able to handle MX_3 systems like $ScCl_3$. While the MD runs take longer than the theory calculations, the MD runs are currently able to handle a wider range of systems.

As was seen in section 1.4, MD and theory currently complement each other. For some types of systems, only one will work, but in an increasing number of areas, both will work, and yield data from different areas.

In the theoretical model, the RIM came quite close to representing the dynamic properties of the system. In chapter 4, we saw quite a number of equations which quantified how far off the RIM was from the correct value. For some applications, these may be small enough for it to be of use.

With $ScCl_3$, the RIM was a long way off. In general, it was so far off as to be of no use. Section 5.1.1 showed that the structure was completely off for the rigid runs. In section 5.1.2, we saw that the thermodynamics and stability of the system were very wrong. Section 5.1.3 showed that the diffusion coefficients were not only wrong, but showed the wrong trends with density to boot. Finally, in section 5.1.4, we discovered that the stress correlation functions didn't converge in the timescale of investigation, indicating the system is behaving on different dynamic timescales.

Clearly, $ScCl_3$ is much more complex than the theoretical model. Given that the theoretical model was designed to be simple, to improve calculation speeds, this shouldn't be a suprise.

Chapter 6

Conclusions

6.1 Run Lengths and Averaging

We have seen that averaging runs can be a powerful technique. It can lead to more accurate results for a given expenditure in computer time than extending run lengths does. For groups with access to clusters, it can allow for efficient computing resource usage, yielding high accuracy results with shorter length runs. It also offers an easy way to take advantage of the power of clusters, which are now more common. It is hoped that other simulational chemists will be able to learn from this work in the areas of cluster use and averaging.

With care, the cluster specific code (MPI or similar) can be abstracted away from the simulation code. This approach allows those with less parallel programming knowledge to take advantage of clustering, and also increases the applicability of any clustering logic to other simulation code.

When investigating averages, we must be careful about repeatable values vs long time values. The best way to avoid the pitfall is to ensure we test values out to longer run lengths than we intend to use, to check the value has stabilisied.

6.2 System Size Effects

While computational resources will generally restrict that largest simulation size we can use, we should still investigate what effect this is having on our results.

For our studies, the system size effects were slight, but not always insignificant (eg $ScCl_3$ viscosity). In an ideal world, we would've used larger simulation sizes, but computing resources prevented us from doing so.

6.3 Theoretical Model Work

With this, molecular dynamics met theoretical models. We saw that MD simulations compelment theories - both have their strengths and weaknesses. MD simulations are good for dynamic properties and thermodynamic properties, while the theory calculations can handle much larger system sizes, and are generally faster.

In our search for a reduced parameter to express dynamic properties, ϵ_∞ appears to be better than $\epsilon(0)$, but it still isn't perfect.

We came up with a number of empirical formulae in chapter 4. Several of these could be particularly helpful in assisting people to decide on the effects of polarisation to their system. Armed with them, it is possible to quantify the effect of polarisaiton, and hence decide if a rigid run will be accurate enough, or if a polarisable run is really required.

6.4 $ScCl_3$ Work

$ScCl_3$ is a much more complex a model than than the theoretical one in chapter 4.

The rigid runs were much worse than than they were for the theoretical system. Unlike the theoretical system, the rigid runs don't properly model the system, and so the results from rigid runs are typically very wrong. As such, we need to include polarisability in this, and similar systems. The RIM appears to be only suitable for a very narrow range of systems.

The Pressure, Neutron Scattering Pattern, Diffusion and Viscosity are much more complex than in the theoretical model. We couldn't make as many empirical predictions as for the theoretical model, and we had to use reference values much more.

Bibliography

[1] Gray-Weale, A., Madden, P.A., *Mol. Phys.* (in press)

[2] Madden, P.A., Wilson, M., Hutchinson, F., *Mol. Phys.* **99**, 10, 811 (2001)

[3] Well, D., Okido, M., *Reports Progr. Phys.* **5**, 21 (1997)

[4] Matsuura, H., Takagi, R., Zbalocka-Malicka, M., Rycerz, L., Szczepaniak, W., *Nucl. Sci. Technol.* **33**, 895 (1997)

[5] Born,M., Von Karman, T., *Physik. Z.* **13**, 297, (1912)

[6] Metropolis, N. Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., Teller, E., *J. Chem. Phys.* **21**, 1087, (1953)

[7] Ewald, P., *Ann. Phys.* **64**, 253, (1921)

[8] Wilson, M., Madden, P.A., *J. Phys. Condens. Matter.* **5**, 2687, (1993)

[9] Tang, K.T., Toennies, J., *J. Chem Phys.* **80**, 3726, (1984)

[10] Wilson, M., Madden, P.A., *J. Phys. Condens. Matter.* **6**, 159, (1994)

[11] Wilson, M., Madden, P.A., *J. Phys. Condens. Matter.* **5**, 6833, (1993)

[12] Morgan, B., *Ionic Mobilities in Molten Salts*, Masters Thesis, University of Oxford (2001)

[13] Tanner, G., *Simulation Studies of the Superionic Conductor* Li_3N, Masters Thesis, University of Oxford (2002)

[14] Addison, C., *Interfaces in Molten Salts*, Masters Thesis, University of Oxford (2002)

[15] Hansen, J.P., McDonald, I.R., *Phys. Rev* A **11**, 2111 (1975)

[16] Weingärtner, H., *Pure Appl. Chem.* **73** (11), 1733 (2001)

[17] Hansen, J.P., McDonald, I.R. *Theory of Simple Liquids (*2^{nd} *Edn.)* p22 (1986)

[18] MPI Forum, http://www.mpi-forum.org/

[19] MPICH, http://www-unix.mcs.anl.gov/mpi/mpich/

[20] OpenMosix, http://openmosix.sourceforge.net/

[21] Bar, M., Cozzini, S., Davini, M., Marmodoro, A., *openMosix vs. Beowulf: a case study*, http://www.democritos.it/activities/IT-MC/openMosix_vs_Beowulf.pdf

[22] PVM, http://www.csm.ornl.gov/pvm/pvm_home.html

[23] Papdopoulos, P., Kohl, J.A., Geist, G.A., *PVM and MPI: a Comparison of Features*, http://www.csm.ornl.gov/pvm/PVMvsMPI.ps

[24] Oxford Supercomputer Centre, http://www.osc.ox.ac.uk/

[25] Bresme, F., Alejandre, J., *J.Chem. Phys.* **118**, 9, 4134 (2003)

[26] Wasse, J.C., Salmon, P.S., *J.Phys. Condens. Matter.* **11**, 2171-2177 (1999)

[27] Salmon, P.S., *Proc. R. Soc.*, **A 445**, 351 (1994)

[28] McGreevy, R.L., Pustai, L., *Mol. Simul.* **1**, 359 (1988)

[29] Keen, D.A., McGreevy, R.L., *Nature* **344**, 423 (1990)

Appendix A

Files and Calls for MPI Helper

This section should talk you through how to use and extend the MPI helper routines for the Madden group code.

A.1 Include Files

Below is a list of all the fortran include files, what they bring in, and why you might wish to use them.

A.1.1 build-paths.inc

Everything you need to go mangle the path settings.

Normally only used by *paths-helper.f* (A.2.3)

A.1.2 mpi-tweaks.inc

Stores the tweaks to variables due to the MPI environment.

You add to these variables when you read in your MPI settings (eg read in "Temperature +5 per thread"), and the scaling by thread number is handled for you. You then need to add the onto the main values, at the point where you read those in.

Normally only needed by readin routines (MPI and normal)

Details of adding new variables are covered in section A.3

A.1.3 use-mpi.inc

All you need to do "real" MPI calls, as well as to play with the MPI related settings.

If you just care about things like the thread number, there are calls you can make to find out, without needing to pull this in.

Used by anything that interacts directly with MPI calls, and anything that gets quite close.

A.1.4 uses-files.inc

All you need to work with file names

Allows you to get a path added onto a filename, but without the ability to go play with pathnames (which is where *build-paths.inc* (A.1.1) comes in).

A.2 Helper Routines

A.2.1 mpireadin.f

Reads in the file *for-mpi.inpt*. Initially uses this to decide if the MPI environment should be set up or not.

Then, uses the values in the file to set up the *mpi-tweaks.inc* (A.1.2) variables, based on the values in *for-mpi.inpt* and the thread number. These variables control the tweaks to things like the temperature, rcut, the potential etc.

To add support for changing a variable across threads, see section A.3

A.2.2 mpi-helper.f

Setup_MPI

Does all the MPI setup for you, fills up the handy variables etc.

Should only be called if *usingmpi* is true and MPI was compiled in...

This should be your first call

Shutdown_MPI

Shuts down the MPI environment.

However, it *does not* notify the other threads, so you need to do that some other way firsr (eg MPIAwaitFinish).

Abend_MPI

Does it's best to send out an "I'm gone" message to thread zero, then shuts down the MPI environment.

Normally only used after an unrecoverable error somewhere.

MPISignalCheck

Performs some Voodo magik to decide if your current run is small enough to run with the fast asyncronised calling, or if you'll need syncronised calls. Sets up any variables as needed for you, to be used by MPIPingPong.

This is needed because of bugs in the MPI environment. If you hit issues with threads stalling and giving you grief about closed nodes, try reducing the threshold in here.

MPIAwaitFinish

For non thread zero, sends out an "I'm finished" message and returns.

For thread zero, it waits for all the "I'm finished" messages to come in (via calls to MPIPingPong), and waits until they're all done. Also prints out periodic "Still Waiting" messages.

MPIPingPong (some_int)

Sends a status message (the argument) between the threads.

This could be send syncronously or asyncronously (see MPISignalCheck), back to thread zero. Thread zero collates these and prints them out, and also checks for other status messages.

SubFetchThread (some_int)

Pops the Thread number into the supplied integer

FetchThread

Returns the Thread number as an Integer

MPIVarCheck (some_double)

Checks to see if that double is unique across all the threads, and returns this fact as a logical.

Thread zero always gets back *False*, as it doesn't need to do anything. If two threads clash on a value, one will get *True* (a clash, and it needs to do something) and one will get *False* (no clash, and doesn't need to do anything), depending on the order they report in.

Handy for checking if random seeds differ etc, and would also be a good starting point for any more complex cross-thread variable checking routine.

A.2.3 paths-helper.f

addpath ('type' , 'path')

Adds the specified path to the paths list, with to be accessed when people ask for that specific type.

bf ('type', 'filename')

Locates the path for that type, then glues it onto the filename for you, and returns the combined lot.

Replace *file='foo'* with *file=bf('foos type','foo')* in *open, inquire* etc to use.

A.3 Adding new MPI "tweak" variables

Start off adding the variable / variable array to the *mpi-tweaks.inc* (A.1.2) file. Firstly, add the variable name to the "Friendly Interface" section, towards the bottom. This will define the variable.

Next, go down to the equivalence block at the bottom. Equivalence in the new variable (/ array) to the next spare space(s) in the mpi block of the correct type.

Finally in *mpi-tweaks.inc*, increase the size of mpi arrays if needed (they're defined as parameters towards the top).

With all that done, your variables will be zeroed and scaled as required automatically for you (that happens on the arrays), and you'll be able to access them by a sensible name.

Now, you need to be able to read in the tweak values. You'll do this in *mpireadin.f* (A.2.1). During the readin section, you should add your readin at the end. Clone an existing block, so you have a *true* / *false* to activate it, and then read in any parameters as needed (into your newly defined variables).

Finally, go and edit whichever of *readin.f* or *setup.f* is appropriate. Find where the reading in of the main variable happens, and add on the mpi difference variable to it. The MPI setup and readin routines will set the difference variable to the required value, or ensure it's zero for non mpi runs.

A.4 Handy Utilities

A.4.1 *.pl

Can be found in /home/MrDaydream/burch/bin/

A.4.2 *.sh

Can be found in /home/MrDaydream/burch/bin/

A.4.3 *.x

Can be found in /home/MrDaydream/burch/anal_code/

A.4.4 makefile

Can be found in /home/MrDaydream/burch/mpi/bulk_code_mpi/

A.5 Handy Data Handling Utilities

A.5.1 average-few.pl

Given a start and end number, and a file name, averages the files from the output directories together, into the current directory.

A.5.2 average-several.pl

Given a series of file names, averages the files into a new file based on the common name, in the current directory.

A.5.3 collate-averages.pl

Given a file name and a number of runs, builds up a series of files for increasing numbers of averages. Normally used during run length and average testing.

A.5.4 do-*.sh

A series of files to automate the data collation and calculation. Each is specific, but there should be a template for most tasks here.

A.5.5 scale-file.pl

Scales the values of column 2 or columns 2+, based on the first value in each column.

A.5.6 two-line-merge.pl

For files that've been split over two lines and XMGrace throws a wobbly, sticks them back together

A.5.7 avgpres.x

Averages over the 3 pressure directions, spitting out a new file of the average pressure.

A.5.8 avgstr.x

Averages over the 5 stress tensors, spitting out a new file of the average stress.

A.5.9 find-pres-avgs.x

Scans a pressure average file, and reports the average pressure for the length of that run.

A.5.10 find-snn.x

Calculate the Snn (number-number structure factor)

A.5.11 find-szz.x

Calculate the Szz (charge-charge structure factor)

A.5.12 positions-to-pdb.x

Convert a positions.out file to the PDB (protein database) format, so it can be used with the PovChem to produce PovRay files. Important step in making pretty pictures...

A.6 Handy Run Related Utilities

A.6.1 check-loads.pl

"Walks" around the machines in the lab, and reports back their current load status, sorted by load. Requires SSH keys or typing lots of passwords.

A.6.2 check-nice.pl

"Walks" around the machines in the lab, and ensures that all your jobs are properly "niced". Requires SSH keys or typing lots of passwords.

A.6.3 find-machines.pl

"Walks" around the machines in the lab, finds the least used, and builds an MPI machines file of these. Requires SSH keys or typing lots of passwords. Really you ought to use your own machine for testing, or Oswald for runs, but this might be handy for any new clusters.

A.6.4 run-new-*.sh

Scripts to handle creating new directories, copying input files and startup positions etc. Each is specific to a type of run, but most are useful as templates.

A.6.5 summary.pl

Figures out stuff about a run, like how long it's been running, how long it's got left to go, when it last ran etc

A.7 Handy Programming Utilities

A.7.1 makefile

Has all sorts of magic in it, so that you can do *make MPI=yes* and it'll work the rest out for you. Allows you to have one code base with MPI and non MPI parts, and compile it selectivly.

A.7.2 check-RCS

Ensures that all fortran, include and tex files, as well as makefiles in the current directory are in RCS and checked in.

A.7.3 find-types.pl

Goes through a *common.inc* file, and figures out what variables haven't been explicitly definded, and spits out the appropriate lines.

Lets you get from *implicit real* to *implicit none*

A.7.4 varmake.pl

A wrapper for ifc, that catches "variable not defined" errors, as caused by swapping *implicit real* for *implicit none*. It counts up the uses of that variable, prints out the lines using it, and finally gives you the explicit define blocks for them.