CONCURRENT COMPUTATION AND ITS APPLICATION TO THE STUDY OF MELTING IN TWO DIMENSIONS

Thesis by

Mark Alan Johnson

In Partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy

California Institute of Technology

Pasadena, California

1986

(Submitted May 9, 1986)

ACKNOWLEDGEMENTS

I thank my advisor, Dr. Geoffrey Fox, for his encouragement and guidance as I pursued research that combined concurrent computation and theoretical physics. I also thank Dr. Michael Cross for the guidance that he provided in my investigation of two-dimensional melting. Although many people have contributed to the stimulating atmosphere of the Concurrent Computation Project, I especially appreciate the discussions I have had with Steve Otto and John Salmon. Throughout my stay at Caltech my parents have been a constant source of support and encouragement. I especially thank my wife, Kathy, for encouraging me and patiently enduring the many nights that I have spent working and my daughter, Sarah, for being such a welcome distraction and a source of joy.

I gratefully acknowledge the financial support that I have received from the National Science Foundation and from the Shell Companies Foundation.

ABSTRACT

We report an investigation of the melting transition in a two-dimensional system of particles that interact through the Lennard-Jones potential. In our investigation we attempt to determine whether the melting transition consists of a conventional first-order transition or a pair of higher-order transitions separated by a region of hexatic phase. We study systems containing 1024 and 4096 particles using Monte Carlo simulations, which we ran on Caltech's concurrent processor. Since the concurrent Monte Carlo algorithm differs significantly from previous applications for the concurrent processor, we also explore various issues of concurrent computation. In particular, the processors must run completely asynchronously in order for the algorithm to be efficient, leading to problems satisfying detailed balance.

We investigated the melting transition along the $T^* = .7$ isotherm using constant-density simulations to measure potential energy, pressure, elastic constants, topological defects, and angular correlations. Using thermodynamic integration, we calculated the free energies of the solid and fluid phases, which we used to locate the melting transition. Both constant-density and constant-pressure simulations of the transition region confirmed the predictions of the free energy analysis. We initialized a sequence of constant-pressure simulations with a configuration from a constant-density simulation of the transition region. Using this technique, we were able to establish upper and lower bounds on the melting pressure and thereby estimate the width of the transition.

The sharpness of the melting transition and the consistency of our various simulations give strong support to the interpretation that the melting transition is first-order. Measurements of the elastic constants and the angular correlation function provided evidence that the Kosterlitz-Thouless mechanism does not correctly describe the melting transition. Thus, we conclude that melting in the two-dimensional Lennard-Jones system at $T^* = .7$ is a first-order transition. Since our simulations of the 1024-particle system exhibited strong finite-size effects in the transition region, we believe that finite-size effects dominated most previous simulations of the transition region.

CONTENTS

LIST OF ILLUSTRATIONS

FIGURE PAGE

LIST OF TABLES

Chapter 1: Introduction

1.1 Preamble

Although the subject of ordering in two-dimensional systems has been studied over a period of many years, several recent advances have renewed interest in the subject. Experimental advances in the area of surface physics have increased theoretical interest in the effects of the reduced number of dimensions in such systems. Theoretical predictions have postulated the existence of a new phase and a phase diagram that is qualitatively different from that of a three-dimensional system.

With the availability of powerful computers, computer simulations have been used to study the properties of two-dimensional systems. Such simulations have several advantages over experimental systems since their results are not obscured by such effects as substrate interactions and second-layer promotion that occur in experimental systems. However, computer simulations generally require large amounts of computer power to obtain good results, especially near phase transitions. Since the lack of sufficient computer power is usually the limiting factor in many types of simulation studies, faster computers would expand the utility of simulations in many fields.

The speed of the fastest sequential computers is unlikely to increase dramatically in the future, so achieving large increments in available computer power requires a different approach to computing. An obvious method of building a faster computer is to connect several sequential computers together to form a concurrent processor. Advances in VLSI technology have produced microprocessors that are very cost-effective and have made the construction of concurrent processors with many processing elements practical. At Caltech we have built several concurrent processors and are learning about many issues involved in concurrent processing by implementing substantial research applications on the machines.

Excited by the promise of large amounts of computer power, we have undertaken a project which combines research in physics and concurrent computation by developing a concurrent algorithm for simulating a two-dimensional system. The implementation of the concurrent algorithm raised several challenging issues dealing with concurrent computation and the results of the simulation provided greater insight into the melting transition in the two-dimensional system.

1.2 Theoretical Background

In 1935 Peierls,[1] and later Landau,[2] argued that long-range order does not exist in the traditional sense in two-dimensional systems. The thermal motion of the atoms breaks the perfect translational symmetry of a crystal at nonzero temperatures in any number of dimensions. However, in a two-dimensional crystal the equilibrium positions of the atoms become uncorrelated due to long wavelength thermal phonons. In a three-dimensional crystal the mean-squared deviation of an atom from its equilibrium position is always finite, regardless of the size of the crystal. In contrast, the mean-squared deviation increases logarithmically with the system size in a two-dimensional crystal. In 1967 Mermin[3] used rigorous inequalities to arrive at the same result. Although the two-dimensional system lacks long-range translational order, Mermin showed that the orientation of the crystal axes around two points remains correlated, regardless of the separation of the two points. Thus, the two-dimensional system possesses long-range orientational order.

In 1972 Kosterlitz and Thouless[4] proposed a new definition of long-range order, which they call *topological long-range order*. Rather than refer to the two-point correlation function, which vanishes at any finite temperature in the two-dimensional system, they base their definition on the overall properties of the system. Possibly the most obvious way to characterize a solid is that it should have an elastic response to a small applied shearing stress. Thus, they propose that the disappearance of long-range topological order be associated with the transition from an elastic response to a fluid response to an applied shearing stress.

Kosterlitz and Thouless propose a general mechanism for melting in a two-dimensional system. At any finite temperature topological defects, such as dislocations in the two-dimensional solid, are thermally excited in the system. However, the dislocations experience an attractive potential that grows logarithmically with their separation, which tightly binds them into pairs with equal and opposite Burgers vectors. The potential energy of an isolated dislocation is a function of the logarithm of the system's size, so free dislocations cannot exist at low temperatures. Since the solid contains no free dislocations, it is rigid and has an elastic response to an applied shearing stress.

As the temperature of the system increases, the number of bound pairs of dislocations and the average length of their bonds also increase. Since the pairs of dislocations can orient themselves in response to an applied shear so that they reduce the shear, the shear modulus of the solid decreases with increasing temperature. Kosterlitz and Thouless show that both the potential energy of a free dislocation and its entropy vary with the logarithm of the system's size. Thus, at low temperatures the potential energy of a dislocation is more important than its entropy in the free energy of the system, so the probability of a free dislocation's existing in a large system is extremely small. However, at high temperatures the entropy of the dislocation becomes more important than its potential energy in the free energy, so the system should contain many free dislocations. At some critical temperature the contributions of the energy and entropy to the free energy exactly balance and the system becomes unstable to the formation of free dislocations. The criterion that expresses the limit of stability is

$$K \equiv \frac{4a^2}{kT_m} \frac{\mu(\mu+\lambda)}{2\mu+\lambda} = 16\pi \tag{1.1}$$

where we use the form given by Abraham,[5] which differs slightly from that given by Kosterlitz and Thouless. In (1.1) T_m is the instability temperature, a is the lattice constant, and μ and λ are the two Lamé elastic coefficients.[6] The value K is known as the Kosterlitz-Thouless K parameter, which has a discontinuity of 16π at the instability temperature.

Kosterlitz and Thouless point out that their instability criterion is only an upper limit of the stability of the solid against the formation of free dislocations. In arriving at their criterion, they considered the dislocation pairs in the solid to be independent. However, the field of one dislocation reduces the Lamé coefficients of the solid, thereby reducing the attractive potential that binds other dislocation pairs. Since the dislocation pairs interact in such a way as to reduce their binding potential, Kosterlitz and Thouless argue that the critical temperature could be renormalized downward by the interaction of the pairs. They predict nothing about the order of the phase transition at T_m; they only note that since free dislocations appear above T_m, the resulting fluid state will not support a shearing stress.

In 1978 Halperin and Nelson[7,8] reported calculations in which they applied renormalization group methods to the model of two-dimensional melting proposed by Kosterlitz and Thouless. Young[9] reported further study of the system with renormalization group techniques in 1979. They showed that the presence of bound pairs of dislocations in the two-dimensional solid renormalizes the Lamé coefficients. Just below the melting temperature, T_m, Halperin and Nelson find that the renormalized Lamé coefficient μ_R has the form

$$\mu_R(T) = \mu_R(T_m^-)|1 + const\,(T_m - T)^{\bar{\nu}}| \tag{1.2}$$

where $\mu_R(T_m^-)$ is limit of μ_R as T_m is approached from below. They obtain a similar expression for the other renormalized Lamé coefficient, λ_R, and calculate the exponent,

$\bar{\nu} = 0.36963 \cdots$, which Young also obtained. In addition, they find a universal relationship involving the renormalized Lamé coefficients that has the same form as (1.1), which Kosterlitz and Thouless obtained from their energy-entropy argument. Thus, (1.1) gives T_m as the actual melting temperature when the renormalized values of the Lamé coefficients are used, as suggested by Kosterlitz and Thouless.

An important feature of melting through the unbinding of dislocation pairs is that an isotropic fluid does not result from such a phase transition. Instead, Halperin and Nelson found that a new type of liquid crystal results from the presence of free dislocations, which they proposed be called the *hexatic* phase. While the hexatic phase does not support a shearing stress, it does retain some of the long-range orientational order of the two-dimensional hexagonal crystal, which gives it a sixfold anisotropy. In addition, they found that the unbinding of dislocation pairs leads to a *higher-order* phase transition.

Phase transitions are characterized by the discontinuities that occur in the various free energy derivatives at the transition. When quantities such as density and potential energy, which are first derivatives of the free energy, are discontinuous at a phase transition, the transition is said to be *first-order*. For instance, the melting transition in a three-dimensional system, such as water, is first-order. The density of ice and water differ at the melting temperature, as do their internal energies, so these quantities are discontinuous across the phase boundary. Since the internal energies of the two phases are different, the transition exhibits a latent heat, which indicates that the specific heat of the system has a singularity at the transition. In addition, first-order transitions exhibit hysteresis, in which the undisturbed system continues in its current phase as it passes through the phase boundary. The system is then in a metastable state that becomes less stable as the system moves farther from the phase boundary, until it abruptly changes into the equilibrium phase.

Phase transitions that are not first-order are known as higher-order phase transitions, where the order of the phase transition refers to which derivatives of the free energy have discontinuities. Since higher-order transitions do not have discontinuities in the first derivatives of the free energy, quantities such as density and internal energy are continuous across the phase boundary. Thus, the transition does not have a latent heat and the system does not exhibit hysteresis when crossing the phase boundary. An interesting feature of a higher-order phase transition is the large fluctuations in quantities such as density at the phase boundary. For instance, when water is at its critical point, the distinction between a low density liquid and a high density gas disappears. Since no latent heat is present at the critical point, large fluctuations in density occur giving rise to the phenomenon known as *critical opalescence.* In addition, the two-point correlation function diverges at the phase boundary, meaning that all of the particles' positions are correlated in the system. Besides leading to strong finite-size effects in the small systems that can be studied by computer simulations, the diverging correlation length leads to a phenomenon known as *critical slowing down* of the simulation, which greatly reduces its rate of approach to equilibrium near the critical region.

The orientational order of the anisotropic hexatic phase results from disclinations in the crystal structure remaining bound together in pairs of opposite disclinicity after the pairs of dislocations unbind. Although the screening effect of the free dislocations reduces the potential energy of a disclination, the pairs remain bound by a logarithmic potential until the system reaches a higher temperature, T_i. At T_i a second higher-order transition occurs, in which the pairs of disclinations unbind and the system becomes an ordinary isotropic fluid.

Thus, Halperin and Nelson postulate the existence of a new phase, the hexatic phase, which separates the solid and fluid phases of the two-dimensional system. An algebraically decaying translational order and long-range orientational order characterize the solid phase, in which thermally excited disclinations are tightly bound into pairs. In a triangular lattice the pairs of disclinations form dislocations, which in turn are more loosely bound into pairs by a logarithmic potential. Since free dislocations do not exist in the solid, it responds elastically to an applied shear.

When the system reaches the temperature T_m, the pairs of dislocations unbind and the system becomes an anisotropic fluid, which retains the sixfold anisotropy of the hexagonal crystal and is known as the hexatic phase. An exponentially decaying translational order and an algebraically decaying orientational order characterize the hexatic phase. The disclinations remain bound into pairs in the hexatic phase, but the pairs are bound more weakly than in the solid since the free dislocations partially screen the binding potential.

At a higher temperature, T_i, the pairs of disclinations unbind and the system becomes an isotropic fluid. Both the translational and orientational order decay exponentially in the ordinary fluid.

The two phase transitions that occur during the melting of the two-dimensional system are higher-order phase transitions. Higher-order phase transitions differ first-order phase transitions by having continuous first derivatives of the free energy and thus lacking a latent heat. Near such transitions large fluctuations and a diverging correlation length occur, which cause critical slowing down in computer simulations due to the very long time scales of the physical system. However, Halperin and Nelson are careful to note that although their description of melting is stable and self-consistent, it is only a possibility. A first-order phase transition could occur in which the system would change directly into an isotropic fluid in a single step, as in three dimensions. Another possibility that they raise is that the system could melt through a single first-order transition in part of its phase diagram and through the two higher-order transitions, with the intervening hexatic phase, in another region of its phase diagram. For instance, Figure 1.1 illustrates a possible phase diagram in which a single first-order melting transition occurs above T_0, as indicated by the heavy line. In contrast, below T_0 a region of hexatic phase separates two higher-order transitions, which the lighter lines indicate.

1.3 Previous Computer Simulations

Attracted by the interesting predictions of Halperin and Nelson, many investigators have used computer simulations to study the phase structure of various two-dimensional systems. Computer simulations have several important advantages over experimental studies of such systems. One advantage is that the residual effects of the third dimension, which are always present in experimental studies, do not exist in computer simulations. Another advantage is that some measurements that are difficult or impossible to perform in an experimental system are easily obtained from simulations. In addition, the investigator specifies exactly the interactions of the particles in a simulation, so uncertainties in the interactions do not complicate the interpretation of the results.

However, computer simulations have many disadvantages, also. Due to the limitations on the amount of computer power that is available for a simulation study, only relatively small systems can be studied in detail. Besides the size of the system being simulated, the length of time corresponding to a simulation is very small because very small time steps must be taken to ensure accurate results. As a result, features with long characteristic times may be impossible to see in a simulation. Near a phase transition both limitations can become very serious. As we mentioned in the previous section, the correlation length diverges and critical slowing down occurs near a higher-order phase transition, making accurate simulations of such a transition very difficult. If the phase transition is instead first-order, the results can be obscured by the strong metastability of the system near the transition. When the characteristic time of the metastability is large, a simulation may not be able to distinguish between stable and metastable equilibrium.

Two basic methods are commonly used to perform computer simulations of many-body systems, such as the two-dimensional system of particles that we will be considering. The first method is called *molecular dynamics.* As its name implies, the method consists of solving for the motion of each of the particles in the system as it interacts with the other particles. First, each of the particles is assigned an initial position and velocity. The force on each of the particles is then evaluated by summing the effects of all the other particles. Next, the force on each particle is integrated over the time step of the simulation giving a new position and velocity for each particle. A molecular dynamics simulation repeats the update procedure many times in order to solve for the dynamical evolution of the system.

The time step chosen for the simulation must be small enough to give accurate results, yet large enough to allow the simulation to evolve the system through a adequate length of time. Generally, a time step is small enough if it is somewhat smaller than the smallest characteristic time in the system, which is typically the mean time between collisions. On the other hand, the total amount of simulation time should be larger than the largest relevant time scale in the system. In a solid such a time scale might be the time a sound wave takes to propagate across the system. However, near a phase transition relevant time scales might be much larger. Since the minimum characteristic time does not change appreciably near a phase transition, the simulation must run for many more time steps when it is near a phase transition. If the simulation does not run long enough, the system may not be in equilibrium when measurements are made, giving results that can be very misleading.

Unfortunately, both the size of the time step and the number of steps required for accurate results are difficult to determine. If two simulations use significantly different time steps and still give nearly identical results, both values are probably sufficiently small, so subsequent simulations can use the larger value without introducing error. The same approach can be used to determine how long a simulation must run for it to give accurate results. When the average values of several measurements stop changing during the simulation, we may assume that the system is in equilibrium and that the measurements are accurate. Of course, both a metastable equilibrium and a very slow approach to equilibrium can closely resemble a stable equilibrium, so the validity of assuming that a system is in equilibrium is often debatable.

The second method of simulating a system is the Monte Carlo method, which is so named because it uses random numbers in its update procedure. We will explain the Monte Carlo method in detail in the following chapter, so we give only a brief overview here. Rather than numerically integrating the equations of motion for each of the particles in the system as the molecular dynamics method does, the Monte Carlo method generates a sequence of configurations of the particles based on the probability of each configuration's occurring. Results from statistical mechanics assign a probability to each configuration, which depends on the potential energy of the configuration and the temperature at which the simulation is running. An important feature of the Monte Carlo method is that it has no concept of time, except in the sense that the system's approach to equilibrium defines the direction of time. Thus, the problem of determining an appropriate value of the time step does not arise. However, the problem of determining whether the system is in equilibrium still occurs in a Monte Carlo simulation.

Having examined briefly the methods used in the computer simulations, we now present a survey of some of the previous results that were obtained in computer simulations. Most of the results that we report are for systems whose particles interact with the Lennard-Jones potential, which is a respectable model of the interaction potential between atoms of the noble gasses. Although the Lennard-Jones potential is a short-range potential, it is often truncated at a fixed distance to reduce the amount of computation that the simulation requires. In the next chapter we will discuss methods of correcting for the effects of the truncated potential.

In 1962 Alder and Wainwright[10] reported one of the early simulations of a two-dimensional system. They performed a molecular dynamics simulation of 870 hard disks in a rectangular cell with periodic boundary conditions. Using various initial conditions, they ran several simulations at various densities and measured the pressure of the system. Since the results of different simulations at the same density were typically within 1% of each other, they concluded that the system was in equilibrium and that the measured values of pressure were accurate.

Their pressure data exhibit a *van der Waals loop* that connects the solid and fluid regimes of the system. Figure 1.2 illustrates a van der Waals loop, which is an anomaly in a plot of pressure versus density where the pressure of the system decreases as its density increases. While a van der Waals loop cannot occur in an infinite system, it can appear in finite systems since the small size of the system tends to stabilize the predominant phase. Thus, the pressure is lower near the solid regime than it would be in an infinite system and higher near the fluid regime. A van der Waals loop indicates the presence of a first-order phase transition and is due to the phase separation's not being complete in the two-phase region as it would be in an infinite system. Alder and Wainwright plot the trajectories of the particles, which show that the two-phase system contains regions of mobile particles that separate domains of localized particles. In addition, they observe fluctuations in pressure near the phase transition that are larger than in the solid or fluid regimes. Alder and Wainwright conclude that a melting transition does take place in the two-dimensional hard disk system and that the transition is first-order.

In 1974 Tsien and Valleau[11] reported the results of a Monte Carlo simulation of a two-dimensional Lennard-Jones system containing 121 particles. They truncated the Lennard-Jones potential at a distance of 3σ, where σ is the Lennard-Jones length parameter. Periodic boundary conditions were imposed on the system with a unit cell in the shape of a 60° rhombus, which conforms to the geometry of the triangular lattice of the solid phase. Pressure measurements made at various densities along four isotherms show the clearest evidence of phase transitions. Two separate regions of van der Waals loops identify the solid-liquid and liquid-gas phase transitions. Measurements of the two-point correlation function that show a distinct smearing of the peaks in the liquid region but not the solid region, support the interpretation of the pressure data. The disappearance of one of the van der Waals loops along the two highest temperature isotherms establishes the existence of a liquid-gas critical point and provides an estimate of its temperature and density. Tsien and Valleau conclude that the phase structure of the two-dimensional Lennard-Jones system is similar to that of common three-dimensional systems, which have a liquid-gas critical point and first-order transitions separating their solid, liquid, and gas phases.

In 1975 Toxvaerd[12] reported the results of a molecular dynamics simulation of a two-dimensional Lennard-Jones system containing 256 particles. He truncated the Lennard-Jones potential at a distance of 2.5σ and imposed periodic boundary conditions on the system. In order to minimize the amount of time that the simulation required to reach equilibrium, he initialized each run with a configuration that was in equilibrium at a slightly higher density. The pressure data along the $T^* = .526$ isotherm show a van der Waals loop that is not present in the $T^* = .651$ pressure isotherm, indicating that the liquid-gas critical point occurs between the two temperatures. The melting transition manifests itself as a van der Waals loop in the pressure data along both isotherms over a range of densities centered somewhat above $\rho^* = .80$. Thus, Toxvaerd's results are consistent with the existence of a liquid-gas critical point and conventional first-order phase transitions separating the phases.

A few years later, Toxvaerd[13] reported the results of a similar study in which he simulated the same 256-particle system at various densities along five isotherms ranging in temperature from $T^* = .65$ to $T^* = 1.3$. Using the method of thermodynamic integration, which we will discuss in Chapter 5, he calculates the free energies of the solid and fluid states, which he uses to locate the melting transition. In conclusion, he states that the system exhibits a first-order solid-fluid phase transition in the range of temperature between .65 and 1.3.

In 1979 Frenkel and McTague[14] reported the results of a molecular dynamics simulation involving 256 particles that interact with the Lennard-Jones potential. They studied the $\rho^* = .88$ isochore at temperatures between $T^* = .25$ and $T^* = 1.25$. Plots of both pressure and potential energy versus temperature show a change in slope at $T_1^* = .36$ and $T_2^* = .57$, which they tentatively identify as Halperin and Nelson's T_m and T_i, respectively. By traversing the range of temperature in both directions, they checked for hysteresis, which they found to be virtually absent. Although the region between T_1 and T_2 could be a two-phase region, they argue against the presence of a two-phase region since the region in question showed no hysteresis, which is characteristic of a two-phase region. In addition, the angular correlation length seems to diverge as it approaches T_2 from above, as predicted by Halperin and Nelson, and the system loses its resistance to shear above T_1. Thus, Frenkel and McTague tentatively identify a regime of hexatic phase bounded by T_1 and T_2.

In a 1980 paper on two-dimensional melting Toxvaerd[15] disputes the claim of Frenkel and McTague that their results are compatible with the Halperin-Nelson predictions. In support of his argument Toxvaerd shows that the molecular dynamics results are compatible with a first-order melting transition. According to Frenkel and McTague's interpretation, the system should be a hexagonal crystal below $T_1^* = .36$. However, they show in their Figure 1(b) that the pressure for $T^* \lesssim .36$ is negative. In an infinite system the equilibrium pressure is never negative since the system can separate into a solid-vapor coexistence, which cannot have a pressure less than that of a vacuum. However, the phase separation may not be able to occur in a finite system due to the stabilizing effects of the boundary conditions, so equilibrium pressures can be negative.

Repeating the simulation of Frenkel and McTague, Toxvaerd observes the breakup of the uniformly distributed solid below $T^* = .36$, supporting his claim that the system lies in a two-phase region. He then simulates the system at $T^* = .50$, which lies between their values of T_1 and T_2, and imposes periodic boundaries on a square box, which is incommensurate with the triangular lattice, to avoid favoring the crystal over a fluid. Plots of the particle positions show that their distribution is nonuniform with some particles forming triangular lattices. As the temperature decreases, the fraction of solid increases, as we would expect in a two-phase system. Toxvaerd concludes that the melting transition is first-order and that the sixfold anisotropy observed by Frenkel and McTague is due to the coexistence of the liquid and solid phases in the two-phase system.

Also in 1980 Abraham[16] reported a different type of Monte Carlo simulation. Instead of simulating a 256-particle Lennard-Jones system under conditions of constant temperature and constant density, he specified the temperature and pressure of the system during his simulations. In order to maintain a constant average pressure, the simulation allowed the volume of the system to expand and contract. A two-phase system is not stable in a constant-pressure simulation, so the melting transition should be abrupt if it is first-order. By simulating the $p^* = .05$ isobar in a temperature range from $T^* = 0$ to about $T^* = .53$, he shows that the system abruptly decreases in density when its temperature increases to $T^* = .48$, following a previously smooth decrease in density. The two-point correlation function and plots of the particle positions show a change from order to disorder when the density drops abruptly. He then decreases the temperature and shows that the system exhibits hysteresis as it passes back through the phase transition. After smoothly increasing in density, the system suddenly freezes at $T^* = .40$. From the discontinuity of the transition between solid and liquid phases, the evident latent heat, and the existence of hysteresis, Abraham concludes that the melting transition of the system is certainly first-order.

In a third paper in 1980 on the subject, van Swol, Woodcock, and Cape[17] reported the results of molecular dynamics simulations of both the Lennard-Jones and the soft-disk systems. In their 2688-particle Lennard-Jones system they investigated the $T^* = .8$ isotherm and the $\rho^* = .8$ isochore, which Frenkel and McTague studied. Along the $T^* = .8$ isotherm they observed transient metastability in both the expanded crystal and the compressed fluid near the phase transition. Two of the densities they studied apparently lie on the phase coexistence line of constant pressure in the two-phase region. The melting pressure that they obtained by direct simulation of the two-phase system is about 20% greater than that reported by Toxvaerd.[15] They note that the difference probably occurs because of the small size of his system or the method that he used to obtain the reference values in his free energy analysis. They find no evidence of a van der Waals loop in their data, but they clearly see hysteresis, which eliminates the possibility of a higher-order phase transition. Thus, they agree with Toxvaerd's conclusion that melting is first-order.

Along the $\rho^* = .8$ isochore, melting started at $T^* = .40$ and completed at $T^* = .55$. At $T^* = .40$ the system seems to be in a solid-vapor coexistence, but only small voids form and the pressure is negative since the phase separation is incomplete. Plots of particle trajectories at $T^* = .45$ show that some particles are localized in triangular lattices while others are mobile, providing evidence of a two-phase coexistence. Since the temperatures bounding the melting region correspond to Frenkel and McTague's T_1 and T_2, they support the conclusions of Toxvaerd[15] that the intermediate region is a conventional two-phase region, not the proposed hexatic phase.

In 1981 Barker, Henderson, and Abraham[18] presented results from both constant-density and constant-pressure Monte Carlo simulations of a two-dimensional Lennard-Jones system containing 256 particles. Using thermodynamic integration, they calculated the free energies of the solid and fluid phases of the system, from which they extracted the location of the phase boundaries of the system. They ran the majority of their simulations along four isotherms with reduced temperatures of .45, .55, .70, and 1.00. They calculated a fluid density of $\rho^* = .801$ at the transition when $T^* = .55$, which is in excellent agreement with the results of van Swol et al.[17] In addition to the Monte Carlo simulations, they used perturbation theory and cell theory to estimate the fluid and solid free energies, respectively, which agree closely with their Monte Carlo data.

After examining particles' trajectories, they conclude that $\rho^* = .84$ on the $T^* = .7$ isotherm is clearly in the two-phase region, as their free energy analysis predicts. The direct coexistence value of the melting pressure is $p^* = 2.50$, which is significantly lower than the value of $p^* = 2.85$ that the free energy analysis gave. They note that the difference is presumably due to finite-size effects that restrict the separation of the two phases and thereby lower the pressure in the simulation of the two-phase system. As in the simulations of van Swol et al.,[17] they obtain higher pressures and solid and fluid densities at the melting transition than Toxvaerd.[15] They also observe evidence of a van der Waals loop in the two-phase region, which provides further indication that the melting transition is first-order. They conclude that all of their results are consistent with first-order transitions separating the solid, liquid, and gas phases and with the existence of a liquid-gas critical point. They also suggest that the sixfold anisotropy observed by Frenkel and McTague is due to solid-liquid coexistence in the two-phase system and not to a hexatic phase.

In another paper Abraham[19] reported that his computer simulations satisfy the Kosterlitz-Thouless instability criterion for a two-dimensional solid but that the instability leads to a first-order transition. His simulation results show that at a constant pressure of $p^* = .05$ the system melts abruptly at a temperature of $T^* = .45$. After measuring the Lamé coefficients, he shows that the Kosterlitz-Thouless K parameter approaches the predicted value of 16π when extrapolated to the observed melting temperature. However, the free energy analysis gives a value of $T^* = .415$ for the melting temperature, in contrast to the observed temperature of $T^* = .45$. Thus, he argues that the system melts at the higher temperature where the metastable solid reaches an instability that allows a dilute concentration of free dislocations to form. The absence of a free surface, which results from the periodic boundary conditions, constrains the system to superheat and to become metastable. By simulating a strip of particles with two free boundaries, he shows that the equilibrium melting point is between $T^* = .40$ and $T^* = .42$. Thus, Abraham concludes that the system satisfies the instability criterion when it melts from a metastable state but that the resulting transition is first-order.

Also in 1981 Toxvaerd[20] reported additional simulations in which he investigated the effects of changing the number of particles in the system. The simulations are similar to his previous ones,[12,13,15] except that he simulated systems with 256 and 3600 particles. Both systems exhibit a van der Waals loop in their pressure data along the $T^* = 1.00$ isotherm. In the melting region the pressure measurements exhibit large fluctuations in the small system but much smaller fluctuations in the large system. The value of the melting pressure that he obtains from a free energy analysis is higher than that obtained by direct simulation of the two-phase region, but he notes that the cumulative error in the free energy analysis is large and the difference may not be significant.

Toxvaerd observed no evidence of significant finite-size effects in the fluid regime. However, the energy per particle in the solid regime was significantly larger in the larger system, as was the defect concentration. As the system approached the transition, the difference in both quantities increased. In addition to the expected clusters of four disclinations, i.e., a bound pair of dislocations, he observed clusters of defects in the larger system as it approached the transition. In contrast, the smaller system seemed to inhibit the formation of defects. After measuring the elastic constants, he found that while the Kosterlitz-Thouless K parameter was significantly greater than its predicted value of 16π

at the melting density that the free energy analysis gave, it was close to 16π at the observed melting density. Thus, Toxvaerd agrees with Abraham's conclusions that the Kosterlitz-Thouless instability criterion seems to be satisfied at the observed melting temperature and that the melting transition is first-order. However, he raises the possibility that the order of the melting transition itself might be a finite-size effect since defect formation was apparently inhibited in the smaller system.

In 1982 Tobochnik and Chester[21] reported the results of Monte Carlo simulations of a 1024-particle Lennard-Jones system. They investigated the system along three isochores, two at the relatively low densities of $\rho^* = .856$ and $\rho^* = .888$ and one at the higher density of $\rho^* = 1.143$. Although they measured many quantities, such as potential energy, pressure, two-point correlation, angular correlation, and elastic constants, and examined the topological defects and particle trajectories, they were unable to reach a firm conclusion about the order of the melting transition at the two lower densities. The problem is that most of the data do not have unambiguous interpretations that would clearly distinguish between a two-phase system and the predicted hexatic phase. At the highest density, they concluded that the melting transition is first-order since the results are less ambiguous.

The short summaries of previous computer simulations that we have provided in this section are meant to give a general impression of the research work that has been done; they certainly do not constitute an exhaustive review of all the results that have been published. However, we hope that they provide an overview of the techniques involved in computer simulations and illustrate some of the important issues that need to be resolved.

1.4 Overview of Research

From the simulations discussed in the previous section, we see that the data produced by computer simulations can often be interpreted in more than one way. In fact, no experimental measurement or computer simulation can determine absolutely the order of a phase transition since any singularities may be unobservably weak. However, carefully examining the sources of ambiguity may allow us to improve our simulation techniques and learn more about the system. An example of an ambiguous feature in the data is the hysteresis that several investigators reported observing as the system passed through its melting transition. As we mentioned in Section 1.2, hysteresis is characteristic of a first-order phase transition. However, distinguishing the hysteresis that is due to a first-order transition from the critical slowing down that a higher-order transition causes may be difficult, leading to an ambiguous interpretation of the simulation.

The discontinuity of such quantities as density and energy across the phase boundary illustrates another instance of the inherent ambiguity in many of the results that a simulation produces.. Since such discontinuities characterize a first-order transition, they seem to provide a useful way of determining the order of the transition. However, a higher-order transition may appear to be discontinuous if the simulation crosses the phase boundary so quickly that the system is not always in equilibrium. Unfortunately, critical slowing down can make determining whether the system is in equilibrium very difficult. On the other hand, a first-order transition would not really be discontinuous anyway, since such a transition is discontinuous only in the limit of an infinite system. In a system with a finite number of particles, an effect called *finite-size rounding* reduces the sharpness of the transition. Thus, if the number of particles in the system is too small, the transition may appear to be continuous and lack hysteresis. Depending on the circumstances, both continuous and discontinuous transitions in the system being simulated can lead to multiple interpretations.

The fundamental problem relating the various ambiguities in the results of computer simulations is the limited amount of available computer power, which manifests itself in two different ways. One way is that the system being simulated can contain only a relatively small number of particles. In the previous section we noted that the number of particles which have been used in simulations varies from a few hundred to a few thousand. With such a small number of particles compared to the number in a macroscopic system, finite-size effects may dominate the simulation results so that the simulation does not exhibit the features of the macroscopic system. A small number of particles usually give adequate results for a single phase system that is not near a phase boundary since the relevant length scales in the problem are on the order of the distance between particles. However, near a phase boundary, length scales that are much larger can occur since the two-point correlation function diverges at a higher-order transition. Even if the transition is first-order and the correlation function does not diverge, the boundary conditions can have a significant ordering effect on the system, especially if they accommodate exactly the triangular lattice of the solid.

The second way in which the limited availability of computer power manifests itself is that a simulation can represent only a relatively narrow range of time scales. The number of updates performed by the simulations discussed in the previous section was usually on the order of thousands. Thus, such simulations would not accurately represent time scales that are more than about a thousand times longer than the smallest time scale in the system, which is typically the mean time between collisions. However, near a phase transition the range of relevant time scales in the system can span many orders of magnitude. If the transition is higher-order and the simulation is experiencing critical slowing down, the system cannot approach the transition too closely and still remain in equilibrium.

If the transition is first-order, simulations must avoid known sources of metastability since the characteristic time scale of a metastable state may be extremely long. For instance, the solid formed by rapidly cooling a liquid contains many lattice defects and is not in its true equilibrium state since the free energy of a perfect crystal is lower. However, a large free energy barrier separates the metastable glass from the perfect crystal, so the characteristic time of the metastability may as well be infinite. Another example of metastability is the hysteresis encountered when passing through a first-order transition. In some cases carefully planning the simulation can avoid or reduce the effects of such metastabilities if their causes are understood.

In the research that follows, we will employ several methods of reducing the ambiguities in the results of our Monte Carlo simulations of the melting transition. First, we limit the scope of our investigation to a single isotherm so that we can concentrate all of the available computer power into a few carefully chosen simulations. Since the pressure in a two-phase region is constant along an isotherm, we choose to simulate the system along an isotherm where the behavior of the pressure should clearly identify any two-phase region. Of course, the pressure will be constant only if the system is large enough that finite-size effects are not significant. However, if the transition is first-order and finite-size effects are important, a van der Waals loop would still identify the two-phase region.

Another method of improving our results involves avoiding some of the metastabilities that have obscured previous results. We will use a method of simulation that has not been reported previously, which should greatly reduce the metastability near a first-order transition and allow very sensitive measurements. The method avoids much of the metastability in the transition region by using a configuration that is in equilibrium in the transition region to initialize a series of constant-pressure simulations. If the initial configuration consists of two coexisting phases, the system should become either all liquid or all solid depending on the simulation pressure since a two-phase configuration is not stable in a constant-pressure simulation. Several such simulations at different pressures should provide upper and lower bounds on the melting pressure and thereby give an indication of the sharpness of the transition.

The final method of improving the results of the simulation is the brute force method, increasing the available computer power. We describe the effort at Caltech to obtain large amounts of computer power through concurrent computation, in which many computing elements work together on a single problem. We will examine the concurrent Monte Carlo update algorithm in detail and carefully establish the correctness of the modified algorithm. The concurrent algorithm presents several interesting issues that had not been explored previously in our concurrent computation project.

After describing the two-dimensional system that we will be simulating and reviewing the Monte Carlo method in a preliminary chapter, we discuss the the concurrent Monte Carlo update algorithm and present a detailed analysis of its efficiency. We then report results of simulations along integration paths in the solid and fluid regions of the phase diagram. Applying the method of thermodynamic integration, we obtain the free energies of the two phases and calculate the parameters of the melting transition, which the analysis assumes to be first-order. In the following chapter we present results of simulations in the transition region, including constant-pressure simulations that we initialized with a configuration from the transition region. After deriving the formulas for the isothermal elastic constants in two dimensions, we present measurements of the elastic constants and calculate the Lamé coefficients and the Kosterlitz-Thouless K parameter. Next, we discuss the results of two additional measurements, the topological defects of the lattice and the angular correlation function. Finally, we summarize the most important results and draws several conclusions.

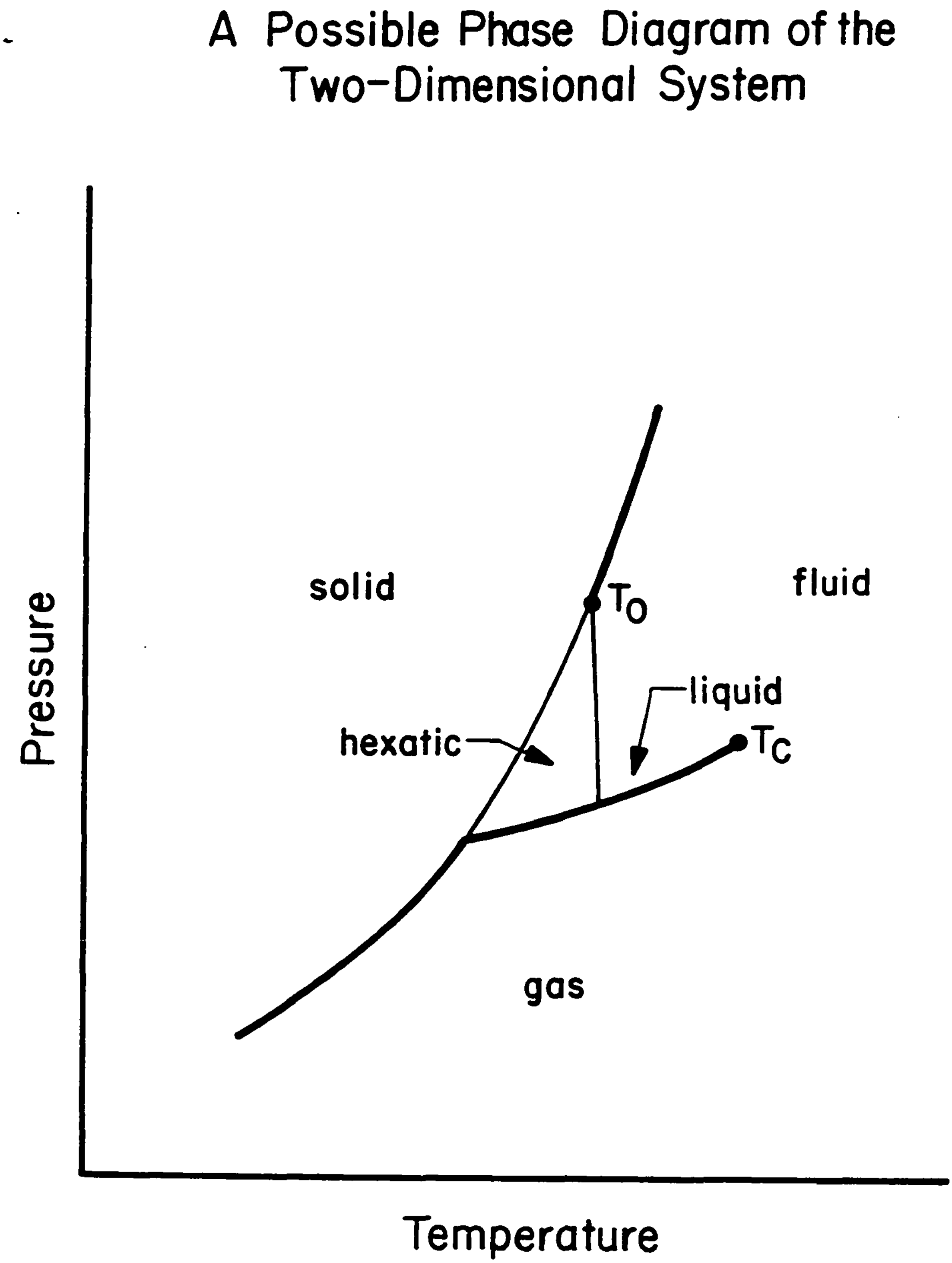


Figure 1.1 A possible phase diagram in which the heavy lines indicate first-order transitions and the lighter lines indicate higher-order transitions. Above T_0 a conventional melting transition occurs, but below T_0 a region of hexatic phase separates two higher-order transitions.

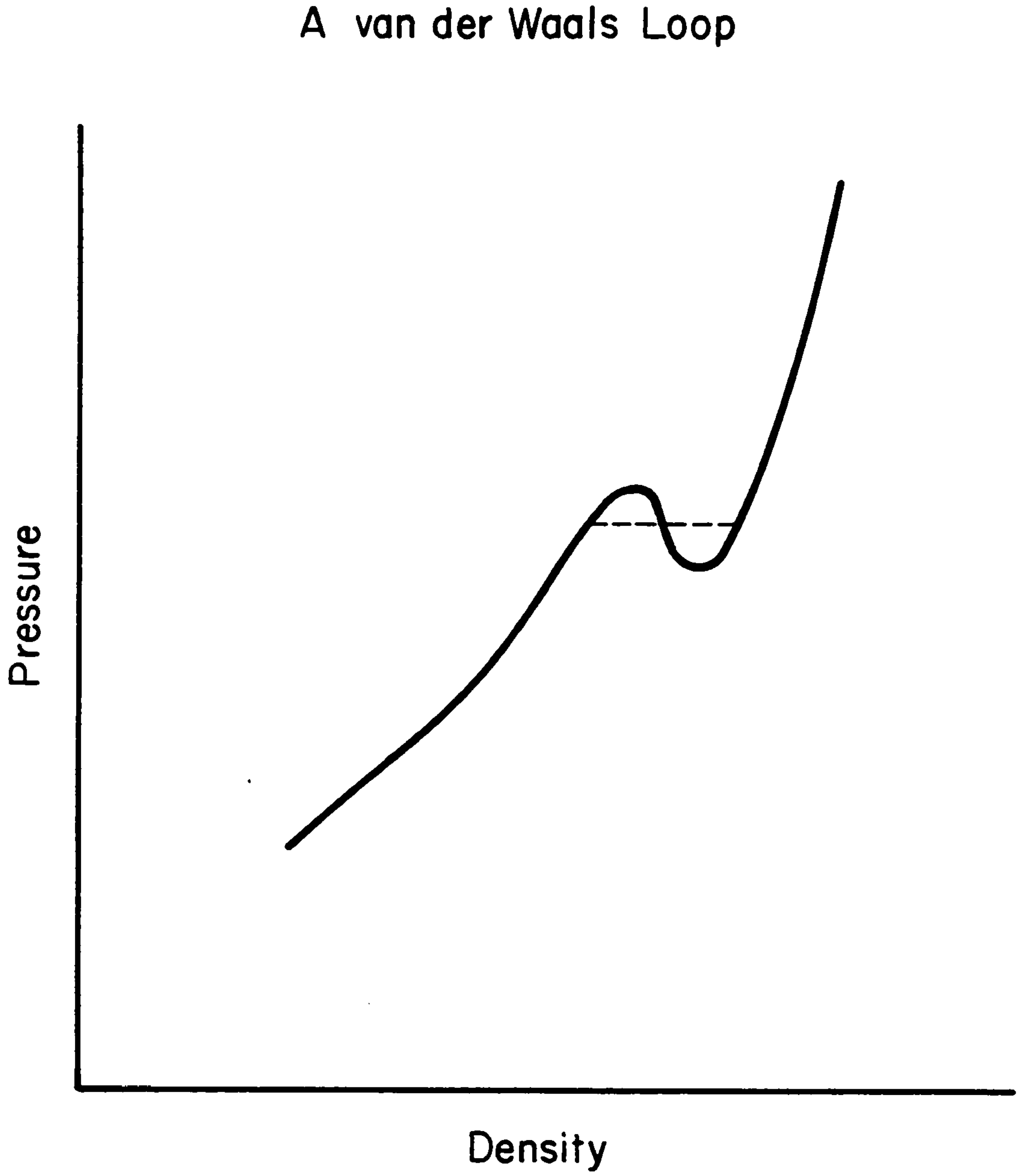


Figure 1.2 A van der Waals loop in which the pressure of the system decreases as its density increases. The dashed line indicates the behavior of an infinite system, which cannot exhibit a van der Waals loop in equilibrium.

Chapter 2: The Monte Carlo Simulation

2.1 Introduction

In the previous chapter we discussed briefly the two methods that are commonly used to perform computer simulations of many-body systems. We now present a detailed review of the Monte Carlo method of simulation to prepare for the discussion of implementing a concurrent Monte Carlo algorithm in the following chapter. Since the concurrent Monte Carlo algorithm differs in several important ways from its sequential counterpart, we must carefully establish the conditions that are necessary for the concurrent algorithm to be correct. In particular, we will note that the specific order in which the simulation updates the many degrees of freedom of the system is not important as long as the simulation updates all of them often enough for the system to reach equilibrium.

After reviewing the Monte Carlo method, we describe the specific system that we will simulate. The system consists of many particles interacting with the Lennard-Jones potential in a rectangular region with periodic boundary conditions. Since the system being simulated is relatively small, the effects of the boundary conditions can be important, so we examine some of the effects due to imposing periodic boundary conditions on the system. As in the previous simulations described in Section 1.3, we will truncate the interaction potential used in our simulation. We then consider some of the effects due to the truncated potential and mention some methods of partially correcting for its effects.

We then conclude the chapter by applying the discussion of the Monte Carlo method to the two-dimensional Lennard-Jones system that we will simulate in order to obtain specific update procedures. In particular, we construct the partition functions that describe both the constant-density and constant-pressure simulations. Using these partition functions, we extend the discussion in Section 2.2 and obtain the update procedure for a constant-pressure simulation.

2.2 Theoretical Basis of Monte Carlo

As we mentioned in Section 1.3, the Monte Carlo method simulates a system by generating a sequence of configurations of the particles in the system according to the probability of each configuration's occurring. From statistical mechanics, we know that the probability P of configuration r's occurring in an equilibrium system is proportional to its *Boltzmann factor*[22]

$$P(r) \propto e^{-\beta E(r)} \tag{2.1}$$

where $\beta = (kT)^{-1}$ and $E(r)$ is the energy of configuration r. The distribution of states resulting from the probabilities given by the Boltzmann factor is known as the *canonical distribution.* The normalization factor for the canonical distribution is the *partition function*, which is the sum of the Boltzmann factors of all possible states of the system. For instance, the partition function for a system of N particles at fixed volume V and temperature T is

$$Z = \frac{1}{h^{2N} N!} \int_{-\infty}^{\infty} d^2p_1 \cdots d^2p_N \int_V d^2q_1 \cdots d^2q_N \, e^{-\beta E} \tag{2.2}$$

where p_i and q_i are the momentum and position of particle i, respectively.

Knowing the partition function of a system allows us to obtain all of its properties by taking the appropriate derivatives of the partition function. Thus, evaluating the partition function of the system numerically would seem to be a method of studying a physical system in equilibrium. In principle, we could calculate the partition function by averaging the value of its integrand over many configurations in which the particles' coordinates are randomly selected. However, since the Boltzmann factor is usually a very sharply peaked function, the partition function cannot be evaluated directly in practice, unless the system contains only a few particles or is in the low-density, high-temperature limit.

Rather than evaluating the partition function itself, we can obtain the value of any physically measurable quantity, A, from its expectation value in equilibrium

$$\langle A \rangle = \frac{\int d^2p_1 \cdots d^2p_N \int d^2q_1 \cdots d^2q_N \; A \; e^{-\beta E}}{\int d^2p_1 \cdots d^2p_N \int d^2q_1 \cdots d^2q_N \; e^{-\beta E}} \; . \tag{2.3}$$

If A depends only on the particles' coordinates and not on their momenta, the total energy E of the configuration in (2.3) can be replaced by the potential energy U since the integrals over the momenta cancel. Still, the remaining $2N$-dimensional integrals are no easier to evaluate directly than the partition function itself. However, if we generate a set of configurations in which each configuration occurs with a frequency proportional to its Boltzmann factor, simply averaging A over the set of configurations gives the expectation value in (2.3). Since A is normally much less sharply peaked than the Boltzmann factor itself, we can evaluate expectation values of the partition function even though the underlying probability distribution for the configurations is very sharply peaked. This method of evaluating expectation values is known as *importance sampling* since only the configurations with large Boltzmann factors have a reasonable probability of occurring.

In 1953 Metropolis et al.[23] reported a method of generating configurations with the required probability distribution. They start with all the particles in an arbitrary configuration, such as a crystal lattice, and then successively update the position of each of the particles with the following procedure. First, they select a new position for the particle:

$$\begin{aligned} x_{new} &= x_{old} + \tfrac{1}{2}\Delta r_1 \\ y_{new} &= y_{old} + \tfrac{1}{2}\Delta r_2 \end{aligned} \tag{2.4}$$

where the r_1 and r_2 are uniformly distributed random numbers between 0 and 1, and Δ is an adjustable parameter. Next, they calculate the change in energy due to moving the particle. If $\Delta E < 0$, i.e., the new position of the particle results in a lower energy, the move is accepted. If $\Delta E > 0$, the move is accepted with a probability of $e^{-\beta \Delta E}$ by selecting another uniformly distributed random number r between 0 and 1 and accepting the move if $r < e^{-\beta \Delta E}$. Otherwise, the move is rejected and the particle is returned to its old position. Regardless of whether the move is accepted, the resulting configuration always counts as a new configuration when computing expectation values.

We now examine whether the Metropolis procedure generates configurations whose probabilities of occurrence are proportional to their Boltzmann factors. We first note that the procedure is *ergodic*; i.e., the procedure allows any point in configuration space to be reached. Since each particle can move anywhere within a square of side Δ with a nonzero probability, it can move anywhere in the simulation volume after a sufficient number of moves. Thus, the Metropolis procedure can generate all possible configurations of the system, making it ergodic.

We now consider an ensemble of many identical systems. For simplicity we assume that the number of states of the system is finite and that the number of systems in state r is n_r. In addition, we label the sequence of particle updates with a time parameter, even though the sequence of updates does not correspond to the true dynamical evolution of the system. The change in the number of systems in state r is given by the master equation[24]

$$\frac{dn_r}{dt} = \sum_s n_s P_{sr} - \sum_s n_r P_{rs} \qquad (2.5)$$

where P_{sr} is the probability of the system making a transition from state s to state r in one update. When the system is in equilibrium, $\frac{dn_r}{dt} = 0$ for all r. Since the system must be in some state after an update, we have

$$\sum_s P_{rs} = 1 \ . \qquad (2.6)$$

Thus, in equilibrium (2.5) becomes

$$n_r = \sum_s n_s P_{sr} \quad . \tag{2.7}$$

The condition in (2.7) is usually satisfied by imposing the stronger condition of *detailed balance* on the system, which is

$$n_r P_{rs} = n_s P_{sr} \tag{2.8}$$

for all states r and s. Detailed balance states that the transition rate from state r to state s is the same as the transition rate from state s to state r. Since the ratio of the states' Boltzmann factors gives the correct ratio of n_r to n_s in the equilibrium system, we obtain from the condition of detailed balance

$$\frac{n_r}{n_s} = \frac{P_{sr}}{P_{rs}} = e^{-\beta(E_r - E_s)} \quad . \tag{2.9}$$

We now check whether the Metropolis update procedure satisfies detailed balance and (2.9) by considering two states, r and s. According to (2.4), a particle has an equal probability of moving anywhere within a square region about its old position and zero probability of moving elsewhere. Thus, state r has a nonzero probability of making a transition to state s in one update if state s differs by the position of a single particle and that particle in state s is within the square region about the same particle in state r. When state r can become state s in a single update, state s can also become state r in a single update since the square region is symmetric about the particle being updated. Thus, we have $P_{rs} = P_{sr}$ for all states r and s before applying the acceptance criteria, which depend on the energies of the states. We see that all states where $P_{rs} = P_{sr} = 0$ satisfy (2.8) trivially.

We now consider the case where $E_r < E_s$ and apply the acceptance criteria of the Metropolis update procedure. When P_{rs} and P_{sr} are nonzero, we get

$$\begin{aligned} P_{sr} &= 1 \\ P_{rs} &= e^{-\beta(E_s - E_r)} \end{aligned} \tag{2.10}$$

which clearly satisfies (2.9). Thus, the Metropolis update algorithm satisfies detailed balance and, as we have already noted, it is ergodic. Together these properties ensure that the distribution of states in the ensemble of systems will approach the canonical distribution.[24]

We now generalize the Metropolis algorithm by considering the system as consisting of many degrees of freedom f_i. In the two-dimensional system of particles, the f_i are the two independent coordinates giving the position of each particle in the system. The first step in the update procedure is to choose new values for a subset of the degrees of freedom using

$$f_i^{new} = f_i^{old} + \tfrac{1}{2}\Delta_i r_i \tag{2.11}$$

where the r_i are uniformly distributed random numbers between 0 and 1. The values of Δ_i do not need to be the same for all the degrees of freedom. In fact, a complicated simulation could have several types of degrees of freedom and require different values of Δ_i. However, choosing each of the f_i^{new} with equal probability in a symmetric region about its value of f_i^{old} is essential. In the two-dimensional system that we will investigate, the simulation updates the two degrees of freedom of a particle together and uses the same value of Δ.

After choosing the new positions of the subset of the degrees of freedom, we measure the energy difference between the new and old configurations and apply the Metropolis acceptance criteria. When a simulation updates several degrees of freedom at the same time, the acceptance criteria must accept or reject all of the degrees of freedom together in order to satisfy detailed balance. We note that the Metropolis algorithm does not specify the order in which the simulation updates the degrees of freedom. As long as it updates all of the degrees of freedom so that it can generate all possible configurations, the distribution of configurations will approach the canonical distribution.

In our discussion of the Monte Carlo method, we have mentioned that an update algorithm which is ergodic and satisfies detailed balance causes the distribution of configurations to approach the canonical distribution, regardless of the system's initial configuration. However, we cannot predict its rate of approach to equilibrium since the Monte Carlo procedure gives no estimate of the number of updates required for the simulation to reach equilibrium. We can only assume that the simulation is in equilibrium when all of its expectation values remain constant as it updates the system. Of course, we can never determine that a simulation has reached equilibrium and that it has obtained accurate expectation values. Thus, metastability and critical slowing down can be serious problems, especially near phase transitions where strong metastability or long time scales often occur.

The values of Δ_i in (2.11) can have a large effect on the simulation's rate of approach to equilibrium, so we should choose their values carefully. If the Δ_i are too small, the degrees of freedom change only a small amount in each update, so the system moves very slowly through configuration space. A similar effect occurs in a molecular dynamics simulation when its time step is too small since the particles can move only a short distance during each update. If instead the values of Δ_i are too large, the acceptance rate will be low since most of the moves are rejected even though particles occasionally move large distances in a single update. In contrast to a molecular dynamics simulation, which gives incorrect results when its time step is too large, the size of the values of Δ_i does not affect the correctness of the Monte Carlo simulation. In addition to the values of the Δ_i, the number of degrees of freedom that the simulation moves in a single update also affects the acceptance rate since it must accept or reject all of the moves together. The results of these considerations is that most simulations update only a few degrees of freedom per update and adjust the Δ_i to obtain about a 50% acceptance ratio.

We now examine the sources of errors that occur in Monte Carlo simulations. Clearly, using configurations that are not in equilibrium to calculate expectation values results in errors. Such errors may be impossible to detect since the system may appear to be in equilibrium when it is really in a metastable state. Another source of error is statistical error in the expectation values, which is always present in a simulation due to thermal fluctuations in the individual measurements. Averaging the measurements over independent configurations reduces the statistical error, but successive configurations are highly correlated since each update changes the configuration only a small amount. Thus, the statistical error of an expectation value may decrease very slowly during a simulation, especially when the simulation is near a phase transition where long time scales are important. Finally, differences between the system that is being simulated and the system that we really want to study introduces systematic errors. For instance, we simulate the melting transition in a two-dimensional system consisting of a few thousand particles when we really want to study the melting transition in the thermodynamic limit. If the system that is being simulated exhibits finite-size effects, which are due to its relatively small number of particles, the expectation values contain systematic errors, even though the the simulation accurately describes the finite system.

2.3 Description of Lennard-Jones System

As we have already mentioned, the particular two-dimensional system that we will investigate consists of many classical particles interacting through the Lennard-Jones potential. The Lennard-Jones potential, which acts between pairs of particles, is given by

$$\phi = 4\epsilon\left[\,(\frac{\sigma}{r})^{12} - (\frac{\sigma}{r})^{6}\right] \tag{2.12}$$

where ϵ is its energy parameter, σ is its length parameter, and r is the distance between the two particles. For a system of N particles interacting through the Lennard-Jones potential, the total potential energy of the system is

$$U = 4N\epsilon\left[\sigma^{12}\langle r^{-12}\rangle - \sigma^{6}\langle r^{-6}\rangle\right] \tag{2.13}$$

where $\langle r^{-12}\rangle$ and $\langle r^{-6}\rangle$ are averages over a configuration and have the form

$$\langle r^{-n}\rangle = \frac{1}{N}\sum_{\langle ij\rangle} r_{ij}^{-n} \;. \tag{2.14}$$

The summation in (2.14) extends over all distinct pairs of particles in the configuration and r_{ij} is the distance between particles i and j.

The Lennard-Jones potential, which we illustrate in Figure 2.1, is characterized by a strongly repulsive core at short distances and an attractive tail that falls off as r^{-6} at large distances. One reason that the Lennard-Jones potential is interesting is that it describes the interaction between atoms of the noble gasses reasonably well. For instance, the two-dimensional Lennard-Jones system should approximately describe a single layer of xenon atoms adsorbed onto graphite.[25] In addition, many investigators have performed both molecular dynamics and Monte Carlo simulations of the two-dimensional Lennard-Jones system, as we noted in Chapter 1.

The Lennard-Jones energy and length parameters, ϵ and σ, respectively, and the number of particles in the simulation, N, allow us to define a set of dimensionless, intrinsic units for the quantities that we will use in the simulations. These reduced quantities are denoted by a superscripted asterisk. For instance, the potential energy, pressure, volume, and density become $U^* = \frac{U}{\epsilon N}$, $p^* = \frac{p\sigma^2}{\epsilon}$, $V^* = \frac{V}{N\sigma^2}$, and $\rho^* = \frac{1}{V^*}$, respectively. Other reduced quantities are generated in a similar way.

In order to make the Monte Carlo simulation computationally feasible, we will not use the full Lennard-Jones potential given by (2.12). Instead, we will truncate the potential at a particle separation of 3σ, which is comparable to the truncation distance used by other investigators.[11,13,17,21] Without some type of truncation, the contributions from all of the particles in the system would have to be summed in order to calculate the potential energy differences required by the Monte Carlo update algorithm. Clearly, treating the short-range Lennard-Jones potential as a long-range interaction is undesirable. By fixing the range of the interaction with an explicit truncation, we greatly reduce the amount of computation needed to calculate the potential energy differences. The particular value of the truncation distance is quite arbitrary, but the distance of 3σ should retain the essential physics of the system while keeping the computational requirements of the simulation within reasonable bounds.

Using the truncated Lennard-Jones potential will affect the results of the simulation in several ways, but many results should not be affected substantially. For instance, the contributions to the potential energy differences from particles that are outside the truncation distance are relatively small, so using the truncated potential should have only a small effect on the acceptance of Monte Carlo moves. In addition, we expect that the particles beyond the truncation distance are distributed rather evenly, so even their small contributions should cancel almost completely. Another way of examining the effect of the truncated potential on the acceptance of Monte Carlo moves is to express the positions of the particles as a sum over the system's normal modes. The normal modes that are not correctly represented in a simulation using the truncated potential are those whose wavelengths are greater than or approximately equal to the truncation distance. Since the number of such modes is small, we expect the effect of the long wavelength modes on the acceptance of Monte Carlo moves to be small.

Since we expect the truncated potential to have little effect on the acceptance of Monte Carlo moves, which determines the physics of the system being simulated, we do not expect the truncated potential to affect the details of the melting transition substantially. From a different point of view, at very high temperatures and densities the geometric packing effects due to the repulsive core of the Lennard-Jones potential dominate the melting transition. The attractive tail of the potential is irrelevant as is the truncation of the potential at large distances. Even at low temperatures and densities, the repulsive core still plays a large role since the fluid phase near the melting transition is difficult to compress. The Lennard-Jones energy parameter, ϵ, gives the characteristic energy of the competition between the attractive and repulsive components of the potential. Since the characteristic energy at the truncation distance is so much weaker than ϵ, we do not expect that using the truncated potential will affect the details of the phase transition substantially.

Although we have shown that the truncated potential should have only a small effect on the evolution of the Monte Carlo simulation, the truncated potential does have a significant effect on some types of measurements. Since the truncated part of the Lennard-Jones potential is always negative, as we see in Figure 2.1, the measured potential energy of the system is always somewhat higher than it would be if the full Lennard-Jones potential were used. The truncated potential also affects related quantities, such as the pressure of the system, in a similar way. One way of correcting for the effects of the truncated potential assumes a uniform distribution of particles beyond the truncation distance and calculates corrections to the various measurements. While the distribution of particles that constitute a fluid is rather uniform when averaged over time, the distribution of particles in a crystal is certainly not uniform, so another method of correction should be found. Fortunately, we have almost completely removed the effects of the truncated potential in the results of our simulations by using additional corrections that we will discuss in Chapter 4.

In addition to specifying the interaction potential, we must also impose boundary conditions on the system in order to specify completely the interaction between particles. We impose periodic boundary conditions on our system in the form of a rectangular unit cell whose dimensions are commensurate with the triangular lattice of the Lennard-Jones crystal. In the relatively small systems that we will simulate, periodic boundary conditions have the important advantage over other types of boundary conditions of not allowing edge effects to dominate the simulation results. We use a rectangular unit cell rather than a 60° rhombus so that the simulation program can more easily determine the location of the boundaries. Since we want to run all of the simulations with the same boundary conditions, we make the unit cell commensurate with a triangular lattice to avoid simulating a solid under an applied shearing stress.

Since the systems being simulated are relatively small, the type of boundary conditions used in computer simulations can have a large effect on the results, especially when the system is near a phase transition. For instance, near a higher-order phase transition the correlation length is very large. When the correlation length is larger than half the distance across the unit cell, the system becomes overcorrelated since the effect of each particle wraps around the system in both directions. However, even if the correlation length does not become large, periodic boundary conditions can still have a large effect due to the periodic structure that they impose on the system. Near the melting transition, the ordering imposed by the periodic boundary conditions favors the solid phase over the fluid phase if the periodicity of the boundaries is commensurate with the periodicity of the crystal lattice.

Another effect of periodic boundary conditions arises because the system effectively has no surfaces.[19] When a solid melts at constant pressure, it must be allowed to expand since the fluid phase typically has a lower density than the solid phase. Thus, under conditions of constant pressure, melting normally starts at the surface of the solid where the resulting fluid is free to expand and progresses into the solid from the surface. When the system has no surfaces, small droplets of fluid cannot form at the true melting temperature because the surrounding solid does not allow the fluid droplets to expand. Due to the surrounding solid, the pressure on any droplets that do form is greater than the externally applied pressure, so the temperature required to form the droplets must be greater than the true melting temperature at the specified pressure. Thus, the periodic boundaries constrain the system to superheat during a constant-pressure simulation, as Abraham[19] observed.

2.4 Monte Carlo Update Procedures

We now combine the Metropolis algorithm with the truncated Lennard-Jones interaction potential and the periodic boundary conditions to obtain the specific Monte Carlo update procedures that we will use in our simulations. The effects of the periodic boundary conditions appear both explicitly in the limits of integration in the partition function and implicitly in the potential energy of the system. Since we will simulate the Lennard-Jones system under conditions of both constant density and constant pressure, we must construct the partition function for each type of simulation as the first step in designing the update procedures. As we mentioned in Section 2.2, the partition function is the sum of the Boltzmann factors of all possible states in the system. In the system that we are studying, all of the degrees of freedom are continuous quantities, so the sum becomes an integral, as in (2.2).

When we specify the temperature and density of the system, the only degrees of freedom are the two coordinates that give the position of each particle. Thus, the partition function is

$$Z_U = \int_0^X dx_1 \cdots dx_N \int_0^Y dy_1 \cdots dy_N \, e^{-\beta U} \tag{2.15}$$

where N is the number of particles and X and Y give the dimensions of the rectangular unit cell. We have used Z_U to indicate that we have ignored the constants and kinetic terms that appear in (2.2) but cancel when we calculate expectation values. The density of the simulation is $\rho^* = \frac{N\sigma^2}{XY}$, so we can change the simulation density by simply changing σ, rather than by changing X and Y.

When we specify the temperature and pressure of the system, the simulation updates the volume of the system in addition to the particles' coordinates so that it maintains a constant average pressure at the specified value. Thus, the volume of the system is an additional degree of freedom, so we must add an integral over the volume of the system to the partition function, giving

$$Z_U = \int_0^\infty dV \int_0^X dx_1 \cdots dx_N \int_0^Y dy_1 \cdots dy_N \, e^{-\beta(U+pV)} \tag{2.16}$$

where p is the simulation pressure and X and Y are now functions of the system volume V. Since we are not integrating over the shape of the system, the ratio of X to Y remains constant. The energy appearing in the Boltzmann factor is no longer simply the potential energy of the system but also includes the mechanical energy of the system. Of course, the mechanical energy is also present in the system maintained at constant density, but no mechanical work is ever performed on the system since its volume is constant, so its mechanical energy is ignored.

Having constructed the partition functions that describe the constant-density and constant-pressure simulations, we can obtain the specific Monte Carlo update procedures using the Metropolis algorithm. Since we already discussed a constant-density simulation during the presentation of the Metropolis algorithm in Section 2.2, we note only that we use the Lennard-Jones potential energy in the acceptance criteria.

However, a more complicated situation arises in the constant-pressure simulation because the simulation updates the volume of the system in addition to the positions of its particles. In order to determine the acceptance criteria for updating the volume, we examine the Boltzmann factor of the partition function given by (2.16). In particular, we must determine the volume dependence in (2.16) and specify exactly what integrating over the volume of the system means. Since the limits of integration, X and Y, depend on the volume of the system, the importance sampling technique of obtaining expectation values does not apply directly to (2.16). Thus, we would like to rewrite (2.16) so that its volume dependence is explicit.

By substituting $x_i' = \frac{x_i}{X}$ and $y_i' = \frac{y_i}{Y}$ into (2.16) we obtain

$$Z_U = \int_0^\infty dV\, V^N \int_0^1 dx_1' \cdots dx_N' \int_0^1 dy_1' \cdots dy_N'\, e^{-\beta(U + pV)} \qquad (2.17)$$

where we have used $V = XY$. Besides making the volume dependence explicit, we have specified that the particles' coordinates are to be rescaled when the volume of the system changes. Including the V^N term in the Boltzmann factor gives the quantity that corresponds to the energy in the Metropolis acceptance criteria,

$$W = U + pV - kTN \ln V \ . \qquad (2.18)$$

For convenience, we will update the positions of the particles separately from the volume in the constant-pressure simulation. Since simply moving a particle affects only U, (2.18) leads to the same acceptance criteria for updating a particle as in the constant-density simulation. However, changing the volume by randomly selecting a new value of V from a symmetric region about its old value causes all three terms in (2.18) to contribute to ΔW. The new volume is always accepted if $\Delta W < 0$ and is accepted with a probability of $e^{-\beta\Delta W}$ if $\Delta W > 0$. Although the position of a particle and the volume could be changed together, we will find that such an update procedure is undesirable in the concurrent update algorithm that we will discuss in the next chapter.

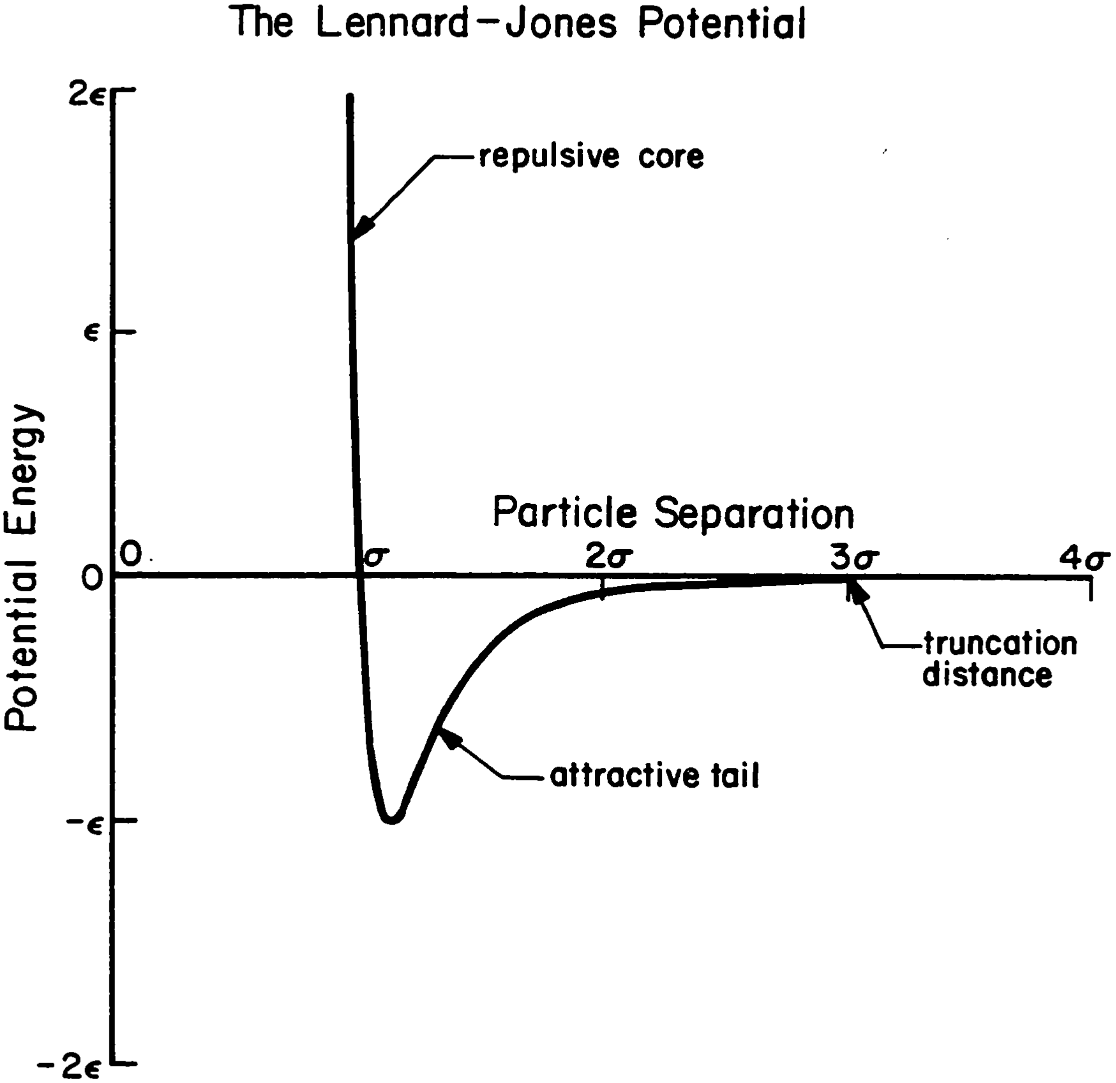


Figure 2.1 The Lennard-Jones potential, whose energy and length parameters are ϵ and σ, respectively. The Monte Carlo simulation truncates the potential at 3σ to reduce its computational requirements.

Chapter 3: Implementation on a Concurrent Processor

3.1 Introduction

Having described the method of Monte Carlo simulation and derived the appropriate formulas for simulating particles interacting through the Lennard-Jones potential in Chapter 2, we are ready to examine the implementation of the simulation on a computer. We first discuss some of the general issues involved in programming the simulation. In particular, we consider methods that take advantage of the short-range nature of the interaction potential. Such general issues depend on the details of the application being programmed but not on type of computer used to run the application.

We then discuss a specific type of concurrent processor, which is the computer that we will use to run the simulation. A concurrent processor consists of many independent processing elements, called *nodes*, that work together on a single application. Such a computer has the potential to provide far greater computer power than what is currently available. Current sequential computers are approaching fundamental limitations on their processing speeds, so substantially increasing the available computer power requires exploiting some type of concurrency. One method of exploiting concurrency in an application involves operating on vectors, which may contain many elements, as if they were single units. In the concurrent computation project at Caltech, we use a different approach and exploit concurrency by assigning separate pieces of the application to the many interconnected nodes comprising the concurrent processor that we built. The Caltech concurrent processor is a *homogeneous machine* since all of the nodes are identical. Although the nodes are not very powerful individually, they can give the performance of a much faster and much more expensive computer by working together.

However, we can realize the potential performance of the concurrent processor only if we can program it to run efficiently, so that most of the nodes are doing useful computation most of the time. In this chapter we explore the major issues and problems that arise when adapting the Monte Carlo algorithm to run on a concurrent processor. Although the actual simulation program consists of many separate pieces, most of the pieces are fairly standard, so we confine our discussion to the Monte Carlo update procedure, which is the essential component of the simulation. We discuss a particularly interesting feature of the concurrent simulation, which is that its results are inherently irreproducible. As an indication of how well we have achieved the goal of obtaining an efficient concurrent implementation, we present a formal analysis of the efficiency of the update algorithm, which includes direct measurements of its efficiency.

3.2 Data Structures and Decomposition

Before discussing the implementation of the Monte Carlo simulation on the concurrent processor, we first review some of the characteristics of the two-dimensional Lennard-Jones simulation. The Monte Carlo procedure updates one particle at a time by randomly selecting a new position for the particle and accepting or rejecting the move based on the resultant difference in potential energy. Thus, we must separately sum the contributions to the potential energy that involve the particle being updated for both its new and old positions and then take the difference of the sums.

An important feature of the Lennard-Jones potential is that it is a *short-range* potential, meaning that the potential decreases rapidly with increasing particle separation. In fact, the potential that we use in the simulation is truncated at a particle separation of 3σ, so all of the particles in the system clearly do not contribute to the potential energy sums of the particle being updated. Thus, the particles in the simulation can be conceptually divided into two groups: the nearby particles, which directly affect the particle being updated, and the distant particles, which have no direct effect. Although the two groups are different for each of the particles, the concept of *spatial locality* is important since the interaction potential is short-range. If the interaction potential were instead long-range, such as the gravitational potential, all particles would directly affect all other particles since the potential does not decrease sufficiently rapidly as the separation of the particles increases. Although the magnitude of the effect still depends on the separation of the particles, spatial locality is not an important concept because the effect of distant particles cannot be ignored.

A good implementation of the Monte Carlo simulation should exploit the short-range nature of the interaction potential to reduce the number of arithmetic operations required to update the particles in the simulation. A standard measure of the number of operations required by an algorithm is its complexity, which is often specified by the dependence of the number of required operations on the number of items being handled by the algorithm. In many cases, the exact number of operations is less important than the functional dependence on N, the number of items. For instance, many sorting algorithms have a complexity of order $N \log N$, and an ordinary long-range force algorithm has a complexity of order N^2. If the coordinates of the N particles in our Monte Carlo simulation reside in one list, a naive update algorithm would have to search the whole list in order to determine which particles contribute to the potential energy sums. Searching the whole list every time that a particle is updated results in an algorithm with a complexity of order N^2 for one Monte Carlo sweep, which updates each of the N particles once. By assigning all of the particles to one list, we have failed to exploit the short-range nature of the interaction potential to minimize the complexity of the problem.

Correctly organizing the coordinates of the particles in the system can greatly reduce the complexity of the Monte Carlo update algorithm. We can avoid searching through a single list containing all of the particles if we separate the particles into several lists in such a way that we can determine which lists need to be searched when any particle is updated. By assigning to each list a distinct portion of the simulation volume and the particles that lie within it, we can take advantage of the spatial locality of the interaction potential by a technique known as *geometric hashing*. Since each of the lists contains the particles within a fixed region of space, called a *cell*, we can easily determine which lists of particles must be searched during the update of any particle. Thus, by dividing the simulation volume into cells and assigning the particles in each cell to separate lists, we can reduce the complexity of the algorithm to order N. The coefficient of N depends on the average number of lists that need to be searched during an update and on the average number of particles per list.

In order to minimize the complexity of the Monte Carlo update algorithm, we must determine the optimal way of dividing the total simulation volume into cells. Actually, rather than divide an arbitrary simulation volume, we will determine the optimal parameters of cells and allow only simulation volumes that are built from such cells. We note that all of the cells should be the same size and shape so that determining which cells could contain particles that affect the particle being updated is relatively simple. The shape of the cells should reflect the periodic boundary conditions imposed on the system as described in Section 2.3, so the cells should be rectangles whose dimensions allow some number of particles to form a perfect crystal. In addition, the cells should be as nearly square as allowed by the boundary conditions so that they preserve a strong sense of spatial locality.

We determine the optimal size of the cells by minimizing the product of the average number of particles per cell and the average number of cells that must be searched during the update of a particle. Clearly, the cells should be rather small since the purpose of dividing the simulation volume into cells is to avoid having all of the particles in one large list. On the other hand, if the cells are smaller than some limit, the number of cells that must be searched increases in inverse proportion to the average number of particles per cell, so the complexity remains constant. Since eight cells completely surround each of the cells, a convenient choice for the size of the cells, which is nearly optimal, is the minimum size that is large enough to ensure that no more than nine cells will need to be searched to update any particle.

Thus, we use cells whose minimum dimension is larger than the range of the interaction potential plus the maximum distance that a particle can move in one update. We recall that the range of the truncated Lennard-Jones potential is 3σ and that (2.4) gives the maximum distance that a particle can move during an update as $\frac{1}{2}\Delta$. Although both σ and Δ vary in order to change the density of the system and adjust the acceptance rate, respectively, we can still fix the size of the cells without imposing unnecessary restrictions on the simulation. We simply choose the cells' minimum dimension to be larger than the largest value of $(3\sigma + \frac{1}{2}\Delta)$ that will occur during a simulation. However, since the shape of the cells should reflect the periodic boundary conditions, we must select the number of particles forming a perfect crystal that the cell will contain before making a final determination of the cells' size.

Applying the previous considerations to the concurrent Monte Carlo update algorithm slightly complicates the previous argument because the simulation does not confine the particles to their original cells. Since each particle is free to move anywhere in the simulation volume, the particles must be sorted periodically into their correct cells. In an implementation of the simulation on a sequential computer, each particle could be stored in its correct cell immediately after being updated. However, in a concurrent implementation of the simulation a much better procedure is to update all of the particles once and then sort them into their correct cells. With a delayed sorting scheme the boundaries of adjacent cells effectively overlap during an update by the maximum distance that a particle can move in one update. Thus, the minimum dimension of the cells must be $(3\sigma + \Delta)$, rather than $(3\sigma + \frac{1}{2}\Delta)$ as we noted previously.

Keeping in mind all of the considerations for determining the size of the cells, we choose the dimensions of the cells such that 16 particles forming a perfect crystal fit into a cell when periodic boundary conditions are imposed at its edges. Since we have fixed the average number of particles per cell, the number of cells determines the total number of particles in the system. Figure 3.1 illustrates a system containing 1024 particles that consists of an 8×8 grid of cells and shows the range of the potential around several of the particles.

Besides reducing the complexity of the algorithm, assigning the particles to cells has another major advantage for a concurrent implementation of the update algorithm. Before the nodes can work together on a problem, we must assign a portion of the problem to each node. We decompose the Lennard-Jones system for concurrent processing by simply assigning the cells to the nodes, so each node updates the particles in one or more cells. Clearly, assigning the same number of cells to each node is important because it balances the computational loads of the nodes. If the problem had not already been broken into cells for a completely different reason, the decomposition might not have been so obvious. However, many types of applications seem to have a natural decomposition. The decomposition of a problem can be characterized by its *decomposition topology*, which specifies the connections between the pieces of the decomposed problem. For instance, the cells assigned to each node in the Lennard-Jones simulation form a two-dimensional grid with periodic boundaries in which each node must be able to communicate with the eight nodes that surround it.

3.3 Description of the Concurrent Processor

The concurrent processor used to run the simulation is an ensemble of independent processing elements communicating with each other by exchanging messages. Each of the independent processors executes its own instruction stream and operates on a separate set of data; such an ensemble is known as a multiple-instruction, multiple-data (MIMD) machine. An environment in which the processors communicate by transmitting and receiving data items is a *message-passing* system. The concurrent processor that we discuss uses message-passing communication exclusively since it provides no facilities for sharing memory between any of the nodes.

Our original concurrent processor, the Mark I, consists of 64 nodes, each about as powerful as an IBM-PC. The memory on each node consists of 8K bytes of EPROM and 128K bytes of dynamic RAM with an extra parity bit for each byte of dynamic RAM that allows single-error detection in the RAM. The processing unit consists of the Intel 8086 microprocessor and the Intel 8087 floating-point coprocessor, which has a nominal speed of about 20 Kflops at a 5 MHz clock rate. Our second-generation concurrent processor, the Mark II, is very similar to the Mark I, except that it runs at a clock rate of 8 MHz and has twice as much memory. Detailed timing measurements of the Mark II system are provided by Otto, Kolawa, and Hey.[26]

In addition to the processor and memory, which are common to all computers, each of the Mark I nodes has six high-speed communication channels that implement the message-passing system. A single channel can connect only two nodes, so all communications are point-to-point. In contrast, if many nodes resided on a common bus, broadcast communication could be a primitive operation. The communication channels, which are implemented in each node's hardware, consist of separate transmitting and receiving components and a collection of wires that connect the transmitting component of one node to the receiving component of another node and vice versa. The receiving component of a channel has a buffer into which the transmitting node can write and from which the receiving node can read. However, the transmitting node must not write into the buffer if it is already full, and the receiving node must not read from the buffer if it is empty, so the channel hardware also provides information on the status of the channel buffer to the transmitting and receiving nodes.

The channel hardware provides its status information to the transmitting and receiving nodes with a separate *interrupt line* for each node. An interrupt line can either interrupt the processor or be polled by the processor, depending on the software support for the communication channels. The channel hardware sets the transmit interrupt line only when its channel buffer is empty. Similarly, the channel hardware sets the receive interrupt line only when its channel buffer is full. Thus, the channel can only transmit and receive data in fixed-length packets whose length depends on the size of the channel buffer. On both the Mark I nodes and the Mark II nodes, all communications involve eight byte packets.

As we have mentioned, a channel consists of both a transmitting and a receiving component so that the nodes that it connects can communicate messages in both directions. If the two components of the channel are completely separate, both nodes that the channel connects can write into each other's channel buffer at the same time without interference. Such a channel is known as *full-duplex* since both nodes can transmit independently. Since the two components of the full-duplex channel are independent, they require two sets of wires to connect them. Using a *half-duplex* circuit, in which both components of the channel use the same wires, reduces the number of wires that the channel requires, but it allows only one of the nodes to transmit at a time. Since the number of wires is not prohibitive and full-duplex channels are easier to use, the channels in both the Mark I and Mark II nodes implement full-duplex communication.

Having described the channels that implement internode communication, we now examine the *interconnection topology* of the concurrent processor, which specifies the pairs of nodes that are connected by a channel. The most general interconnection topology is the fully interconnected topology in which communication channels connect all pairs of nodes, allowing any node to communicate directly with any other node. However, building a machine with so many channels implemented in hardware is completely impractical unless the machine contains relatively few nodes. If the ensemble consists of N nodes, each node must have $N-1$ communication channels in order to communicate directly with all of the other nodes in the system, so the total number of channels in the ensemble is $\frac{N^2-N}{2}$. Since the total number of channels increases so rapidly as the number of nodes increases, building a large system using the fully interconnected topology is nearly impossible.

While many different interconnection topologies have been investigated, we will consider only the *hypercube*, which can be thought of as a grid topology where the dimensionality of the grid depends on the number of nodes. The N nodes in a d-dimensional hypercube, a d-cube for short, are indexed by d bit binary numbers and a communication channel connects pairs of nodes whose indices differ by only one bit.[27] Another way of defining a hypercube, which provides a more geometrical description, is recursive definition in terms of grids. We start with a 0-cube, which consists of a single node with no communication channels, i.e., a standard sequential computer. By connecting two 0-cubes with a communication channel, we construct a 1-cube. Connecting the corresponding nodes of two 1-cubes with an additional channel results in a 2-cube, which forms a square with the nodes at its vertices and communication channels on its edges. In general, connecting the corresponding nodes of two $(d-1)$-cubes with an additional channel per node results in a d-cube.

Thus, a d-dimensional hypercube consists of $N = 2^d$ nodes with d communication channels per node. The total number of channels in an ensemble of N nodes is $\frac{N \log_2 N}{2}$, which grows only slightly faster than the number of nodes. Since the required number of channels grows so slowly, it imposes little restriction on the number of nodes that can be connected as a hypercube. Although channels do not directly connect all of the nodes in a hypercube, the maximum distance between nodes is $\log_2 N$ steps, which also grows very slowly with increasing N. The six communication channels on the Mark I nodes are exactly enough to allow the 64 nodes to be connected as a 6-cube.

An important feature of the hypercube topology for many scientific and engineering applications is that it contains grids of each dimensionality that is lower than the dimensionality of the hypercube. In addition, if the number of nodes in each dimension of a grid is a power of two, the hypercube topology also ensures that the grid can have periodic boundaries. In grids whose dimensionality is lower than that of the hypercube, not all of the communication channels may be used. For instance, each node in a two-dimensional grid with periodic boundaries uses only four channels, leaving two of the channels on each node in the Mark I machine unused. An example of a grid that uses all of the channels is the $4 \times 4 \times 4$ grid with periodic boundaries, which is isomorphic to the 6-dimensional hypercube.

In addition to the 64 nodes of the Mark I concurrent processor, we have a *control processor* that handles tasks which do fit into the symmetry of the ensemble. Such tasks include the inherently sequential components that typically occur in concurrent algorithms and in the communication between the concurrent processor and a user or a file system. The control processor is appropriately named because one of its major tasks in a typical application is coordinating and monitoring the concurrent processor. We note that, because of its special role, the control processor has very little computation to perform in a good algorithm, which minimizes the amount of sequential components.

3.4 Programming a Concurrent Processor

The communication channels discussed in the previous section must have software support before we can use them in an application program. As we noted, the interrupt lines that report the channels' status can either be polled by the processor or cause the processor to be interrupted, depending on the type of software that supports the channels. Thus, all software implementations of internode communications can be divided into two groups based on the use of the interrupt lines. The far simpler type of implementation is polled communications, which we will discuss first.

In the most basic implementation of polled communications, only two routines, a transmitting routine and a receiving routine, are needed. The polled transmitting routine writes a single packet of data from the node's memory, starting at the location indicated by the routine's first argument, to the channel indicated by its second argument. However, before the processor can write the packet to the channel, it first determines that the channel buffer is ready by polling the transmit interrupt line until the line indicates that the buffer is empty. The length of time that the polled transmitting routine spends testing the interrupt line is indefinite, so it imposes a strong synchronizing condition on the application program by blocking the transmitting node until the receiving node empties the channel buffer.

The polled receiving routine reads a single packet of data from the channel indicated by its second argument and stores it in memory starting at the location given by its first argument. However, before reading a packet, it must determine that the channel buffer contains valid data by polling the receive interrupt line until the line indicates that the buffer is full. As in the case of the transmitting routine, the length of time that the polled receiving routine spends testing the interrupt line is indefinite. Thus, the polled receiving routine also imposes a strong synchronizing condition on the application by blocking the receiving node until the transmitting node writes a packet.

The strong synchronizing conditions that the polled communication routines impose have several consequences for application programs. The most important effect of the synchronization is that it forces the nodes to execute their programs in a loose lockstep. However, the application program imposes the lockstep through its use of the communication routines; the nodes do not actually run synchronously as they would if they shared a single instruction stream. Since the synchronization occurs only when the application program calls one of the communication routines, the lockstep may be sloppy, with the nodes proceeding at their own pace between communication steps in the algorithm. If one node has less computation to perform than nodes with which it communicates, the node wastes time while waiting to read or write. Thus, an efficient concurrent algorithm must carefully balance the computational loads of nodes that communicate. Balancing the loads between every communication step in the algorithm is essential because the slowest node always determines the amount of time taken by each computational step.

Another limitation on the polled communication routines results from requiring that all transmitting and receiving operations be programmed explicitly. Such a requirement limits the application program to using *expected* communications, so a node cannot write to another node unless the destination node knows in advance to receive the message. Although requiring expected communications may seem like a severe constraint, experience has shown us that it is not as serious as it first appears. For instance, the communication requirements of an application can be data-dependent as long as each node knows what data it will receive at every stage in the algorithm.

We refer to the polled communication routines as the *Crystalline Operating System* (CrOS) since they are most useful for implementing concurrent algorithms which are sufficiently regular and predictable. The problems must be regular in the sense that the participating nodes can expect the communications and that they distribute the computational load evenly so that the synchronizing effects of the polled communications do not adversely affect the algorithm. Applications that have these properties tend to be very symmetric, which is why we refer to them as being crystalline. In many of the applications programmed so far using CrOS, some type of underlying grid gives the application a regular structure which helps to ensure that all of the nodes communicate together. Often, all of the nodes apply the same algorithm to equal amounts of data, so the computational loads are well balanced. Although the routines that comprise CrOS do not constitute a complete operating system in its normal sense, they do provide an adequate programming environment for developing and running many applications on a concurrent processor.

Unfortunately, the routines that comprise CrOS do not provide an adequate environment for implementing a Monte Carlo simulation of the Lennard-Jones system on the concurrent processor. Due to the absence of an underlying grid, the simulation does not have a regular structure that would allow the necessary communications to be easily specified in advance. Since the communications needed during the update of a particle depend in detail on the position of the particle within its cell, each node will typically have very different communication requirements, which change as the particles are updated. In addition, the communications are not restricted to adjacent nodes in the hypercube since each node must communicate with the eight nodes that surround it in the decomposition topology, and the channels in a hypercube can only connect four of the eight directly. In order to transmit a message to a nonadjacent node, the transmitting node sends the message to an intermediate node, which forwards the message to its final destination.

Besides the difficulty of specifying the communication requirements in advance, the computational loads are not well balanced in the nodes between every communication step in the update algorithm. The amount of time taken to update a particle depends on the number of cells that must be searched for particles that contribute to the potential energy difference due to moving the particle. Since obtaining the contributions from some cells may involve communication with other nodes, the amount of time taken to update the particle also depends on the amount of communication required. Another source of load imbalance occurs because not all of the nodes contain the same number of particles since the particles are free to move between nodes. Fortunately, the difference in the number of particles is small in simulations near the melting transition, so we expect the differing number of particles to cause much less load imbalance than having the communication routines synchronize all of the nodes during each update.

In contrast to the regular applications for which the CrOS routines are adequate, efficiently programming *irregular* applications, such as the Lennard-Jones simulation, requires a more general communication system. Before introducing the more general communication system, we consider what features such a system should have so that irregular problems can be implemented efficiently. First, such a system should allow the nodes to run completely asynchronously; it should not force all of the nodes to proceed through their programs in a loose lockstep as the CrOS routines do. If the communication routines did not synchronize the nodes as the CrOS routines do, the computational loads would not have to be balanced between each communication step in the algorithm. Allowing the nodes to run completely asynchronously in the Monte Carlo update algorithm would eliminate the major source of load imbalance since the total computational load of each node is nearly identical. A general communication system should also allow unexpected communications so that the destination node does not have to know in advance that a message is coming. Finally, such a system should allow messages to be communicated between any pair of nodes in the concurrent processor as if communication channels connected all pairs of nodes.

We implement such a general communication system by using the interrupt lines from the communication channels to interrupt the processor, rather than by polling the lines as we did to implement the CrOS routines. When a communication interrupt occurs, the processor suspends its current task and executes a special routine called an *interrupt handler*, which then transmits or receives a packet depending on the type of interrupt that occurred. If the transmitting interrupt handler adds a header to each packet containing its destination, the receiving interrupt handler can forward a packet if its destination is another node. Since the interrupt handler can forward a packet towards its destination, messages can be routed between any pair of nodes without affecting the application program.

Although many interfaces between an *interrupt-driven operating system* (IDOS) and an application program are possible, any interface has two basic functions. First, the application program must be able to instruct the IDOS to send a message to a specific destination node. Second, it must be able to determine when a message has arrived and where it can be found. Since the node receives messages as they arrive, the application program does not need to anticipate the communications as it does when using the CrOS routines. In addition, an IDOS need not synchronize communicating nodes as the CrOS routines do, so application programs can run completely asynchronously. Since an IDOS is much more complicated than CrOS, we will not present the details of the system that we designed to implement the concurrent Monte Carlo update algorithm. Instead, we refer the reader to a detailed discussion of its design and implementation.[28]

Before we can run an application program on a concurrent processor, we must assign the decomposed pieces of the problem to the nodes of the concurrent processor. The assignment requires a mapping from the decomposition topology of the application onto the interconnection topology of the concurrent processor. As we mentioned previously, the hypercube contains grids of each dimensionality that is lower than the dimensionality of the hypercube, so the proper mapping can directly connect adjacent regions in such a decomposition topology with a communication channel. A routine that maps a hypercube onto a general Cartesian grid is part of a utility library that supports the communication routines.[29] Although communication channels directly connect adjacent nodes in the two-dimensional decomposition topology of the Lennard-Jones simulation, channels cannot directly link the diagonally connected nodes. However, when an application uses interrupt-driven communications, a communication channel does not need to link connected regions of the decomposition topology since the communication routines transparently route a message to its destination. Even though a random mapping from any decomposition topology onto a concurrent processor with any interconnection topology would work, the specific mapping is still important since it should minimize the average distance that a message must travel.

3.5 Concurrent Update Algorithm

As we have mentioned, the Monte Carlo update algorithm requires the calculation of the potential energy difference due to moving a particle. We calculate the difference by first summing the contributions from each particle within the range of the truncated Lennard-Jones potential for both the new and old positions of the particle being updated. However, in the concurrent update algorithm, some of the contributing particles may be in cells that are in other nodes, so a method of obtaining the contributions from other nodes is needed.

Since the cells correspond to fixed regions of the simulation volume, determining which of the eight surrounding cells are within the range of the potential is easy. Figure 3.1 illustrates the range of the potential with the large circles surrounding several of the particles. An efficient method of obtaining the contributions from particles in the cells that are within the range of the potential but reside in other nodes uses a request-response protocol. The protocol involves sending to each such cell a request containing the coordinates of the new and old positions of the particle being updated. A node that receives such a request responds by returning the sum of the contributions to both the old and the new potential energies from the particles in the indicated cell. While waiting for the responses, the node updating the particle sums the contributions from particles in its own cells. When all of the responses have arrived, the node calculates the difference in potential energy and applies the acceptance criteria to determine whether to accept or reject the move.

In the concurrent Monte Carlo update algorithm, all of the nodes update particles simultaneously using the same procedure. While waiting for other nodes to respond to its requests, a node responds to the requests that it receives from other nodes in addition to performing the necessary sums for its own update. However, a problem arises when a node receives a request whose response requires the contribution from the particle that the node is currently updating. When such a conflict occurs, the nodes that are involved must work together to resolve the problem correctly.

We examine the problem caused by such conflicts by first reviewing the discussion of the Monte Carlo procedure given in Section 2.2. While evaluating the difference in the potential energy due to moving a particle, all of the other particles in the system remain fixed. Thus, the Monte Carlo algorithm is inherently sequential since one update must completely finish before another can start. The system evolves from one state to another in a progression of states differing by the position of a single particle. Such a progression, known as a *Markov chain*, arises from the condition of detailed balance which we imposed to ensure that the distribution of states of the system approachs the canonical distribution. Breaks in the Markov chain of states, which can occur when conflicts are not resolved correctly, correspond to violations of detailed balance and result in an incorrect distribution of states. Although detailed balance requires a strictly sequential ordering of the updates, the specific sequential ordering is irrelevant; only the existence of such an ordering is essential.

Since an efficient concurrent implementation of the Monte Carlo procedure requires updating many particles simultaneously, establishing a sequential ordering of the updates can be difficult. However, if all of the particles being updated at the same time are well separated from each other, the concurrent algorithm satisfies detailed balance trivially. The finite range of the interaction potential ensures that particles separated from the particle being updated by more than the range of the potential have no effect on the acceptance of the move. Performing such a set of concurrent updates sequentially would give identical results regardless of the specific ordering of the updates. Unfortunately, the concurrent implementation cannot easily guarantee that only well-separated particles will be updated together since the nodes are running asynchronously. Thus, the concurrent algorithm must reliably detect and correctly resolve conflicts as they occur during the simulation.

Receiving a request whose response requires the contribution from the particle that the node is currently updating is the only method by which a node can detect a conflict. Fortunately, the radial symmetry of the Lennard-Jones potential ensures that all of the nodes involved in the conflict can detect it by this method. Figure 3.2 illustrates a conflict by showing the old and new positions of two particles in adjacent nodes and the range of the interaction potential about each of the particles' old positions. Resolving such a conflict and thereby preserving a sequential ordering of the updates require that one of the nodes delay its update until the other has finished. Of course, complicated conflicts involving many nodes can also arise, so resolving a conflict correctly may involve the coordination of many nodes.

When a node detects a conflict, it must choose between only two alternatives in order to resolve the conflict. It can either send the response immediately using the old position of the particle that it is updating, or it can delay sending the response until its current update finishes, and use the updated position of the particle. In the former case, the node is deciding that the other node's update should finish before its own, and in the latter case, it is deciding that its own update should finish before that of the other node. Each node involved in the conflict makes a similar decision and the result must be a consistent ordering of the conflicting updates.

If the decisions of all of the nodes involved in the conflict do not assign a consistent ordering to the updates, the concurrent Monte Carlo update algorithm will not simulate the Lennard-Jones system correctly. For instance, if two nodes both send responses to each other based on the old positions of the particles being updated and then accept both moves, the simulation violates detailed balance. In such a situation, one of the nodes based its acceptance of the move on an incorrect potential energy difference that did not account for the new position of the particle in the other node. On the other hand, if two nodes both decide to delay sending a response to each other until their own updates finish, neither node will receive a response to one of its requests. Since neither node is able to finish its update, *deadlock* occurs, which causes the simulation to halt abruptly. When all of the nodes involved in a conflict agree on a consistent ordering of their updates, the concurrent update algorithm both avoids deadlock and satisfies detailed balance.

A reliable method of resolving conflicts assigns a sequential ordering to the updates based on the order in which they started. But how can a node determine when another node initiated its current update? Since the interrupt-driven communication routines do not synchronize the nodes as the CrOS routines do, the nodes run completely asynchronously, which makes such a determination difficult. One method of providing information that allows arbitrarily complicated conflicts to be resolved correctly is to label each of the requests generated by an update with the starting time of the update. The label records the starting time of the update since the update as a whole, rather than the individual requests, must be assigned a consistent ordering when a conflict arises. Although the method works even if the clocks in all of the nodes are different, using a clock that is identical in all of the nodes balances the computational loads optimally. Such a global clock allows asynchronous processes to impose weak synchronizations, such as consistently ordering several conflicting updates, without reducing their efficiency as much as a stronger synchronization would.

Labeling each request with the time that its update started makes the method of resolving conflicts obvious. When a node detects a conflict, it sends a response immediately based on the old position of the particle that it is updating if the requesting node's update started before its own update. However, if the requesting node's update started after its own update, the node delays its response until its current update finishes and then uses the new position of the particle to calculate the response. However, since the clocks that generate the time labels have a finite resolution, the conflicting updates will occasionally start at the same time. In such a situation, a convenient method of resolving the conflict uses the unique processor number that labels each node. If the processor number of the requesting node is less than the processor number of the node that receives the request, it sends the response immediately; otherwise it delays it.

Using the starting times of the conflicting updates to order the updates results in an ordering that is always consistent, regardless of the number of nodes involved in the conflict. In addition to arguing that the method works, we can also reliably test whether the simulation actually resolves all conflicts correctly. If deadlock occurs when the simulation resolves a conflict incorrectly, the problem is immediately obvious because the concurrent processor abruptly halts during a Monte Carlo sweep. However, determining the exact sequence of events that led to deadlock is very difficult since deadlock results from events in several nodes whose interactions are very sensitive to the timing of the events. Thus, a strong *a priori* argument that deadlock cannot occur, such as the method that we just described provides, is almost essential for designing a concurrent algorithm that always avoids deadlock.

If the simulation incorrectly resolves a conflict by allowing several nodes to send responses based on the old positions of particles that they are updating, the simulation may violate detailed balance and give incorrect results. Although deadlock does not reveal the error, the simulation can detect a violation of detailed balance by updating the total potential energy of the system while it updates the particles. Since calculating the difference in the potential energy due to moving a particle is part of the update procedure, a node simply adds the difference to the current potential energy when it accepts a move in order to obtain the new potential energy. The requirement that the updates have a consistent sequential ordering ensures that the correct potential energy will result after many updates.

If the concurrent update algorithm resolves all conflicts correctly, the potential energy obtained by updating its value during the simulation will agree with the potential energy obtained by directly measuring its value when no particles are being updated. The only difference between the two values should be due to roundoff error, whose expected magnitude can easily be calculated. A difference that is significantly larger than the expected roundoff error reveals that the simulation used an incorrect energy difference in at least one of its updates and violated detailed balance. Thus, we can be very confident that the simulation is always resolving conflicts correctly if deadlock does not occur and the two methods of obtaining the total potential energy are always in agreement.

Although conflicts are clearly an undesirable situation because they require that a synchronizing condition be imposed on the asynchronous processes, they are not disastrous to the efficiency of the algorithm. While waiting for all of its responses to arrive, a node can respond to requests from other nodes and sum the contributions from its own particles to the potential energy difference due to moving the particle that it is updating. The node whose update will finish first during a conflict must delay sending its responses to the other nodes involved in the conflict, but it can still calculate the responses. In the current implementation of the simulation, the node calculates both possible responses, which differ only by the contribution of the particle being updated, so that when its update finishes, it can send the appropriate responses immediately. Thus, we see that resolving conflicts by requiring one node to finish its update before sending its responses to other nodes only slightly reduces the concurrency of the algorithm.

In Section 3.2 we mentioned that the concurrent update algorithm should sort the particles into their correct cells after updating all of the particles once, rather than sort each particle immediately after updating it. We can now examine why sorting the particles immediately is difficult in the concurrent algorithm. We begin by reviewing several important features that the concurrent algorithm should possess in order to run efficiently. The individual nodes should run asynchronously, so they do not have to wait for the slowest node to finish its current update before they start their next one. In addition, the nodes must correctly resolve conflicts as they arise so that the simulation avoids deadlock and satisfies detailed balance. If the update algorithm satisfies these three conditions, it cannot also move particles from one node to another while updating other particles, so it can sort the particles only after it finishes updating all of their positions.

The problem with moving a particle to a cell in another node while updates are occurring is that the update algorithm cannot guarantee that the particle being moved will be included in all of the necessary potential energy sums. To illustrate, we consider the following situation that can arise if the algorithm moves particles to their correct cells immediately after updating them. Moving a particle to a cell in another node requires that the node containing the particle remove it from its old cell and send a message to the other node, instructing it to add the particle to its new cell. If both nodes receive a request from a third node whose update requires the contribution from the particle being moved, a situation can arise in which neither node will include the contribution of the particle in their response. Since the nodes are running asynchronously, the node to which the particle is moving has no advance notice that the particle will arrive, so it may send its response to the third node before the particle arrives. However, the node that originally contained the particle may have sent the particle and removed it from its old cell before responding to the third node. Since it has already sent the particle to another node, it will not include the contribution from the particle in its response. The result is that neither of the nodes involved in moving the particle includes its contribution in the response that they send to the third node, so a violation of detailed balance occurs.

As long as the nodes are running asynchronously, the update algorithm cannot move particles between nodes while other particles are being updated and still always satisfy detailed balance and avoid deadlock. Since the node to which the particle is moving does not know in advance that the particle is coming, it cannot simply wait until it receives the particle to send responses that should include the particle. Likewise, the node sending the particle cannot know that the node receiving the particle has already sent responses that should have included the particle. In order to supply either node with the information needed to ensure that the update procedure satisfies detailed balance, the algorithm would have to synchronize the nodes in some way. A solution that does not significantly reduce the algorithm's efficiency is to sort the particles into their correct cells after updating all of their positions so that the algorithm needs to synchronize the nodes only after each complete Monte Carlo sweep.

Having discussed the concurrent algorithm for updating the positions of the particles in the system, we now examine the procedure for updating the volume of the system. In Section 2.4 we discussed that although a constant-density simulation requires updating only the particles' positions, a constant-pressure simulation also requires updating the volume of the system in order to maintain a constant average pressure. We effectively change the volume of the system by changing the Lennard-Jones length parameter, σ, which results in rescaling all of the particles' coordinates. In contrast to updating the position of a single particle, which has only a local effect, updating σ affects the interaction potential between all pairs of particles whose separation is less than the range of the truncated potential. In fact, updating σ even changes the range of the truncated Lennard-Jones potential since we specified the truncation distance in terms of σ. Because of the global effect of changing σ, the algorithm cannot update σ while performing concurrent updates of the particles' positions.

Since updating σ is an inherently sequential component of the constant-pressure update algorithm, we update σ only after each complete Monte Carlo sweep of the system. While we were able to distribute the computational load of updating the particles' coordinates rather uniformly among the nodes, we cannot do the same with the computation load of updating σ since it is a global parameter. Although any of the nodes could update σ, we generally assign to the control processor such components of an algorithm that do not fit into the symmetry of the concurrent processor. Since we cannot update σ concurrently with the particles' coordinates, the efficiency of updating σ is terrible. However, the overall loss of efficiency resulting from the sequential component in the algorithm is obviously small since it involves only one degree of freedom in contrast to the many degrees of freedom associated with the positions of the particles. Although Amdahl's Law[30] states that the sequential components of an algorithm place an upper limit on the number of useful nodes, our 64 node concurrent processor is nowhere near the limit in this case. Since we typically use a larger concurrent processor to run a larger problem, rather than to run the same problem faster, we usually do not see the effect of Amdahl's Law.[31]

3.6 Irreproducibility of Simulation Results

An interesting feature of the concurrent Monte Carlo update algorithm is that it gives irreproducible results. The irreproducibility arises because the algorithm does not impose a specific ordering on the sequence of particle updates. As we discussed in the previous section, the algorithm assigns a definite ordering only when one is necessary to resolve a conflict. While detailed balance requires that a consistent sequential ordering of the updates exist, it does not require any specific ordering. The algorithm incorporates no specific ordering since imposing a predetermined ordering would require synchronizing the nodes in some way, which would reduce the efficiency of the algorithm.

Still, a computer is a deterministic machine, so how can an algorithm give irreproducible results? The answer is that while any node of the concurrent processor is deterministic in isolation, its interaction with the other nodes may not be. Even though all of the nodes comprising the Mark I machine use the same clock signal, each of the communication channels connecting the nodes contains an asynchronous component, which allows a type of race condition to occur. Since the update algorithm is sensitive to the race condition, it can affect the ordering of updates in the simulation and lead to irreproducible results.

In order to illustrate the sensitivity of the update algorithm, we first examine the effect that a slight change in the arrival time of a request can have. If a node receives a request before finishing its current update, it generates a response to the request while waiting to receive the responses for its update. The node may send the response immediately or delay it in order to resolve a conflict, but the node will always send the response before starting to update another particle. However, if a request arrives just after the node receives all of the responses for its update, the node will not respond to the request until it begins waiting to receive the responses for its next update. Thus, depending on whether the node receives the request before or after receiving the last response that it requires to complete an update, it may send the response promptly or after a considerable delay. A delay in sending the response could allow updates in other nodes to finish before the one in the node that sent the request, which could change the ordering of the updates and affect the acceptance of subsequent updates. Such a situation illustrates one way in which a race condition involving two messages can affect the acceptance of an update.

Since the algorithm does not impose a specific ordering on the updates in the simulation, situations that are sensitive to the arrival times of messages can arise in complicated ways. However, before such a situation can affect the simulation, the arrival time of a critical message must actually change by the required amount. A synchronous system is not sensitive to small changes in the timing of events because a system clock regulates all timing. In contrast, an asynchronous system does not have a clock to regulate the timing of events, so the relative timing of several events can differ by arbitrarily small amounts. Since part of the circuit forming the communication channels is asynchronous, the arrival times of messages can vary slightly because of many effects, which can change the order in which messages on different channels arrive. The sensitivity of the algorithm to the order in which messages arrive and the possibility of changes occurring in that order produce an algorithm whose results are inherently irreproducible.

Thus, if we run a simulation twice using exactly the same initial data, including the random number seeds, the results of the two simulations will eventually differ. Once the decision of whether or not to accept a single move differs between the two simulations, the Markov chain of states describing the simulation bifurcates. The simulations then proceed along completely different paths as the effect of changing one update reaches all of the other particles in the system. We have observed the bifurcation of the Markov chain in simulations that gave identical results for many Monte Carlo sweeps before abruptly differing. We note emphatically that the inherent irreproducibility of the simulation results in no way indicates that the concurrent update algorithm is incorrect. It merely illustrates that the specific sequential ordering of the updates is irrelevant. If we were to implement a different concurrent algorithm in which a specific ordering of events was important, the algorithm would have to be insensitive to small changes in the arrival times of messages so that such irreproducibility could not arise.

3.7 Efficiency of Concurrent Update Algorithm

Having developed an intuitive sense that an algorithm is efficient when it uses most of the nodes for useful computation most of the time, we now formalize the concept of efficiency. A formal definition of efficiency allows us to measure the efficiency of a concurrent algorithm and to compare its effectiveness to that of other concurrent algorithms. In addition, we can analyze a concurrent algorithm in order to calculate its expected efficiency and obtain its dependence on parameters such as the number of degrees of freedom per node.

We measure the effectiveness of a concurrent algorithm by comparing its performance to that of its sequential counterpart. The increase in the computational speed of a concurrent algorithm is its *speedup*, S, which is defined as

$$S = \frac{T_{seq}}{T_{par}} \tag{3.1}$$

where T_{seq} is the time taken by the sequential algorithm and T_{par} is the time taken by the parallel algorithm. In order for the comparison to be meaningful, we assume that the sequential machine has the same computing power as one of the nodes in the concurrent machine and enough memory to run the algorithm. We define the efficiency, ϵ, as the average fraction of the time that each node performs useful computation. In terms of S, the efficiency is

$$\epsilon = \frac{S}{N} \tag{3.2}$$

where N is the number of nodes in the concurrent processor.

Four effects can reduce the efficiency of a concurrent algorithm from its ideal efficiency of $\epsilon = 1$. In the discussion of the update algorithm we mentioned that communication overhead, load imbalance, and sequential components reduce the concurrent algorithm's efficiency. We account for these three effects by writing T_{par} as

$$T_{par} = t_{calc} + t_{comm} + t_{wait} + t_{seqc} \tag{3.3}$$

where t_{calc}, t_{comm}, and t_{wait} are the average amounts of time that each node spends calculating, communicating, and waiting on other nodes, respectively. The total amount of time that is required to perform the sequential components of the concurrent algorithm is t_{seqc}.

The fourth effect, which is not present in the update algorithm, occurs when the concurrent algorithm is fundamentally different from the sequential algorithm. The concurrent implementation of an application may use a different algorithm because some algorithms that work well on a sequential computer cannot be decomposed so that they run efficiently on a concurrent processor. For instance, the *quicksort* algorithm, which is often the optimal algorithm for sorting on a sequential machine, does not perform well on a concurrent processor, so other algorithms are used.[32] In order for the efficiency of an algorithm to be a meaningful measure of its performance, using the best algorithm for each machine is essential when evaluating the relative performance of the concurrent algorithm. Using a sequential algorithm that is not optimal could result in efficiencies that are greater than one, but they would have little meaning.

We now express the efficiency of an algorithm in terms of the effects that reduce its efficiency. We first consider an algorithm in which the only significant effect reducing its efficiency is communication overhead, allowing us to neglect t_{wait} and t_{seqc}. In addition, we assume that the concurrent and sequential algorithms are basically the same algorithm so that $T_{seq} = N\, t_{calc}$. Using (3.2) and (3.3), we obtain

$$\epsilon = \frac{t_{calc}}{t_{calc} + t_{comm}} \tag{3.4}$$

which shows that ϵ is the fraction of the time that the nodes spend performing calculation. If we extend (3.4) to include the effect of sequential components in the concurrent algorithm, we have $T_{seq} = N\, t_{calc} + t_{seqc}$, which gives

$$\epsilon = \frac{t_{calc} + \frac{1}{N} t_{seqc}}{t_{calc} + t_{comm} + t_{seqc}} \ . \tag{3.5}$$

Although we could generalize (3.5) to include the effects of load imbalance by adding a term containing t_{wait}, as we did with t_{seqc} in (3.5), we will use a different method. The load balance of an algorithm is the average fraction of the time that the nodes spend performing communication or calculation instead of idly waiting on other nodes, so we express its effect as that fraction. The efficiency of an algorithm in which we must account for load imbalance is

$$\epsilon = \kappa \frac{t_{calc} + \frac{1}{N} t_{seqc}}{t_{calc} + t_{comm} + t_{seqc}} \tag{3.6}$$

where we define κ as

$$\kappa = \frac{t_{calc} + t_{comm} + t_{seqc}}{t_{calc} + t_{comm} + t_{seqc} + t_{wait}} \ . \tag{3.7}$$

As a simple example of calculating κ, we consider an algorithm that consists of a period of computation in which no idle time occurs, followed by a synchronizing operation, such as the CrOS routines provide. We assume that the algorithm contains no sequential components, so we can neglect t_{seqc}. If the period of computation takes longer in some nodes than in others, the algorithm has a load imbalance, and the other nodes must wait for the slowest node when they reach the synchronizing operation. In such a situation, $(t_{calc} + t_{comm})$ is the average amount of time that each node takes to finish its computation and $(t_{calc} + t_{comm} + t_{wait})$ is the amount of time that the slowest node takes to finish. The latter expression is the total execution time of the concurrent algorithm, so at least one node must have taken that long to finish its computation. Although the terms in (3.7) may not have such a simple interpretation in a more complicated situation, κ correctly accounts for the effect of load imbalance, regardless of how it arises.

The final effect that we must include in a general expression for the efficiency of an algorithm is the effect that arises when the concurrent algorithm is fundamentally different from the sequential one. As with the effect of load imbalance, we account for the algorithmic effect with an overall multiplicative factor. We define α as

$$\alpha = \frac{T_{seq}(optimal)}{T_{seq}(concurrent)} \tag{3.8}$$

where $T_{seq}(optimal)$ is the time taken by the optimal sequential algorithm and $T_{seq}(concurrent)$ is the time taken by the sequential algorithm on which we based the concurrent algorithm. In addition to fundamental differences in the concurrent and sequential algorithms, extra indexing or looping overhead in the concurrent algorithm can reduce α from its ideal value of 1. Inserting α into (3.6) gives the general expression for the efficiency an algorithm, which is

$$\epsilon = \alpha\kappa \frac{t_{calc} + \frac{1}{N} t_{seqc}}{t_{calc} + t_{comm} + t_{seqc}} \tag{3.9}$$

Of course, we do not intend to imply that the quantities in (3.9) are independent of each other. For instance, the sequential components of an algorithm could be considered a load imbalance since one node must perform more computation than the others. However, conceptually separating the effects that reduce the efficiency of an algorithm is useful since it allows a better understanding of the algorithm, which may lead to improvements in it.

We can estimate the efficiencies of many concurrent algorithms with (3.9) before we implement them by measuring the characteristic times of calculation and communication and counting the number of such operations in the algorithm. In many algorithms, including the Monte Carlo update algorithm, we know that $\alpha = 1$ and that t_{seqc} is negligible. Although estimating κ is simple in some algorithms, we cannot calculate the load balance in the update algorithm by simply examining the algorithm since the asynchronous interaction of many nodes determines κ as the algorithm runs. However, we can obtain the efficiency of the update algorithm by measuring T_{seq} and T_{par} directly and using (3.1) and (3.2).

We obtain values of T_{seq} and T_{par} for systems of various sizes by measuring the average length of time that 20 Monte Carlo sweeps take. In order to be consistent, each measurement uses the same starting configuration, which is a perfect triangular lattice. Since the update algorithm can be run on only one node, we can easily measure values of T_{seq}. Although memory limitations restrict each node to a maximum of 64 cells, which corresponds to a total of 1024 particles, the linear dependence of T_{seq} on n, the average number of particles per node, allows us to extrapolate T_{seq} reliably to larger values of n. We report the results of the timings in Table 3.1, whose first three columns give the configuration of nodes, the average number of particles per node, and the average time per sweep in seconds at $T^* = .7$ and $\rho^* = .86$. The fourth column of Table 3.1 reports the efficiency of the update algorithm. Since we cannot estimate κ, which appears in the fifth column in Table 3.1, by simply examining the update algorithm, we will use the measured efficiencies from Table 3.1 and (3.9) to calculate κ. However, before we can use (3.9) to calculate κ, we must first know t_{calc} and t_{comm}. Since $N\,t_{calc} = T_{seq}$ when $\alpha = 1$, we can use the extrapolated values of T_{seq} to obtain t_{calc}.

Table 3.1

Efficiency of Monte Carlo Update Algorithm

Nodes	n	Time	ϵ	κ
1 × 1	64	2.54	1.000	1.000
1 × 1	256	10.14	1.000	1.000
1 × 1	1024	40.56	1.000	1.000
2 × 2	16	1.26	.504	.753
2 × 2	64	3.92	.647	.825
2 × 2	256	13.57	.747	.855
2 × 2	1024	47.95	.846	.908
4 × 4	16	1.29	.491	.735
4 × 4	64	3.95	.642	.819
4 × 4	256	13.33	.761	.871
4 × 4	1024	47.64	.851	.914
8 × 8	16	1.33	.477	.713
8 × 8	64	4.00	.634	.808
8 × 8	256	13.52	.750	.858
8 × 8	1024	48.04	.844	.907

Obtaining an accurate value of t_{comm} is more difficult because we cannot measure t_{comm} directly, as we can t_{calc}. Instead, we measure the characteristic time of the basic communication operation, which consists of sending a request and receiving a response, and we calculate the expected number of such operations during a Monte Carlo sweep. Measuring the communication speed of the interrupt-driven communication system under several different conditions allows us to calculate the amount of time that the various stages of the communication system take. From these times, we calculate the total amount of time involved in sending a request, which includes the time that other nodes spend receiving and forwarding the request. Since sending a response is nearly identical to sending a request, we assume that the two take the same amount of time. From these measurements, we obtained $t_1 = 3.168\,ms$ and $t_2 = 4.454\,ms$, where t_1 and t_2 are the characteristic times of request-response pairs that travel a distance of one step and two steps, respectively.

In order to calculate the expected number of requests per node during a Monte Carlo sweep, we refer to the cell that Figure 3.3 illustrates, in which the x and y dimensions of the cell are L_x and L_y, respectively. In order to obtain the contributions to the potential energy difference due to moving a particle in the cell, the update algorithm sends a separate request to each cell that is within $\mu\sigma$ of the particle if the cell resides in another node. As we discussed in Section 3.2, the distance $\mu\sigma$ is the sum of the range of the potential and twice the maximum distance that a particle can move during an update, so we obtain $\mu\sigma = 3\sigma + \Delta$. Updating a particle that lies above the horizontal dashed line in Figure 3.3 requires sending a request to cell A if it resides in another node. Similarly, updating a particle that lies to the right of the vertical dashed line requires sending a request to cell B if it resides in another node. If the particle lies both above the horizontal line and to the right of the vertical line, a request must also be sent to cell C if it resides in another node. In the last case, some of the particles in the specified region may not be within $\mu\sigma$

of cell C, but we use the indicated boundaries to avoid floating-point computation, which is much slower than integer computation on our concurrent processor.

We now calculate the average number of requests that a node sends while updating each of the particles in a cell once. We assume that the three neighboring cells, A, B, and C, reside in other nodes and we treat each one separately. Since the average number of particles per cell is 16, the average number of requests sent to cell A is

$$\nu_A = 16 \frac{\mu\sigma L_x}{L_x\, L_y} \tag{3.10}$$

in which the number of particles is multiplied by the fraction of the cell's area that is above the horizontal dashed line in Figure 3.3. Since we specify the reduced density, ρ^*, rather than σ in a simulation, we rewrite (3.10) in terms of ρ^* using $\rho^* = \frac{16\sigma^2}{L_x\, L_y}$, which gives

$$\nu_A = 4\mu\sqrt{\rho^*}\,(\frac{L_x}{L_y})^{\frac{1}{2}} \ . \tag{3.11}$$

Similarly, the calculations for regions B and C respectively give

$$\nu_B = 4\mu\sqrt{\rho^*}\,(\frac{L_y}{L_x})^{\frac{1}{2}} \tag{3.12}$$

$$\nu_C = \mu^2\rho^* \ . \tag{3.13}$$

We can use ν_A, ν_B, and ν_C to calculate the average number of requests that a node sends during an update sweep of a cell since all of the cells that surround it are similar to one of the labeled cells in Figure 3.3. The average number of requests depends on which of the cell's edges face cells that reside in other nodes. Since cells that reside in other nodes may face 0, 1, or 2 edges of a cell along both the x and y directions, we must treat each of the nine possibilities separately. We label the possible cases with n_x and n_y, which are the numbers of adjacent cells that reside in other nodes along the x and y directions, respectively. Requests sent to cells A and B are always one-step requests since the hypercube topology can be mapped onto a two-dimensional grid so that communication channels connect adjacent regions of the grid. However, since channels cannot also connect regions along a diagonal in the grid, requests that a node sends to cell C are two-step requests if cells A and B both reside in other nodes. We summarize the number of one-step and two-step requests that are required for the various combinations of n_x and n_y in Tables 3.2 and 3.3, respectively.

Table 3.2

Number of One-Step Requests per Cell				
			n_x	
		0	1	2
n_y	0	0	$\nu_1 + 2\nu_3$	$2\nu_1 + 4\nu_3$
	1	$\nu_2 + 2\nu_3$	$\nu_1 + \nu_2 + 2\nu_3$	$2\nu_1 + \nu_2 + 2\nu_3$
	2	$2\nu_2 + 4\nu_3$	$\nu_1 + 2\nu_2 + 2\nu_3$	$2\nu_1 + 2\nu_2$

Using the information in Tables 3.2 and 3.3, we obtain the average number of requests sent by a node during a Monte Carlo sweep of all of its cells for several different configurations of cells. Table 3.4 reports the total number of one-step and two-step requests, n_1 and n_2, respectively. Since we chose the dimensions of a cell so that it would accommodate 16 particles arranged in a triangular lattice, we have $\frac{L_x}{L_y} = \frac{2}{\sqrt{3}}$. In

Table 3.3

Number of Two-Step Requests per Cell

			n_x	
		0	1	2
	0	0	0	0
n_y	1	0	ν_3	$2\nu_3$
	2	0	$2\nu_3$	$4\nu_3$

addition, we use $\rho^* = .86$ and the nominal value of $\mu = 3.25$ to obtain $\nu_A = 11.22$, $\nu_B = 12.95$, and $\nu_C = 9.08$. Substituting these values into the entries in Table 3.4 and using the characteristic times t_1 and t_2, we obtain the values of t_{comm} reported in Table 3.5. Finally, we substitute the values of t_{comm}, t_{calc}, and ϵ into (3.9) to calculate the values of κ reported in the last column of Table 3.1.

We see from Table 3.1 that the efficiencies range from slightly under 50% for the smallest simulations to about 85% for the largest simulations. The values of κ for small n show a slight decrease as the number of nodes increases, but the effect disappears for larger n. The concurrent algorithm does a respectable job of balancing the computational loads of the nodes with κ ranging from about 70% to 90%. As we expect, the load balance improves with increasing n since the fluctuations in the computational loads average better.

Although the request-response protocol introduces a potential instability into the load balance, the data in Table 3.1 provide no evidence of such an instability. The

Table 3.4

Total Number of Requests per Node		
Cells	n_1	n_2
1×1	$2\nu_1 + 2\nu_2$	$4\nu_3$
2×2	$4\nu_1 + 4\nu_2 + 8\nu_3$	$4\nu_3$
4×4	$8\nu_1 + 8\nu_2 + 24\nu_3$	$4\nu_3$
8×8	$16\nu_1 + 16\nu_2 + 56\nu_3$	$4\nu_3$

Table 3.5

Total Communication Time per Node			
Cells	n_1	n_2	t_{comm}
1×1	48.3	36.3	0.315
2×2	169.4	36.3	0.698
4×4	411.4	36.3	1.465
8×8	895.5	36.3	2.999

instability could occur when some nodes fall slightly behind the others by spending more time responding to requests. If the nodes that sent the requests quickly send another batch of requests as they start updating their next particles, the nodes responding to the requests might not be able to update their own particles and could fall further behind. At the end of the Monte Carlo sweep, the nodes that finished sooner would have to wait for the slower nodes to finish, thereby reducing κ. If the algorithm exhibited such an instability, we would not expect κ to increase as n increased.

In conclusion, we see that the concurrent Monte Carlo update algorithm is reasonably efficient, even for small n. Although we could not calculate κ, and thus the efficiency, by analysis of the algorithm alone, we were able to obtain values of κ indirectly. In addition, our measurements and calculations of the various parameters that affect the efficiency gave us a quantitative understanding of the dependence of the algorithm on both the number of nodes and the average number of particles per node.

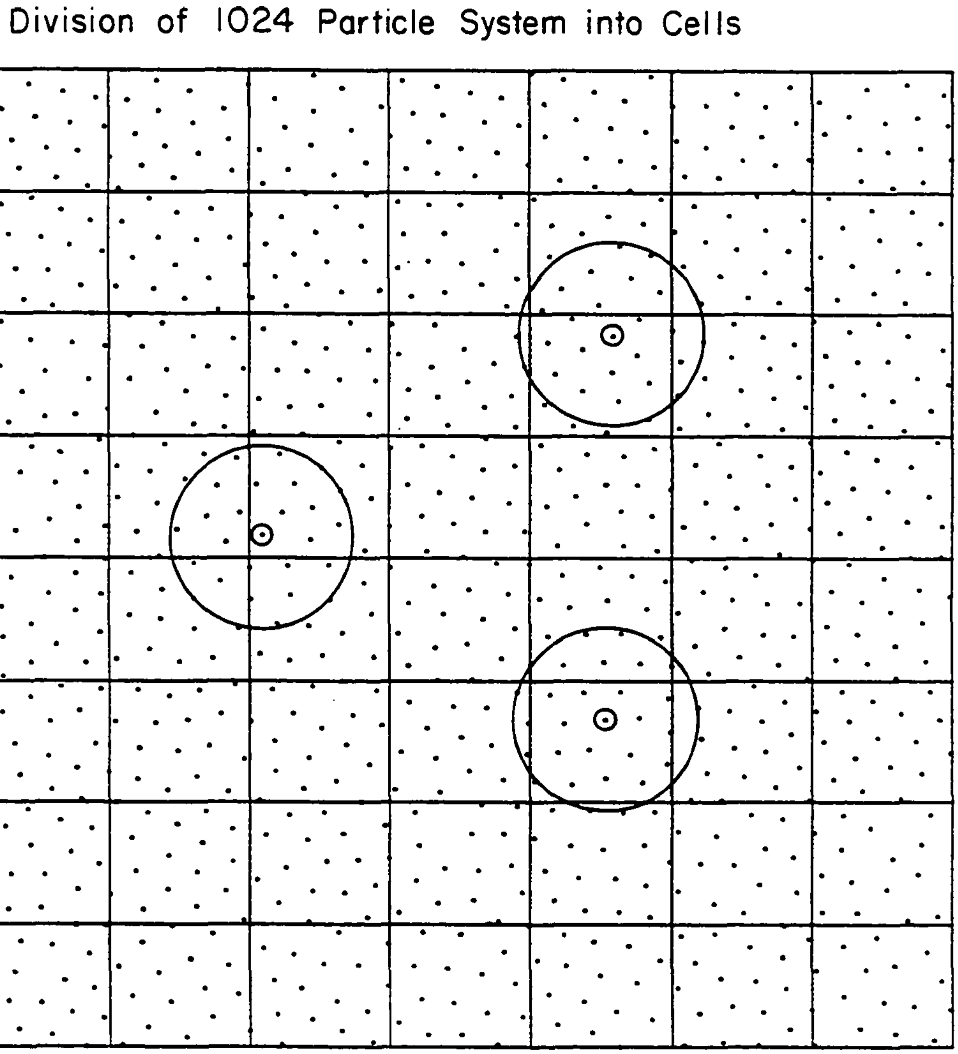


Figure 3.1 A system containing 1024 particles that consists of an 8 × 8 grid of cells. The large circles illustrate the 3σ range of the potential about the particles indicated by the smaller circles.

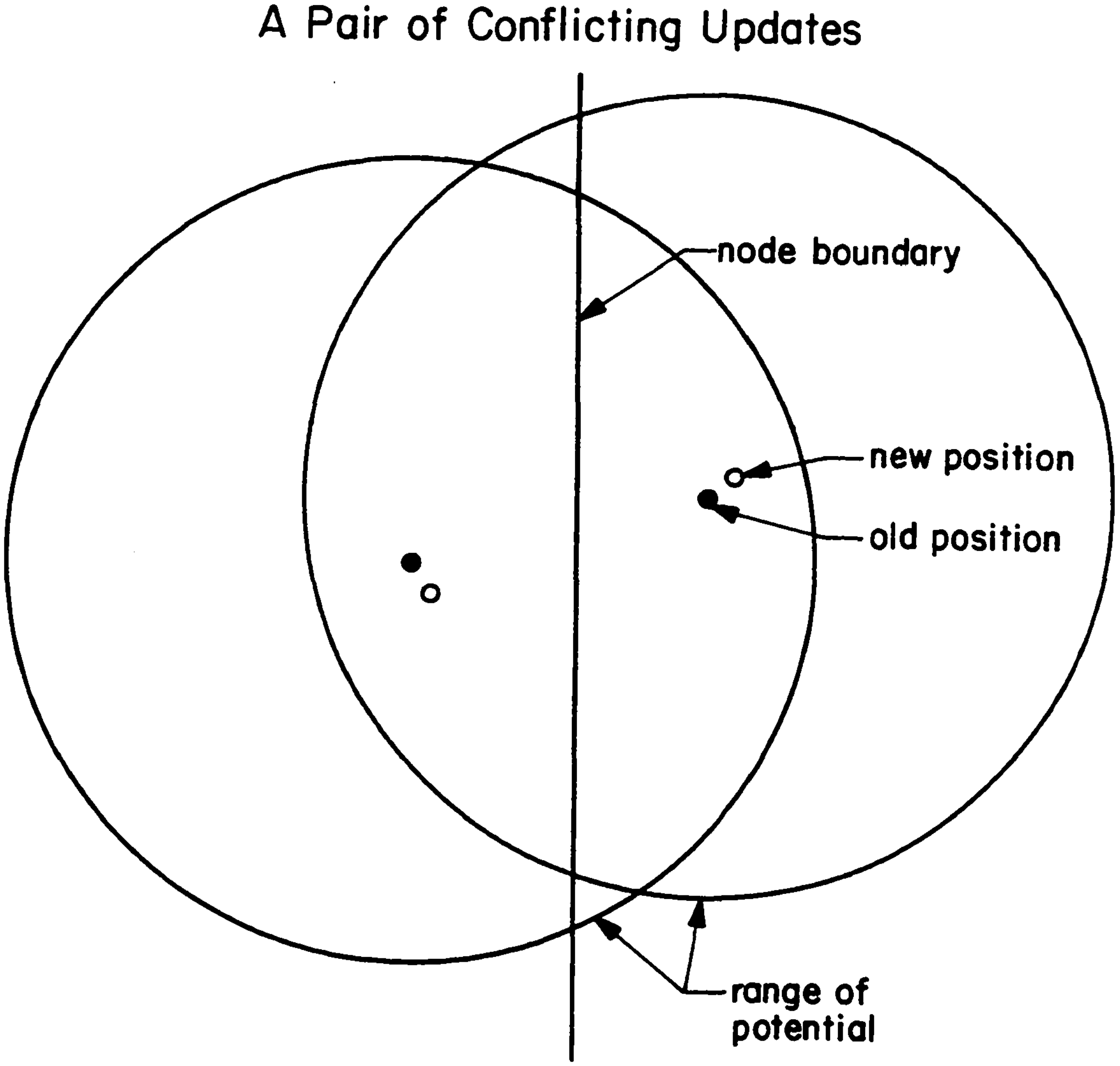


Figure 3.2 A conflict involving two particles that adjacent nodes are updating simultaneously. Since the ranges of the interaction potential about the particles overlap, the updates cannot occur independently of each other.

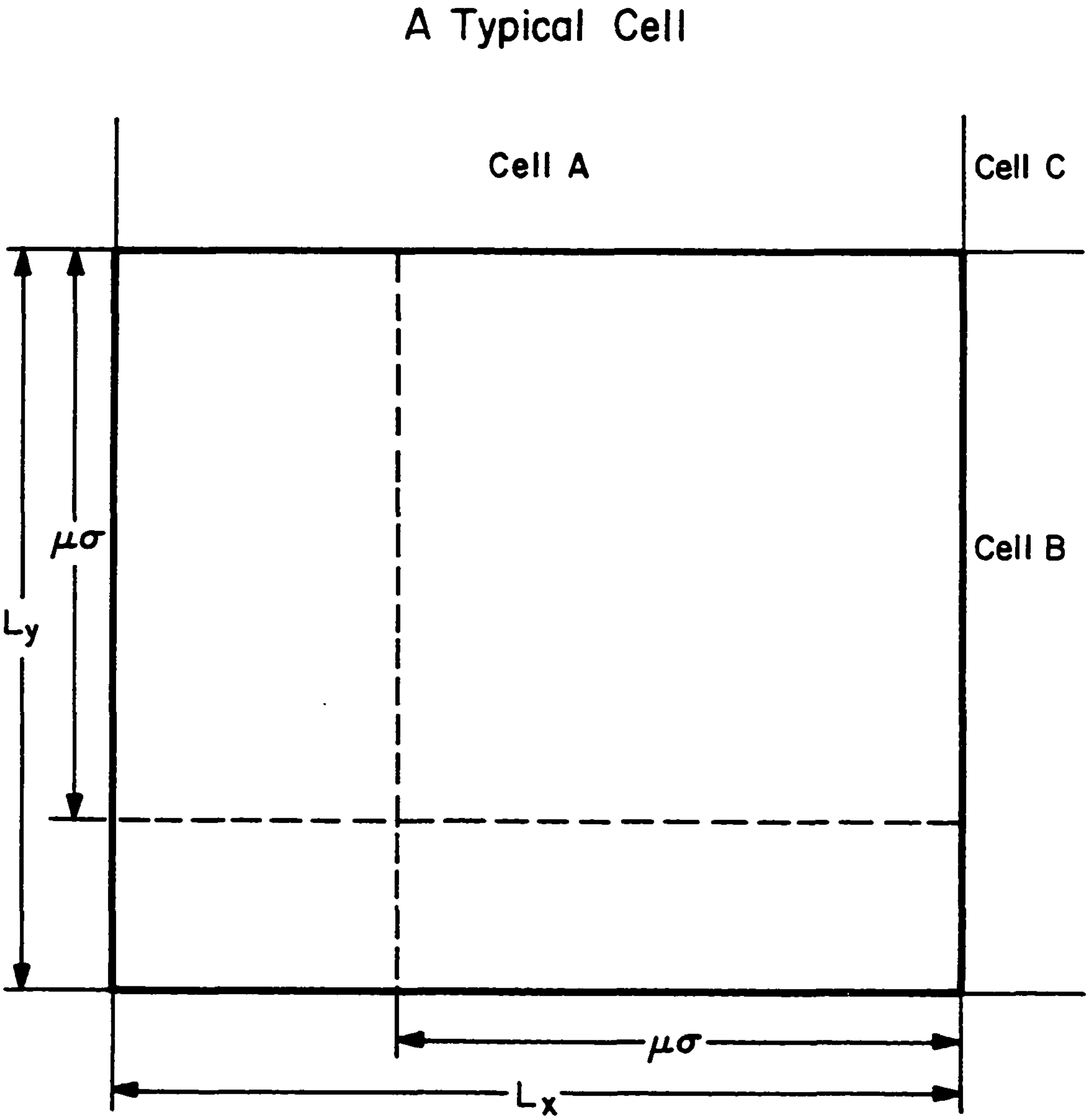


Figure 3.3 A diagram of a typical cell and the three types of neighbors that surround it. The dashed lines indicate the regions of the cell that are within $\mu\sigma$ of its edge. Updating particles that lie within these regions requires information from the neighboring cells.

Chapter 4: Simulations of Solid and Fluid Regions

4.1 Measurement of Potential Energy and Pressure

Since we have introduced the Lennard-Jones system and described the implementation of the update algorithm that simulates the system, we now discuss simulating of the system and measuring some of its properties. Constant-density simulations, in which we specify the temperature and the density of the system, generated the results that we report in this chapter. Two important quantities that we measure in such simulations are the potential energy and pressure of the system. From (2.13), the potential energy of the system in reduced units is

$$U^* = 4[\sigma^{12}\langle r^{-12}\rangle - \sigma^6\langle r^{-6}\rangle] \tag{4.1}$$

where (2.14) expresses $\langle r^{-12}\rangle$ and $\langle r^{-6}\rangle$ as a sum of the indicated quantity over all pairs of particles divided by N, the total number of particles in the system. As we mentioned in Section 3.5, the sums in (4.1) do not need to be recalculated for each measurement since the update algorithm can easily adjust their values as it updates the particles.

Before we can measure the pressure of the system, we must derive a formula that expresses the pressure of the system as a function of the particles' coordinates and the temperature and density of the system. Although we can use several methods to derive an expression for the pressure, we present only one of them here. In addition, we give two other derivations, which are based on the partition function of the constant-density system and the virial theorem, in Appendices A and B, respectively. The derivation that we present here uses the exponent of the Boltzmann factor in the partition function for the constant-pressure simulations, which is given in Section 2.4 as

$$W = U + pV - kTN \ln V \ . \tag{2.18}$$

Since updating the volume of the system maintains an average pressure of p, W is minimized when the pressure of the configuration equals the specified simulation pressure p. Thus, we obtain an expression for the pressure of a configuration by setting

$$\left[\frac{\partial W}{\partial V}\right]_{config} = 0 \tag{4.2}$$

and solving for p. The subscript *config* indicates that the relative positions of the particles are held constant; only an overall scaling factor changes during differentiation. Substituting (2.18) into (4.2) and solving for p gives

$$p^* = -\frac{\partial U^*}{\partial V^*} + \rho^* T^* \tag{4.3}$$

which we have converted to reduced units.

Since we effectively change the volume of the system by varying the length parameter σ, we rewrite the derivative in (4.3) in terms of σ. Using $V^* = \frac{V}{N\sigma^2}$, we obtain

$$\frac{\partial V^*}{\partial \sigma} = -2\frac{V^*}{\sigma} \ . \tag{4.4}$$

Applying the chain rule and (4.4) to (4.3) gives

$$p^* = \frac{\rho^* \sigma}{2}\frac{\partial U^*}{\partial \sigma} + \rho^* T^* \tag{4.5}$$

where $\rho^* = \frac{1}{V^*}$. Substituting (4.1) into (4.5) gives the final expression for the pressure of a configuration:

$$p^* = 12\rho^* [2\sigma^{12}\langle r^{-12}\rangle - \sigma^6\langle r^{-6}\rangle] + \rho^* T^* \ . \tag{4.6}$$

We see that the pressure of a configuration depends on the same sums as the potential energy, so we can measure the pressure of the system as easily as its potential energy.

As we described in Section 2.2, the expectation value of a quantity is its average over many configurations. In addition to calculating the expectation value of each measurement, we would also like to extract an estimate of its statistical error. However, we cannot use the normal expression for the standard deviation of a mean, which assumes that the set of measurements is uncorrelated.[53] Successive values of the pressure and potential energy are correlated since each Monte Carlo sweep changes the configuration by only a small amount. Calculating the standard deviation of a mean from a set of n correlated measurements does not reduce its standard deviation from that of the individual measurements by the normal factor of $n^{-\frac{1}{2}}$ since the n measurements are not independent. In order to calculate the true standard deviation of the average, we must account for the effect of the correlated measurements.

An easy method of obtaining an error estimate of the mean of a correlated set of measurements involves grouping the measurements into blocks of equal size. Averaging the measurements in each block gives a set of blocked measurements, which we use to calculate the standard deviation of the mean. Although the individual blocked measurements have a smaller standard deviation than the original measurements, the number of blocked measurements is also smaller. If the original set of measurements were uncorrelated, the effects would balance exactly, so the standard deviation calculated with the blocked measurements would be identical to that calculated with the original measurements. However, since the set of original measurements is correlated, the improvement in the standard deviation of the individual blocked measurements does not balance the reduction in the number of such measurements, so calculating the standard deviation of the mean with the blocked measurements gives a larger value. If we increase the number of measurements in each block, the standard deviation that is calculated with the blocked measurements increases as long as the set of blocked measurements is correlated. For sufficiently large blocks, the blocked measurements are nearly uncorrelated, so we obtain a true estimate of the statistical error of the mean. In addition, the block size at which the standard deviation of the mean stops increasing gives a rough estimate of the sweep-to-sweep correlation length of the measurement.

4.2 Corrections for the Truncated Potential

In Section 2.3 we discussed truncating the Lennard-Jones potential at a distance of 3σ in order to reduce the amount of computation required to calculate the potential energy differences during an update. We argued that although using the truncated potential should have only a small effect on the evolution of the simulation, it would affect some of the measurements, such as the potential energy and pressure. Since the effects of the truncated potential enter through the values of $\langle r^{-12} \rangle$ and $\langle r^{-6} \rangle$, we begin by breaking the $\langle r^{-n} \rangle$ into the two pieces

$$\langle r^{-n} \rangle_{true} = \langle r^{-n} \rangle_{<3\sigma} + \langle r^{-n} \rangle_{>3\sigma} \qquad (4.7)$$

where (2.14) defines $\langle r^{-n} \rangle$. The first term on the right side of (4.7) is the value obtained from the Monte Carlo simulation and the second term is the correction term that we would like to evaluate.

A simple method of correcting for the truncated potential assumes a uniform distribution of particles outside the 3σ cutoff. Calculating the resulting corrections to the values of $\langle r^{-n} \rangle$ gives the integral

$$\langle r^{-n} \rangle_{>3\sigma} = \frac{1}{2} \int_0^{2\pi} \int_{3\sigma}^{\infty} \rho r^{-n} \, r \, dr \, d\theta \qquad (4.8)$$

where ρ is the uniform density and $n \geq 3$. The factor of one-half arises from counting the interaction potential between each pair of particles twice in the integral. Integrating (4.8) gives

$$\langle r^{-n} \rangle_{>3\sigma} = \frac{\pi\rho}{n-2} (3\sigma)^{-(n-2)} . \tag{4.9}$$

Rewriting (4.9) in reduced units and substituting 6 and 12 for n gives

$$\langle r^{-6} \rangle_{>3\sigma} = \frac{1}{324} \pi\rho^{*}\sigma^{-6} \tag{4.10}$$

and

$$\langle r^{-12} \rangle_{>3\sigma} = \frac{1}{590490} \pi\rho^{*}\sigma^{-12} . \tag{4.11}$$

However, a uniform distribution of particles outside the 3σ cutoff is only an approximation to the true distribution of particles. Although the uniform distribution is a very good approximation for particles in an ideal gas, it is a poor approximation for particles in a crystal lattice. Thus, we expect that the approximation is better at lower densities than it is at higher densities. Since the attractive component of the potential gives the particles a tendency to clump together, we expect that the true values of $\langle r^{-12} \rangle$ and $\langle r^{-6} \rangle$ are larger than the uniform density approximation predicts. The value of $\langle r^{-12} \rangle$ is relatively insensitive to particles beyond 3σ compared to the value of $\langle r^{-6} \rangle$, so the true values of the potential energy and pressure should be somewhat lower than those given by the uniform density approximation. However, at high densities the nearly hard core of the potential dominates the interaction between particles, inhibiting the clumping due to the attractive component of the potential, so the previous argument does not apply.

A measure of the system that is related to the distribution of particles is the two-point correlation function. If we knew the two-point correlation function, we could extract from it the average density outside the 3σ cutoff and use the true average density in (4.8). By using the true distribution of particles, the calculation of the corrections would involve no approximations, so we would obtain the correct values of the potential energy and pressure. However, the two-point correlation function contains much more information than we need to calculate the corrections for the truncated potential since we simply integrate over all the details of the correlation function. Rather than measuring the two-point correlation function and extracting the corrections, we measured the corrections directly. Using a few configurations from each simulation density, we calculated the contributions to $\langle r^{-12} \rangle$ and $\langle r^{-6} \rangle$ from the particles outside the 3σ cutoff. Subtracting the correction obtained with the uniform density approximation gives the residual correction. Since the residual correction is small and relatively insensitive to the specific configuration, we easily obtained values whose errors were sufficiently small. We discuss actual measurements of the residual corrections and plot their values as a function of density in Section 4.4.

4.3 Thermalizing the System

Before we can obtain any meaningful measurements from a simulation, we must *thermalize* the system in order to bring it into equilibrium. Minimizing the number of Monte Carlo sweeps needed to thermalize the system requires that the initial configuration be chosen carefully, especially in the solid regime and near phase transitions, where the system may exhibit strong metastability. In the solid regime, an appropriate starting configuration is a *cold start* in which the particles are arranged in a regular crystal lattice. For the two-dimensional Lennard-Jones system, the lowest energy crystal lattice is the triangular lattice, in which all of the particles have six equidistant nearest neighbors. At low temperatures and high densities, we expect the deviations of the particles from the lattice sites to be small, so the cold start provides a good initial configuration for such simulations.

Another good choice of an initial configuration is one that is in equilibrium at a nearby temperature or density. Since the system approaches equilibrium faster when its entropy is increasing, configurations that are in equilibrium at slightly higher densities or lower temperatures usually thermalize fastest. Often, the entropy changes only a small amount during thermalization, so the thermalizing time is rather insensitive to whether the entropy is increasing or decreasing. However, in some situations the change in entropy during thermalization may be much larger, making the thermalizing time very sensitive to the choice of an initial configuration. For instance, in simulations near a phase transition the difference in entropy between two configurations may be large, even when they are in equilibrium at only slightly different temperatures. In Chapter 6 we will illustrate the importance of using a good initial configuration when we discuss simulations of the transition region that are very sensitive to their initial configurations.

Besides the choice of an initial configuration, another factor that affects the thermalizing time of a simulation is the size of the system. A smaller system and a larger system require about the same number of Monte Carlo sweeps to reach equilibrium if finite-size effects do not dominate the smaller system. Since a Monte Carlo sweep of the system with fewer particles takes less time, the smaller system thermalizes faster. We take advantage of this property by first thermalizing a smaller system and then using it to initialize a larger system. For instance, we can create an initial configuration with four times as many particles by repeating a configuration from the smaller system twice in each direction. The periodic boundary conditions ensure that the four copies of the smaller configuration fit together correctly, so if the smaller system is in equilibrium, the initial configuration for the larger system will also be in equilibrium, except for finite-size effects.

In our simulations we used both methods of reducing the amount of time required for thermalization. The smallest system that we could simulate conveniently assigned one cell to each of the 64 nodes in the concurrent processor, so the whole system contained 1024 particles. We began a sequence of simulations with the one at the highest density, $\rho^* = .95$, and used a cold start to initialize the system. After thermalizing the system with

10,000 Monte Carlo sweeps, we used the resulting configuration to initialize a simulation at the next lower density. We then repeated the procedure to provide an initial configuration for each of the densities that we intended to simulate. While thermalizing the simulation at each of the densities, we observed that the average values of the potential energy and pressure stabilized, providing an indication that the system was in equilibrium.

After thermalizing each of the simulations of the 1024-particle system, we used the resulting configurations to initialize simulations of a system containing 4096 particles. Since the finite-size effects are small outside the transition region, we expect the initial configurations in the solid and fluid regions to be nearly in equilibrium. Although we will discuss the principal reasons for preferring the larger system in Chapter 6 when we discuss simulations of the transition region, we will mention two advantages of using the larger system in simulations that are outside the transition region. One advantage is that any residual finite-size effects that are present in the smaller system are reduced in the larger system, giving more accurate results. The second advantage is that the same amount of computer time gives more precise expectation values of the potential energy and pressure when used to simulate the larger system.

At first, simulating a system under the same conditions of temperature and density with a fixed amount of computer time seems to result in a statistical error that is independent of the number of particles in the system. The distribution of the individual measurements of a quantity such as the pressure has a standard deviation that is proportional to $N^{-\frac{1}{2}}$, where N is the number of particles in the system. Likewise, the standard deviation of the average of n measurements is proportional to $n^{-\frac{1}{2}}$, even if the measurements are correlated. Although the standard deviation of the average also depends on the sweep-to-sweep correlation length of the system, the decrease in the standard deviation is the same for both systems as n increases if finite-size effects are not important. Thus, the simulation of the smaller system must perform four times as many Monte Carlo sweeps to achieve the same statistical error as the simulation of the larger system. However, simulating the larger system requires four times as much computation per sweep, so the effects seem to balance.

On a sequential computer the competing effects balance, so a simulation obtains a statistical error that is independent of the number of particles in the system. However, the same is not true when the simulation runs on our concurrent processor, since the efficiency of the update algorithm improves as the number of particles per node increases. As we discussed in Section 3.7, the communication overhead is a smaller fraction of the total time and the load balance improves when we simulate a larger system. Table 3.1 shows that the the efficiency of simulating the system with 1024 particles on an 8×8 grid of nodes is about 48%, while the efficiency of simulating the 4096-particle system is about 63%. Although a Monte Carlo sweep of the larger system requires four times as much computation as a sweep of the smaller system, a sweep of the larger system takes only about three times as long. Thus, the same amount of computer time gives a lower statistical error in measurements such as the potential energy and pressure when it is used to simulate the larger system.

4.4 Simulation Results

We now describe the sequence of constant-density simulations in the solid and fluid regimes of the Lennard-Jones system. We have confined most of our simulations to the $T^* = .7$ isotherm in order to avoid spreading the available computer power too thinly. According to calculations by Barker et al.,[18] the temperature $T^* = .7$ lies well above the critical temperature of the two-dimensional Lennard-Jones system, which they estimate to be $T_c^* = .533$. Thus, our system should not exhibit a liquid-gas transition, so we expect to observe only solid and fluid regions separated by a melting transition. In addition to simulating the system along the isotherm, we obtained data along the $\rho^* = .95$ isochore to complete an integration path for the free energy analysis that is discussed in the following chapter.

We chose to study the melting transition along an isotherm because a first-order phase transition has an especially distinctive signature along an isotherm. If the melting transition is first-order, the solid and fluid phases are separated by a two-phase region in which both phases coexist. As the simulation traverses the two-phase region at constant temperature by changing the density of the system, the fraction of the system in each phase changes, but the system's pressure remains constant. In sharp contrast, the pressure varies rapidly with changing density in the adjacent regions of solid and dense fluid, which are both relatively incompressible. Although finite-size effects can change the constant pressure behavior in the two-phase region by preventing a complete separation of the two phases, a van der Waals loop still identifies the region, as we mentioned in Section 1.3. Thus, if the melting transition is first-order, the transition region should be easy to identify. If instead the melting transition is higher-order, the transition region should smoothly connect the solid and fluid regions, and we expect changes in the slope of the pressure and potential energy as functions of density to identify the transition region.

While we were thermalizing the simulations of the 1024-particle system, we observed a van der Waals loop in the pressure data. As mentioned in the previous section, we used a cold start as the starting configuration at $\rho^* = .95$ and initialized simulations at successively lower densities with a thermalized configuration at the previous density. As the density decreased, the equilibrium pressure of the system also decreased, until the system reached $\rho^* = .84$, where the pressure increased sharply during thermalization. Thus, the system exhibited a van der Waals loop in its pressure, indicating that all of the simulations at higher densities whose pressures were lower than that of the system at $\rho^* = .84$ were inside the loop. Since the pressure at $\rho^* = .83$ was significantly lower than at $\rho^* = .84$, the density $\rho^* = .83$ appeared to be outside the transition region. Although the van der Waals loop indicates the presence of a phase transition, we believe that interpreting the loop as evidence that the transition is first-order would be premature. As we noted in Section 1.4, a higher-order transition may appear discontinuous if the system passes through the transition region so quickly that it is not always in equilibrium. For now we will exclude the range of density from $\rho^* = .84$ to $\rho^* = .87$, which appears to lie in the transition region, but in Chapter 6 we will present results from simulations in that range of density.

Using the thermalized configurations of the 1024-particle system, we initialized simulations of the system with $N = 4096$ and again thermalized the simulations. Tables 4.1 to 4.4 report the measurements of the potential energy and pressure at each of the simulation densities that lie along the $T^* = .7$ isotherm and outside the transition region. The columns labeled U^*_{mc} and p^*_{mc} report the measured values of potential energy and pressure, respectively, which include the corrections obtained with the uniform density approximation. In addition, we calculated the residual corrections to the potential energies and pressures as discussed in Section 4.2. Since we found that the statistical errors in several values of U^*_{cor} and p^*_{cor} were small compared to the errors in U^*_{mc} and p^*_{mc}, we used only one configuration to calculate some of the values, which do not include error estimates. The last column of the tables gives the number of Monte Carlo sweeps used to evaluate each expectation value. In Figure 4.1 we plot the pressure data from Tables 4.2 and 4.4 and connect the data points with a smooth curve to guide the eye. The transition region lies in the gap between the solid and fluid regions. We also plot the values of p^*_{cor} in Figure 4.2 for the same data and see that the qualitative predictions about the behavior of the residual correction in Section 4.2 are accurate.

Tables 4.5 and 4.6 report the results of the simulations along the $\rho^* = .95$ isochore in the same format as the other tables. Since we calculated the potential energy and pressure at $T^* = 0$ exactly, using a static triangular lattice, the measurements involved no Monte Carlo sweeps.

Table 4.1

Solid Potential Energies for T^*=.7 and N=4096

ρ^*	U^*_{mc}	U^*_{cor}	U^*_{total}	sweeps
0.95	−2.74450±.00047	.00266±.00005	−2.74184±.00047	10000
0.94	−2.75533±.00026	.00176	−2.75357±.00026	10000
0.93	−2.76064±.00051	.00036	−2.76028±.00051	10000
0.92	−2.75926±.00040	−.00052	−2.75978±.00040	10000
0.91	−2.75116±.00040	−.00143	−2.75259±.00040	10000
0.90	−2.73720±.00061	−.00220	−2.73940±.00061	10000
0.89	−2.71958±.00036	−.00300	−2.72258±.00036	10000
0.88	−2.69808±.00028	−.00356	−2.70164±.00028	15000

Table 4.2

Solid Pressures for T^*=.7 and N=4096				
ρ^*	p^*_{mc}	p^*_{cor}	p^*_{total}	sweeps
0.95	6.5914±.0037	.0076±.0001	6.5990±.0037	10000
0.94	5.8910±.0020	.0050	5.8960±.0020	10000
0.93	5.2495±.0039	.0010	5.2505±.0039	10000
0.92	4.6739±.0032	−.0014	4.6725±.0032	10000
0.91	4.1642±.0031	−.0039	4.1603±.0031	10000
0.90	3.7169±.0051	−.0059	3.7110±.0051	10000
0.89	3.3104±.0031	−.0080	3.3024±.0031	10000
0.88	2.9490±.0022	−.0094	2.9396±.0022	15000

Table 4.3

Fluid Potential Energies for T^*=.7 and N=4096

ρ^*	U^*_{mc}	U^*_{cor}	U^*_{total}	sweeps
0.83	−2.4465 ±.0012	−.00246±.00003	−2.4490 ±.0012	70000
0.82	−2.41573±.00098	−.00224±.00005	−2.41797±.00098	5000
0.81	−2.3831 ±.0011	−.00219±.00006	−2.3853 ±.0011	5000
0.80	−2.35046±.00057	−.00195	−2.35241±.00057	10000
0.78	−2.29331±.00061	−.00163±.00004	−2.29494±.00061	5000
0.75	−2.20127±.00048	−.00128±.00005	−2.20255±.00048	5000
0.70	−2.04530±.00033	−.00079	−2.04609±.00033	5000
0.65	−1.89298±.00046	−.00054±.00004	−1.89352±.00046	5000
0.60	−1.75000±.00096	−.00038	−1.75038±.00096	5000
0.55	−1.61590±.00073	−.00022±.00003	−1.61612±.00073	5000
0.50	−1.4960 ±.0016	−.00017	−1.4962 ±.0016	5000
0.45	−1.3929 ±.0015	−.00051±.00006	−1.3934 ±.0015	5000
0.40	−1.2693 ±.0013	−.00036±.00006	−1.2697 ±.0013	5000
0.35	−1.1429 ±.0019	−.00035±.00003	−1.1433 ±.0019	5000
0.30	−1.0226 ±.0023	−.00024±.00004	−1.0228 ±.0023	5000
0.25	−0.8977 ±.0014	−.00035±.00005	−0.8981 ±.0014	5000
0.20	−0.7468 ±.0012	−.00027±.00004	−0.7471 ±.0012	5000
0.15	−0.5936 ±.0012	−.00009±.00004	−0.5937 ±.0012	5000
0.10	−0.41732±.00055	−.00003±.00001	−0.41735±.00055	5000

Table 4.4

Fluid Pressures for T^*=.7 and N=4096

ρ^*	p^*_{mc}	p^*_{cor}	p^*_{total}	sweeps
0.83	2.6051 ±.0085	−.0061 ±.0001	2.5990 ±.0085	70000
0.82	2.3941 ±.0079	−.0055 ±.0001	2.3886 ±.0079	5000
0.81	2.2012 ±.0089	−.0053 ±.0001	2.1959 ±.0089	5000
0.80	2.0368 ±.0053	−.0047	2.0321 ±.0053	10000
0.78	1.6667 ±.0060	−.0038 ±.0001	1.6629 ±.0060	5000
0.75	1.2385 ±.0044	−.0029 ±.0001	1.2356 ±.0044	5000
0.70	0.7591 ±.0029	−.0017	0.7574 ±.0029	5000
0.65	0.4955 ±.0017	−.0011 ±.0001	0.4944 ±.0017	5000
0.60	0.3420 ±.0026	−.0007	0.3413 ±.0026	5000
0.55	0.2544 ±.0016	−.0004 ±.0001	0.2540 ±.0016	5000
0.50	0.2091 ±.0018	−.0003	0.2088 ±.0018	5000
0.45	0.17857±.00080	−.00069 ±.00008	0.17788±.00080	5000
0.40	0.1532 ±.0011	−.00043 ±.00008	0.1528 ±.0011	5000
0.35	0.13336±.00057	−.00037 ±.00004	0.13299±.00057	5000
0.30	0.11827±.00081	−.00019 ±.00003	0.11808±.00081	5000
0.25	0.10458±.00030	−.00025 ±.00004	0.10433±.00030	5000
0.20	0.08969±.00029	−.00016 ±.00002	0.08953±.00029	5000
0.15	0.07326±.00019	−.00004 ±.00002	0.07322±.00019	5000
0.10	0.05439±.00006	−.000009±.000004	0.05438±.00006	5000

Table 4.5

Solid Potential Energies for ρ^*=.95 and N=4096

T^*	U^*_{mc}	U^*_{cor}	U^*_{total}	sweeps
0.0	−3.38280	.00908	−3.37372	0
0.1	−3.28485±.00011	.00854±.00002	−3.27631±.00011	5000
0.2	−3.18951±.00016	.00691±.00006	−3.18260±.00017	5000
0.3	−3.09599±.00020	.00569±.00007	−3.09030±.00021	5000
0.4	−3.00479±.00028	.00456±.00005	−3.00023±.00028	5000
0.5	−2.91708±.00058	.00382±.00004	−2.91326±.00058	5000
0.6	−2.83119±.00040	.00320±.00006	−2.82799±.00040	5000
0.7	−2.74450±.00047	.00266±.00005	−2.74184±.00047	10000

Table 4.6

Solid Pressures for ρ^*=.95 and N=4096

T^*	p^*_{mc}	p^*_{cor}	p^*_{total}	sweeps
0.0	0.9824	.0258	1.0082	0
0.1	1.8522±.0009	.0243±.0001	1.8765±.0009	5000
0.2	2.6913±.0012	.0197±.0002	2.7110±.0012	5000
0.3	3.5124±.0015	.0162±.0002	3.5286±.0015	5000
0.4	4.3128±.0022	.0130±.0001	4.3258±.0022	5000
0.5	5.0827±.0045	.0109±.0001	5.0936±.0045	5000
0.6	5.8354±.0031	.0091±.0002	5.8445±.0031	5000
0.7	6.5914±.0037	.0076±.0001	6.5990±.0037	10000

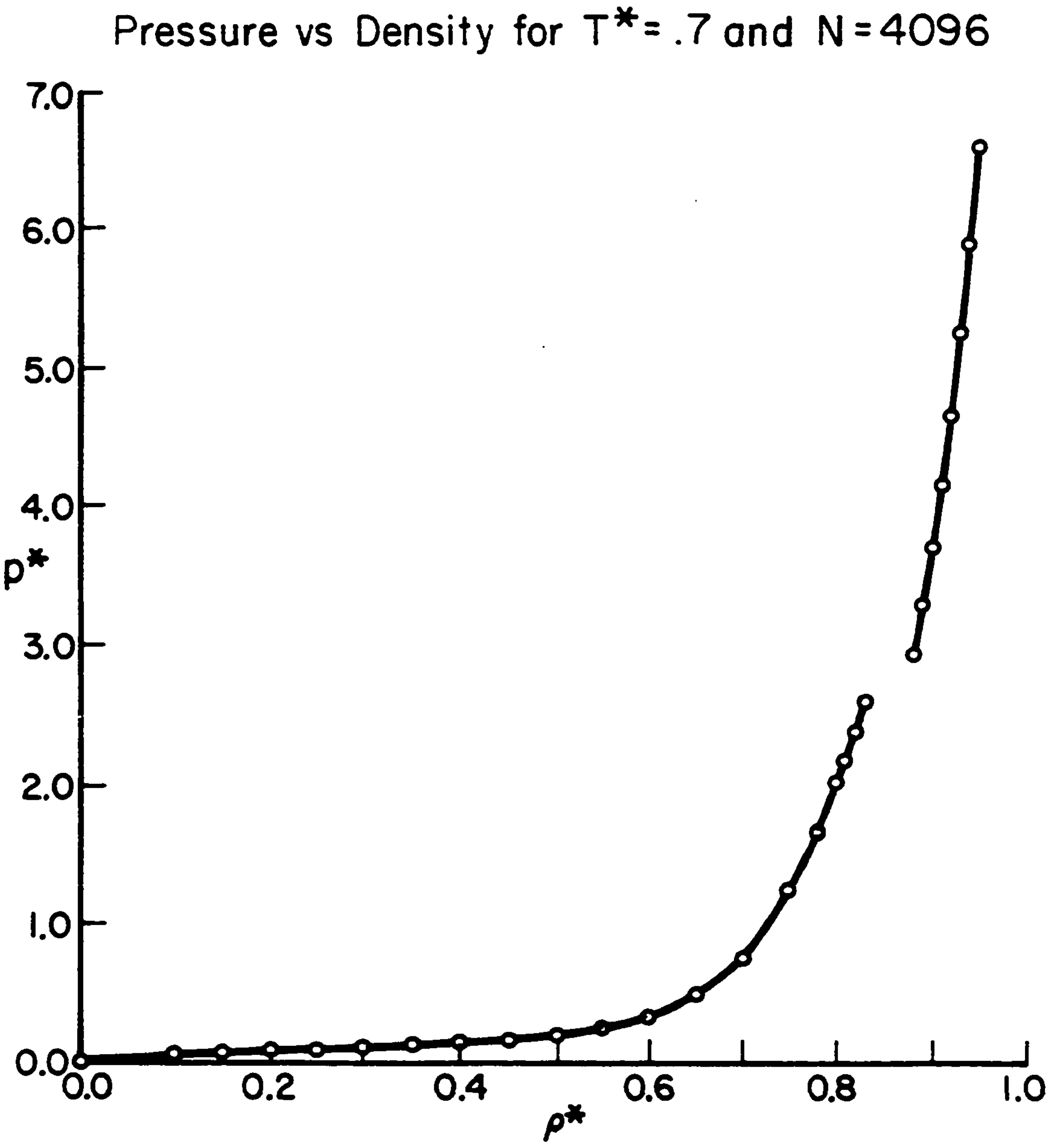


Figure 4.1 A pressure versus density plot of the data from Tables 4.2 and 4.4. The melting transition lies in the gap that separates the lines connecting the data points.

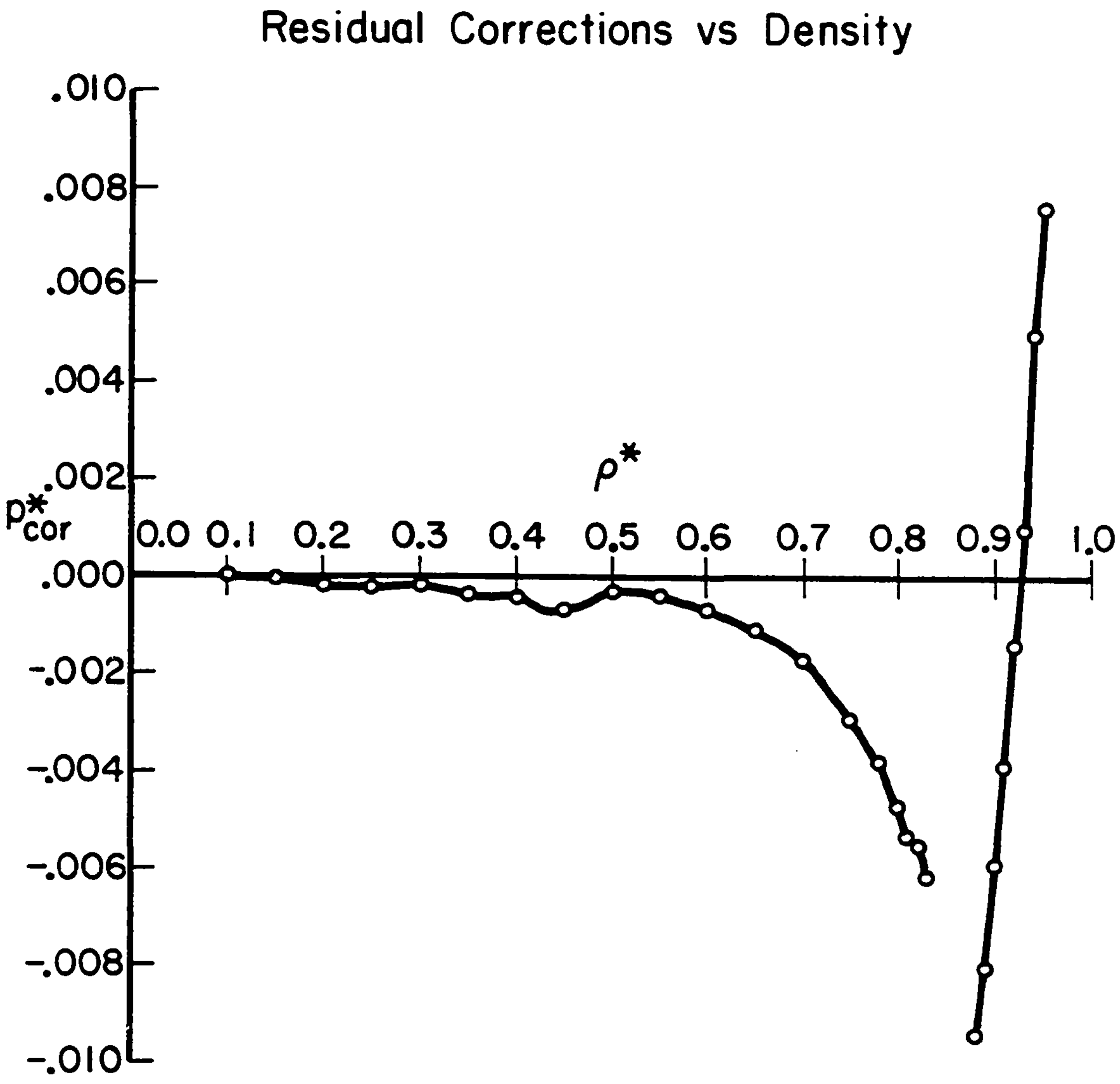


Figure 4.2 A plot of the residual corrections to the pressure versus density. The residual correction exhibits the qualitative features that we discussed in Section 4.2. The melting transition lies in the gap that separates the lines connecting the data points.

Chapter 5: Free Energy Analysis

5.1 Thermodynamic Integration

In this chapter we assume that the melting transition of the two-dimensional Lennard-Jones system is first-order and calculate the parameters of the transition using the Monte Carlo data presented in Tables 4.1 to 4.6. We use a procedure known as *thermodynamic integration* to obtain the free energies of the solid and fluid phases, which we then use to predict the location of the phase transition.[18] In the following chapter, we will check our predictions by comparing them to the results of directly simulating the system in the transition region. If the predictions of the free energy analysis disagree with the simulation results, we must question the validity of the assumption that the transition is first-order. On the other hand, strong agreement between the predictions and the simulation results would provide evidence that the assumption is correct.

A single component system maintained at a constant temperature T and average pressure p is in equilibrium when its *Gibbs free energy* is minimized.[34] The Gibbs free energy is

$$G \equiv E + pV - TS \tag{5.1}$$

where E is the total kinetic and potential energy of the system, V is its volume, and S is its entropy. In order for arbitrary proportions of two phases to coexist in equilibrium, the Gibbs free energy per particle, G^*, must be equal in both phases. Although two distinct phases cannot coexist if the transition is higher-order, the system still minimizes G^* across the phase transition. The assumption that the phase transition is first-order enters the calculation when we extrapolate G^* into the transition region, using its trajectories in the solid and fluid phases. Extrapolating G^* from the solid and fluid data would miss any unexpected phases in the transition region, such as the proposed hexatic phase, and give incorrect results.

Since we can obtain only expectation values from a Monte Carlo simulation, as described in Section 2.2, we cannot directly measure G^*, which cannot be expressed as such an expectation value. In terms of the partition function Z, the entropy of the system is[35]

$$S = k\,(\ln Z + \beta E)\ . \tag{5.2}$$

In order to make the Monte Carlo calculation computationally feasible, we used the method of importance sampling, which requires only the relative probability of occurrence of a configuration, rather than its absolute probability. Without knowing the absolute probabilities of all possible configurations, we cannot directly measure quantities such as S and G that are functions of $\ln Z$.

Instead, we calculate the free energy of the system, using thermodynamic integration, which is a standard method of evaluating quantities that depend on the value of the partition function itself. Using formulas that relate derivatives of the free energy to quantities that can be directly measured directly in a Monte Carlo simulation, we obtain the free energy by integrating the derivatives along appropriate paths. We begin the derivation of the necessary formulas with the First Law of Thermodynamics[36]

$$dQ = dE + dW \tag{5.3}$$

where Q is the heat absorbed by the system, W is the mechanical work performed by the system, and E is its internal external energy. Using $dQ = TdS$, which is valid for a quasi-static process, we obtain

$$T\,dS = dE + p\,dV\ . \tag{5.4}$$

The simulation data from the previous chapter provide integration paths along the $T^* = .7$ isotherm and the $\rho^* = .95$ isochore, so we want to differentiate the free energy with respect to these variables. We switch to using T and V as the independent variables by substituting $T\,dS = d(TS) - S\,dT$ into (5.4), which gives

$$dF = -p\,dV - S\,dT \tag{5.5}$$

where the Helmholtz free energy is defined as $F \equiv E - TS$. Taking the derivative of $F(T, V)$ gives

$$dF = \left(\frac{\partial F}{\partial T}\right)_V dT + \left(\frac{\partial F}{\partial V}\right)_T dV \ . \tag{5.6}$$

Equating the coefficients of dT and dV in (5.5) and (5.6) gives

$$\left(\frac{\partial F}{\partial T}\right)_V = -S \tag{5.7}$$

and

$$\left(\frac{\partial F}{\partial V}\right)_T = -p \ . \tag{5.8}$$

The simulation can easily measure the pressure of the system, so (5.8) gives the thermodynamic derivative that we will use to integrate along an isotherm. Since the number of particles in the system is fixed, we can rewrite (5.8) in terms of $\rho = \frac{N}{V}$, using

$$\frac{\partial V}{\partial \rho} = -\frac{N}{\rho^2} \ . \tag{5.9}$$

Substituting (5.9) into (5.8) using the chain rule and converting to reduced quantities give

$$\left(\frac{\partial F^*}{\partial \rho^*}\right)_{T^*} = \frac{p^*}{\rho^{*2}} \ . \tag{5.10}$$

As we have mentioned, the simulation cannot measure S directly, so (5.7) is not a useful derivative for integrating the free energy along an isochore. However, we can construct a slightly different derivative that the Monte Carlo simulation can measure, which is

$$\frac{\partial}{\partial T}\left(\frac{F}{T}\right)_\rho = \frac{1}{T}\left(\frac{\partial F}{\partial T}\right)_\rho - \frac{F}{T^2} . \tag{5.11}$$

Substituting (5.7) into (5.11) gives

$$\frac{\partial}{\partial T}\left(\frac{F}{T}\right)_\rho = -\frac{S}{T} - \frac{E - TS}{T^2} \tag{5.12}$$

which leads to a formula that is useful for integrating the free energy along an isochore. Converting (5.12) to reduced quantities gives

$$\frac{\partial}{\partial T^*}\left(\frac{F^*}{T^*}\right)_{\rho^*} = -\frac{U^*}{T^{*2}} \tag{5.13}$$

where we have replaced the total internal energy E with the potential energy U. The reference values of the free energy that we calculate in the next section account for the kinetic energy exactly, so we must ignore its contribution to the derivatives of the free energy. The reason that the kinetic energy enters the calculation exactly is that the kinetic degrees of freedom are noninteracting, so their contribution to the partition function can be solved analytically.

Thus, we have constructed two derivatives, (5.10) and (5.13), that allow us to integrate the Helmholtz free energy along paths of constant temperature and constant density, respectively, using the data reported in Chapter 4. Tables 4.2 and 4.4 provide measurements of the system's pressure along the $T^* = .7$ isotherm and Table 4.5 provides measurements of its potential energy along the $\rho^* = .95$ isochore. However, before we can numerically integrate the free energy derivatives, we need reference values of known free energy for both the solid and fluid phases.

5.2 Reference Values of the Free Energy

Since we must be able to calculate the reference values of the free energy exactly, we use the convenient limits of an ideal gas and a harmonic solid.[18] As the fluid phase of the Lennard-Jones system approaches $\rho^* = 0$ at $T^* = .7$, it becomes an ideal gas, whose free energy can be obtained by evaluating its partition function. Likewise, as the solid phase approaches $T^* = 0$ at $\rho^* = .95$, it becomes a harmonic crystal, whose free energy can be calculated using a normal mode expansion. Since we use the reference values only in their appropriate limits as boundary conditions for the integrals of the free energy derivatives, they are not approximations and introduce no error into the calculation. In Section 5.4 we will show that the reference values join smoothly onto the Monte Carlo measurements when we actually perform the numerical integration.

We calculate the free energy of an ideal gas in two dimensions by starting with the partition function, Z, for a set of N noninteracting particles, which is

$$Z = \frac{1}{h^{2N} N!} \int_{-\infty}^{\infty} e^{-\frac{p_1^2 + \cdots p_N^2}{2mkT}} d^2p_1 \cdots d^2p_N \int_V d^2q_1 \cdots d^2q_N \ . \tag{5.14}$$

Since the particles are noninteracting, we can immediately evaluate the integrals over the particles' coordinates, which give V^N, where V is the volume of the system. In addition, we can factor the integrals over the particles' momenta to obtain

$$Z = \frac{V^N}{h^{2N} N!} \left[\int_{-\infty}^{\infty} e^{-\frac{p^2}{2mkT}} d^2p \right]^N \ . \tag{5.15}$$

The integral in (5.15) is a Gaussian integral whose value is

$$\int_{-\infty}^{\infty} e^{-\frac{p^2}{2mkT}} d^2p = 2\pi mkT \tag{5.16}$$

where $p^2 = p_x^2 + p_y^2$ since the system is two-dimensional. Substituting (5.16) into (5.15) gives

$$Z = \frac{1}{N!}\left[\frac{2\pi mkTV}{h^2}\right]^N \tag{5.17}$$

for the partition function of an ideal gas. Using (5.2) and the definition of the Helmholtz free energy, we obtain

$$F = -kT \ln Z \ . \tag{5.18}$$

Substituting (5.17) into (5.18) and using Stirling's approximation to expand the factorial gives

$$F = -kT\left[-(N\ln N - N) + N\ln\left(\frac{2\pi mkTV}{h^2}\right)\right] \ . \tag{5.19}$$

We now convert (5.19) to reduced quantities and obtain

$$F^* = -T^* - T^*\ln T^* + T^*\ln\rho^* + T^*\ln\left(\frac{2\pi\hbar^2}{m\epsilon\sigma^2}\right) \ . \tag{5.20}$$

Next, we calculate the free energy of a harmonic crystal exactly, using standard methods of lattice dynamics, which are discussed in detail by Feynman[37] and used in calculations by Phillips and Bruch.[38] The Helmholtz free energy of a crystal that consists of particles executing simple harmonic motions about their equilibrium positions consists of two components,

$$F = F_{static} + F_{dyn} \tag{5.21}$$

where F_{static} is the static lattice potential energy U_0. F_{dyn} is the sum of the contributions from each of the crystal's M normal modes, which we write as

$$F_{dyn} = \sum_{i=1}^{M} F_i \ . \tag{5.22}$$

In order to calculate the F_i, we first write the partition function for each of the normal modes, which are in thermal equilibrium at the temperature T. The partition function for mode i is the sum over all of its possible excitations, which is

$$Z_i = \sum_{n=0}^{\infty} e^{-\frac{E_n^i}{kT}} \ . \tag{5.23}$$

From quantum mechanics we know that the energy of each excitation is

$$E_n^i = \hbar\omega_i(n + \tfrac{1}{2}) \ . \tag{5.24}$$

Substituting (5.24) into (5.23) and summing the resulting geometric series give

$$Z_i = \frac{e^{-\frac{\hbar\omega_i}{2kT}}}{1 - e^{-\frac{\hbar\omega_i}{kT}}} \ . \tag{5.25}$$

By substituting (5.25) into (5.18), we obtain

$$F_i = kT \ln [2 \sinh (\frac{\hbar\omega_i}{2kT})] \tag{5.26}$$

for the free energy of each mode.

We now convert (5.26) to reduced quantities, using: $F^* = \frac{F}{\epsilon N}$, $T^* = \frac{kT}{\epsilon}$, $\Lambda^* = \frac{\hbar}{\sqrt{m\epsilon\sigma^2}}$, and $\omega^* = \left[\frac{m\sigma^2}{\epsilon}\right]^{\frac{1}{2}}\omega$. In addition, we take the limit of (5.26) when

$T^* \gg \Lambda^* \omega^*$ since we want the free energy for the classical two-dimensional crystal. Substituting the classical limit of (5.26) into (5.22) gives

$$F^*_{dyn} = \frac{1}{N} \sum_{i=1}^{M} T^* \ln\left(\frac{\Lambda^* \omega_i^*}{T^*}\right) . \tag{5.27}$$

The total number of normal modes in the crystal is the same as the total number of coordinate degrees of freedom. Since each particle in a d-dimensional system has d such degrees of freedom, a two-dimensional system with N particles has $M = 2N$, which we substitute into (5.27) to obtain

$$F^*_{dyn} = -2T^* \ln T^* + 2T^* \ln \Lambda^* + \frac{T^*}{N} \sum_{i=1}^{M} \ln \omega_i^* . \tag{5.28}$$

Substituting (5.28) into (5.21) and expanding Λ^* for comparison with the free energy of an ideal gas in (5.20) gives

$$F^* = U_0^* - 2T^* \ln T^* + T^* \ln\left(\frac{\hbar^2}{m\epsilon\sigma^2}\right) + \frac{T^*}{N} \sum_{i=1}^{M} \ln \omega_i^* \tag{5.29}$$

By comparing (5.20) and (5.29), we see that $\hbar$ and the particle mass m correctly cancel in the classical system when we take the difference of the free energies at the same temperature.

Before we can use (5.29) as a reference value in the free energy analysis, we must solve for the M normal mode frequencies and sum their logarithms. Since solving for the normal mode frequencies is rather lengthy, we leave a detailed discussion of the calculation for Appendix C and only quote its result here. For the Lennard-Jones system with N=4096 and the periodic boundary conditions specified in Section 2.3, the sum of the logarithms of the normal mode frequencies is

$$\frac{1}{N} \sum_{i=1}^{M} \ln \omega_i^* = 5.19514 . \tag{5.30}$$

5.3 Methods of Numerical Integration

Before continuing with the discussion of the free energy analysis, we need to examine methods of numerically integrating the free energy derivatives that will give accurate results with relatively few data points. Methods of numerical integration consist of two steps, converting a discrete set of data points into a continuous function and then integrating the continuous function. The method of converting the set of data points, rather than the values of the data points, determines the form of the function produced by the first step, so the integration is independent of the values of the data points. Thus, integrating the function and substituting the result into the formula that converts the set of data points into the function give a formula for the numerical integral directly in terms of the set of data points.

Since the integration in the second step is exact, the quality of a method of numerical integration depends entirely on the type of continuous function into which it converts the discrete set of data points. For instance, the trapezoidal rule converts a set of data points into a continuous function by simply connecting adjacent data points with a straight line. In contrast, Simpson's rule generates the continuous function by connecting each set of three data points with a parabola so that adjacent parabolas share a data point. Although the continuous function produced by Simpson's rule is much better than that produced by the trapezoidal rule, neither function is particularly accurate in most cases. If we could easily increase the number of data points, these methods would be adequate, since increasing the number of data points improves the accuracy of the result. However, increasing the number of data points generated by a Monte Carlo simulation involves much more computation than improving the method of converting the data points into a continuous function. Consequently, we must find better methods of converting a discrete set of data points into a continuous function so that we can obtain more accurate results from a fixed set of data points.

In some situations we may know that a power series accurately describes the function that the set of data points represents, so we use the n data points to determine the $m+1$ coefficients of an m-degree polynomial, which we denote P_m. In order to determine the coefficients, we first assume that each data point consists of a value x_i that parameterizes its measured value y_i, which samples a normal distribution whose standard deviation is σ_i. We then construct $d_i = y_i - P_m(x_i)$, which measures the deviation of the measured value from the polynomial at x_i. Since y_i samples a normal distribution, the probability P of obtaining each value of d_i is

$$P(d_i) \propto \exp\left(\frac{d_i^2}{\sigma_i^2}\right) \tag{5.31}$$

and the combined probability of obtaining the set of n measurements is the product of (5.31) for each of the individual measurements.

We obtain the familiar set of least-squares conditions for the coefficients of P_m by applying the *principle of maximum likelihood*, which states that the set of measurements that we obtained was the most likely such set of measurements.[39] Setting to zero the derivative of the combined probability with respect to each of the undetermined coefficients, c_j, and using $w_i = \sigma_i^{-2}$ give

$$\sum_{i=1}^{n} w_i [y_i - P_m(x_i)] \frac{dP_m(x_i)}{dc_j} = 0 \tag{5.32}$$

for the $m+1$ conditions that determine the c_j. As a specific example we consider the case where $m = 3$ and the polynomial has the form

$$P_3(x_i) = c_0 + c_1 x_i + c_2 x_i^2 + c_3 x_i^3 \ . \tag{5.33}$$

Applying the conditions given by (5.32), we obtain four simultaneous equations, which we write in matrix form as

$$\begin{bmatrix} \sum w_i & \sum w_i x_i & \sum w_i x_i^2 & \sum w_i x_i^3 \\ \sum w_i x_i & \sum w_i x_i^2 & \sum w_i x_i^3 & \sum w_i x_i^4 \\ \sum w_i x_i^2 & \sum w_i x_i^3 & \sum w_i x_i^4 & \sum w_i x_i^5 \\ \sum w_i x_i^3 & \sum w_i x_i^4 & \sum w_i x_i^5 & \sum w_i x_i^6 \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} \sum w_i y_i \\ \sum w_i x_i y_i \\ \sum w_i x_i^2 y_i \\ \sum w_i x_i^3 y_i \end{bmatrix} . \qquad (5.34)$$

Solving (5.34) by an appropriate method, such as Gaussian elimination, gives the values of the four coefficients in (5.33) that minimize the sum of the squares of the d_i. After fitting the polynomial to the set of data points, we integrate the polynomial to calculate the numerical integral of the data points.

The result of the numerical integration is accurate only when the polynomial correctly describes the function that the data points represent. Although the fit of a polynomial to the data points improves as the degree of the polynomial increases, a problem known an the Runge phenomenon[40] can occur when a high-degree polynomial is fit to equally spaced data points. As a result of the Runge phenomenon, the polynomial exhibits oscillations which increase in amplitude as the distance from its origin increases, due to the competing effects of the large powers of x. While such a polynomial may accurately fit the data points, the large oscillations make it a poor method of interpolating between them, so the value of the numerical integral could be seriously in error. On the other hand, if a low-degree polynomial fits the data well, the resulting polynomial should be useful for interpolating between the data points as well as extrapolating beyond them, which we must do to obtain the free energy of the system in the transition region. Still, in many cases a polynomial is simply the wrong function to use because a single polynomial cannot fit all of the data points. An improvement would be to use a set of polynomials, each of which models only a portion of the interval that the data points encompass, similar to the parabolas used in Simpson's rule.

Such a set of polynomials comprise spline functions, which provide a general method of connecting a set of data points with a smooth curve. A generalized spline function of degree m connects adjacent points with a polynomial of degree m.[41] Two of the $m+1$ conditions that uniquely specify each of the polynomials require the polynomial to pass through the two data points that it connects, guaranteeing that the spline is continuous. The remaining $m-1$ conditions require the first $m-1$ derivatives of the polynomial to be continuous on the closed interval bounded by the two data points, making the spline smoother as its degree increases. For instance, the trapezoidal rule uses the spline with $m=1$, which connects adjacent data points with a straight line and has no continuous derivatives. The most commonly used spline is the cubic spline, which has $m=3$ and thus has continuous first and second derivatives.

We now consider the cubic spline that connects $n+1$ data points, which consist of measured values y_i that are parameterized by x_i. We order the values of x_i so that $x_0<x_1<\cdots<x_n$ and parameterize the each of the n intervals where $x_{i-1}<x<x_i$ with

$$\begin{aligned} h_i &= x_i - x_{i-1} \\ d_i &= \frac{y_i - y_{i-1}}{h_i} \quad . \\ t &= \frac{x - x_{i-1}}{h_i} \end{aligned}$$

The cubic spline function $q_i(x)$ for $i = 1, 2, 3, \cdots, n$ is

$$q_i(x) = ty_i + (1-t)y_{i-1} + h_i t(1-t)[(k_{i-1} - d_i)(1-t) - (k_i - d_i)t] \tag{5.35}$$

where $k_0, k_1, \cdots, k_n$ satisfy

$$h_{i+1}k_{i-1} + 2(h_i + h_{i+1})k_i + h_i k_{i+1} = 3(h_i d_{i+1} + h_{i+1}d_i) \quad . \tag{5.36}$$

Since (5.36) contains $n+1$ unknowns and only $n-1$ equations, we must specify two additional conditions on the k_i. Setting the second derivative of the spline to zero at x_0 and x_n is the usual method of obtaining the two additional conditions, which leads to the equations

$$\begin{gathered} 2k_0 + k_1 = 3d_1 \\ k_{n-1} + 2k_n = 3d_n \end{gathered} \quad . \tag{5.37}$$

We can easily verify that the cubic spline connects all of the data points and has continuous first and second derivatives by differentiating (5.35) and substituting the conditions given by (5.36) and (5.37). Differentiating (5.35) also provides the interpretation that k_i is the derivative of the spline function at x_i. We note that the equations given by (5.36) and (5.37) constitute a linear tridiagonal set of equations, which can be solved in an amount of time that grows linearly with n, so cubic splines are useful even when n is large.

We now perform the second step of numerical integration, which involves integrating the cubic spline that interpolates between the data points. In order to make the integration easier, we rewrite (5.35) as

$$q_i(t) = y_{i-1} + h_i k_{i-1} t - h_i(2k_{i-1} + k_i - 3d_i)t^2 + h_i(k_{i-1} + k_i - 2d_i)t^3 \ . \tag{5.38}$$

Integrating (5.38) over the whole interval between the two data points that bound it gives

$$\int_{x_{i-1}}^{x_i} q_i(x)\,dx = h_i \int_0^1 q_i(t)\,dt = h_i y_{i-1} + h_i^2 [\, \tfrac{1}{12}(k_{i-1} - k_i) + \tfrac{1}{2} d_i \,] \ . \tag{5.39}$$

Since the cubic spline smoothly interpolates between data points by using a set of polynomials, it is the basis of a generally useful method of numerical integration. In contrast to polynomial fitting, the cubic spline works well even in situations when fitting all of the data points to a single polynomial is inappropriate. In addition, equally spaced data points do not cause problems for the cubic spline as they can when a high-degree polynomial is fit to them. However, the cubic spline has larger errors near its endpoints than in the middle of the interval because in (5.37) we arbitrarily set the curvature at the endpoints to zero. In contrast to a low-degree polynomial that fits the data well, the cubic spline has little ability to extrapolate beyond the data since it does not assume a functional dependence that is valid over the whole interval. Since both methods of numerical integration have advantages and disadvantages, we will use both of the methods in the calculation of the free energy.

Before calculating the free energy of the system, we examine the two sources of error that result from numerically integrating the data. The first source is systematic error, which is the difference that would remain between the true value of the integral and the result of the numerical integration if each of the data points were known exactly. The systematic error depends on how accurately the continuous function used by the method of numerical integration models the true function that the data points represent. Although we cannot determine the systematic error without knowing the true function, comparing the results of methods that use different functions gives a rough estimate of the size of the systematic error. Of course, if the data points represent a function that is not relatively smooth between adjacent data points, the systematic error may be large even if several methods of numerical integration agree closely. In such a situation, the data points do not convey sufficient information about the function that they represent to allow the function to be integrated accurately, so more data points are clearly needed.

The second source of error is statistical error, which arises from the statistical errors in the Monte Carlo data that are being integrated. An easy method of propagating statistical errors through a complex analysis, such as numerically integrating many data points, involves repeating the analysis on many data sets generated from the original data. We assume that each of the original data points samples a normal distribution that is centered on the measured value y_i and has a standard deviation of σ_i, which is the measured standard deviation of y_i. Sampling the distributions associated with each of the original data points generates a new data set, which we can use in the analysis as if it were the original data set. Repeating the analysis on many such data sets allows the effect of the statistical error in each of the data points to be seen in the results of the analysis. Although sampling the input data from normal distributions does not ensure that the results of the analysis will be normally distributed, we expect any deviations from a normal distribution to be small so that expressing each result in terms of its mean and standard deviation is adequate.

In order to sample the distribution associated with a data point, we must generate normally distributed random numbers with a specified mean and standard deviation. Starting with two independent random numbers that are uniformly distributed between 0 and 1, the Box-Muller's transformation[42] generates two independent random numbers that are normally distributed with a mean of 0 and a standard deviation of 1. The Box-Muller's transformation is

$$\begin{aligned} n_1 &= \sqrt{-2\ln r_1}\cos(2\pi r_2) \\ n_2 &= \sqrt{-2\ln r_1}\sin(2\pi r_2) \end{aligned} \tag{5.40}$$

where n_1 and n_2 are the normally distributed random numbers and r_1 and r_2 are the uniformly distributed random numbers. Using n_1 and n_2, we sample a distribution with a mean of y_i and a standard deviation of σ_i by constructing $y_i + n_1\sigma_i$ and $y_i + n_2\sigma_i$.

5.4 Determination of Phase Boundaries

The first integration that we perform calculates the Helmholtz free energy of the solid at $T^* = .7$ and $\rho^* = .95$, using the values of U^* in Table 4.5. Integrating the free energy derivative (5.13) along an isochore from $T^* = 0$ to $T^* = T_0^*$ gives

$$F^*(\rho_0^*, T_0^*) = F_{harm}^*(\rho_0^*, T_0^*) - T_0^* \int_0^{T_0^*} \frac{U^* - (U_0^* + T^*)}{T^{*2}} dT^* \qquad (5.41)$$

where F_{harm}^* is the harmonic contribution to the total free energy given by (5.29). Subtracting the harmonic contribution to U^* in the integrand of (5.41) balances the contribution of F_{harm}^* and makes the integrand finite at $T^* = 0$. The first term in the harmonic potential energy, U_0^*, is the static lattice potential energy, and the second term, T^*, arises from the equipartition theorem result for a two-dimensional harmonic oscillator.

We first evaluate (5.41) using the polynomial fitting method of numerical integration since we expect a power series expansion of U^* in powers of T^* to be valid near $T^* = 0$. The polynomial to which we fit the data in Table 4.5 is

$$U^* = U_0^* + T^* + c_2 T^{*2} + c_3 T^{*3} + c_4 T^{*4} + c_5 T^{*5} \qquad (5.42)$$

which has four undetermined coefficients. We use a fifth-degree polynomial because polynomials of lower degree do not fit all of the data points to within their statistical errors.

In Table 5.1 we report the values of the undetermined coefficients in (5.42) and the value of F^* at $T^* = .7$ and $\rho^* = .95$ that we obtained by integrating (5.41). We emphasize that we did not first calculate the average values of the coefficients and then use their average values to calculate F^*. Instead, we calculated each average and standard deviation separately, using the values obtained from analyzing the data sets that were generated from the original data. The value of F_{harm}^* that enters into the result is $F_{harm}^* = -0.77396$,

Table 5.1

Integration Parameters along $\rho^* = .95$ Isochore	
c_2	−0.356 ±0.017
c_3	1.00 ±0.12
c_4	−1.75 ±0.26
c_5	1.10 ±0.18
F^*	−0.6767±0.0012

which contains all terms in (5.29) that do not cancel with terms in (5.20) at equal temperature. The standard deviations of the coefficients and the free energy indicate that the Runge phenomenon has not caused a major problem with fitting a relatively high-degree polynomial to the equally spaced data points. Figure 5.1 illustrates the results of fitting the data points in Table 4.5 to (5.42) with a plot of the integrand of (5.41) versus T^*, which clearly shows the absence of any significant oscillations in the polynomial.

As a check on the results of the polynomial fitting method, we repeat the evaluation of (5.41), using the cubic spline method of numerical integration. However, we must extrapolate from the data point at $T^* = .1$ to $T^* = 0$ because the data point at $T^* = 0$ is U_0, the static lattice potential energy, which enters (5.41) exactly. An easy way of extrapolating the cubic spline uses a straight line with a slope equal to the slope of the spline at $T^* = .1$. The result of integrating the cubic spline and propagating the statistical errors as described in the previous section is $F^* = -0.6792 \pm 0.0016$. Although extrapolating the spline should be less accurate than extrapolating the polynomial, we find that the results of the two methods of numerical integration are consistent, indicating that the systematic error of the integration is relatively small. However, Figure 5.1 shows that the spline and the polynomial give different results for the integrand of (5.41) near $T^* = 0$, where the statistical error of the data is largest. In fact, the statistical consistency of the two methods is probably due to the large uncertainty of the data points near $T^* = 0$. Since we expect that fitting the data to a polynomial allowed a more accurate extrapolation, we will use its value of F^* in subsequent analysis.

Using the value of F^* at $T^* = .7$ that we obtained by integrating along the $\rho^* = .95$ isochore, we now integrate the free energy along the $T^* = .7$ isotherm. Integrating the thermodynamic derivative in (5.10) gives

$$F^*(\rho_s^*, T^*) = F^*(\rho_0^*, T^*) + \int_{\rho_0^*}^{\rho_s^*} \frac{p^*}{\rho^{*2}} \, d\rho^* \qquad (5.43)$$

which we evaluate by numerically integrating the pressure data in Table 4.2. Since we know F^* at $\rho^* = .95$, we set $\rho_0^* = .95$ in (5.43). However, we do not know ρ_s^*, which is the density of the solid at the phase transition, since it is one of the quantities for which we are solving.

In order to locate the phase transition, we will need to extrapolate beyond the data points in Table 4.2, so we use the polynomial fitting method to integrate (5.43). Since the pressure data in the solid region is very smooth, we expect a power series expansion of p^* in powers of ρ^* to describe the whole region adequately. Although the exact location of the origin of the power series is not important, the series is most accurate when its origin is near the middle of the region. We found that the third-degree polynomial

$$p^* = c_0 + c_1(\rho^* - a) + c_2(\rho^* - a)^2 + c_3(\rho^* - a)^3 \qquad (5.44)$$

adequately fits all eight data points. Substituting (5.44) into (5.43) and integrating gives

$$F^*(\rho_s^*) = F^*(\rho_0^*) + (c_0 - ac_1 + a^2a_2 - a^3c_3)\left(\frac{1}{\rho_0^*} - \frac{1}{\rho_s^*}\right)$$

$$+ (c_1 - 2ac_2 + 3a^2c_3)\ln\left(\frac{\rho_s^*}{\rho_0^*}\right)$$

$$+ (c_2 - 3ac_3)(\rho_s^* - \rho_0^*) + \tfrac{1}{2}c_3(\rho_s^{*2} - \rho_0^{*2}) \ . \tag{5.45}$$

In Section 5.1 we gave the condition for locating the phase transition in terms of the Gibbs free energy rather than the Helmholtz free energy, so we convert from F^* to G^*, using

$$G^* = F^* + \frac{p^*}{\rho^*} \tag{5.46}$$

which requires that the system contain a fixed number of particles. Since we know p^* at $T^* = .7$ and $\rho^* = .95$, we can easily convert the reference value $F^*(\rho_0^*)$ into $G^*(\rho_0^*)$. We then use the expression for p in (5.44) with (5.45) and (5.46) to convert $F^*(\rho_s^*)$ into $G^*(\rho_s^*)$ and obtain

$$G^*(\rho_s^*) = G^*(\rho_0^*) + (c_1 - 2ac_2 + 3a^2c_3)\ln\left(\frac{\rho_s^*}{\rho_0^*}\right) + 2(c_2 - 3ac_3)(\rho_s^* - \rho_0^*)$$

$$+ \tfrac{3}{2}c_3(\rho_s^{*2} - \rho_0^{*2}) \ . \tag{5.47}$$

In Table 5.2 we report the results of fitting the coefficients in (5.44) to the measured pressures in Table 4.2 with the origin set to $a = .91$. Although the data do not determine a precise value of c_3, its effect on the polynomial over the range of densities in which we

Table 5.2

Integration Parameters along Solid $T^* = .7$ Isotherm		
c_0	4.1626	± 0.0020
c_1	48.52	± 0.13
c_2	282.4	± 2.8
c_3	760	±130

are interested is small.

Before we can solve for the parameters of the melting transition, we must calculate G^* for the fluid, which requires establishing an integration path in the fluid region. Using the free energy of an ideal gas as a reference value in the limit where $\rho^* = 0$, we integrate the free energy derivative in (5.10) along the $T^* = .7$ isotherm and obtain

$$F^*(\rho_0^*, T_0^*) = F^*_{ideal}(\rho_0^*, T_0^*) + \int_0^{\rho_0^*} \frac{p^* - \rho^* T^*}{\rho^{*2}} d\rho^* \ . \tag{5.48}$$

The contribution to the free energy of an ideal gas at the indicated temperature and pressure is F^*_{ideal}, which we calculated in Section 5.2. Subtracting the pressure of an ideal gas from the integrand in (5.48) balances the contribution of F^*_{ideal} and makes the integrand finite at $\rho^* = 0$.

Since a single low-degree polynomial does not adequately fit all of the pressure data in Table 4.4, we use the cubic spline method of numerical integration to evaluate the integral in (5.48). We improve the accuracy of the cubic spline near $\rho^* = 0$ by incorporating two of the virial coefficients into it. At low densities we can express the pressure as a series expansion in powers of the density,[43] which gives the virial expansion

$$p = kT(\rho + B_2\rho^2 + B_3\rho^3 + \cdots) \tag{5.49}$$

in which the virial coefficients, B_2, B_3, $\cdots$, depend on temperature but not on density. By substituting (5.49) into (5.48), we see that the value of the integrand in (5.48) at $\rho^* = 0$ is $B_2^* T^*$. Thus, knowing the second virial coefficient provides an additional data point that could not be obtained directly by measuring the pressure of the system and eliminates the need to extrapolate the spline to $\rho^* = 0$. In addition, the fourth virial coefficient gives the correct curvature of the spline at its $\rho^* = 0$ endpoint, so we can avoid arbitrarily setting its curvature to zero as we did to obtain one of the equations in (5.37).

Before we can use the modified cubic spline to interpolate the data in Table 4.4, we must obtain the values of the second and fourth virial coefficients for the two-dimensional Lennard-Jones system at $T^* = .7$. Morrison and Ross,[44] who calculated and published values of the second and third virial coefficients, obtained $B_2^* = -2.7613$ at $T^* = .7$. In Appendix D we derive the expression for the second virial coefficient and obtain $B_2^* = -2.761223$, which is the value that we use to evaluate (5.48). Glandt[45] reported calculations of the fourth virial coefficient in which he obtained $B_4^* = 10.1924$ at $T^* = .7$. In Figure 5.2 we plot the integrand of (5.48) versus ρ^*, which we calculated with the modified cubic spline. The result of integrating the modified cubic spline from $\rho^* = 0$ to $\rho^* = .81$ and propagating statistical errors as described in the previous section is $F^* = -1.3104 \pm 0.0013$, which includes $F_{ideal}^* = -0.8475$.

We integrate only to $\rho^* = .81$, using the cubic spline method since we must extrapolate the pressure data in Table 4.4 to locate the phase transition, and the polynomial fitting method often extrapolates more accurately. After fitting a third-degree polynomial to the five highest density data points, we integrate the free energy from $\rho^* = .81$ to the density of the fluid at the phase transition, which we denote ρ_f^*. In Table 5.3 we report the results of fitting the coefficients in (5.44) to the five data points in Table 4.4 that lie in the density range of $\rho^* = .78$ to $\rho^* = .83$ with the origin set to $a = .81$. Although the data do not determine precise values of c_2 and c_3, their effects are relatively small over the range of densities in which we are interested. In any case, the statistical error in the final results of the free energy analysis will correctly account for their uncertainty. The integration formula (5.47) gives G^* for the fluid when we replace ρ_s^* with ρ_f^* and use $\rho_0^* = .81$ as the density of its reference value.

Table 5.3

Integration Parameters along Fluid $T^* = .7$ Isotherm		
c_0	2.2015±	0.0056
c_1	17.46 ±	0.56
c_2	67 ±	23
c_3	2800	±1100

Since we have expressed G^* for both the solid and fluid phases in terms of their densities by fitting the coefficients in (5.44) to measured pressures in both regions and using (5.47), we can now solve for the location of the phase transition. In addition to having the same temperature and pressure, the solid and fluid phases must have equal values of G^* in order for arbitrary proportions of them to coexist in equilibrium, as we mentioned in Section 5.1. Thus, we locate the phase transition on the $T^* = .7$ isotherm by setting

$$G^*_{solid}(p^*_m) = G^*_{fluid}(p^*_m) \tag{5.50}$$

where p^*_m is the melting pressure.

Starting with an initial guess of p^*_m, the iterative algorithm that we use to solve (5.50) evaluates G^* for the solid and fluid phases and corrects p^*_m, until the two values of G^* are equal. Since (5.47) expresses G^* in terms of the density, we must solve two third-degree polynomials to obtain the solid and fluid densities at the current estimate of p^*_m before we can evaluate (5.47). To solve for the densities, we use Newton's method, which gives

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \tag{5.51}$$

for successive approximations to the solution of $f(x)=0$. After using the resulting densities to obtain values of G^* for the two phases, we correct p^*_m using Newton's method and repeat the procedure.

The derivative that Newton's method requires to correct the estimate of the melting pressure is $\frac{\partial G^*}{\partial p^*}$, which we derive by starting with

$$T\,dS = dE + p\,dV \ . \tag{5.4}$$

We switch to using T and p as the independent variables by substituting $T\,dS = d(TS) - S\,dT$ and $p\,dV = d(pV) - V\,dp$ into (5.4), which gives

$$dG = S\,dT + V\,dp \tag{5.52}$$

where we use the definition of G from (5.1). Taking the derivative of $G(T,p)$ gives

$$dG = \left(\frac{\partial G}{\partial T}\right)_p dT + \left(\frac{\partial G}{\partial p}\right)_T dp \ . \tag{5.53}$$

We now equate the coefficients of dp in (5.52) and (5.53) and convert to reduced units to obtain

$$\left(\frac{\partial G^*}{\partial p^*}\right)_{T^*} = \frac{1}{\rho^*} \ . \tag{5.54}$$

Following the procedure discussed in the previous section, we propagate the statistical errors of the data points in Tables 4.2 and 4.4 by repeating the entire analysis many times with data sets generated from the original data. In Table 5.4 we report the melting pressure and the solid and fluid densities at the melting pressure that the free energy analysis predicts, along with their standard deviations. Figure 5.3 illustrates the predicted location of the melting transition with a plot of pressure versus density, which shows the polynomials that were fit to the solid and fluid pressures and the values and statistical errors of ρ_f^*, ρ_s^*, and p_m^*.

Table 5.4

Predicted Location of Melting Transition	
p_m^*	2.724 ±0.034
ρ_f^*	0.8350±0.0015
ρ_s^*	0.8733±0.0011

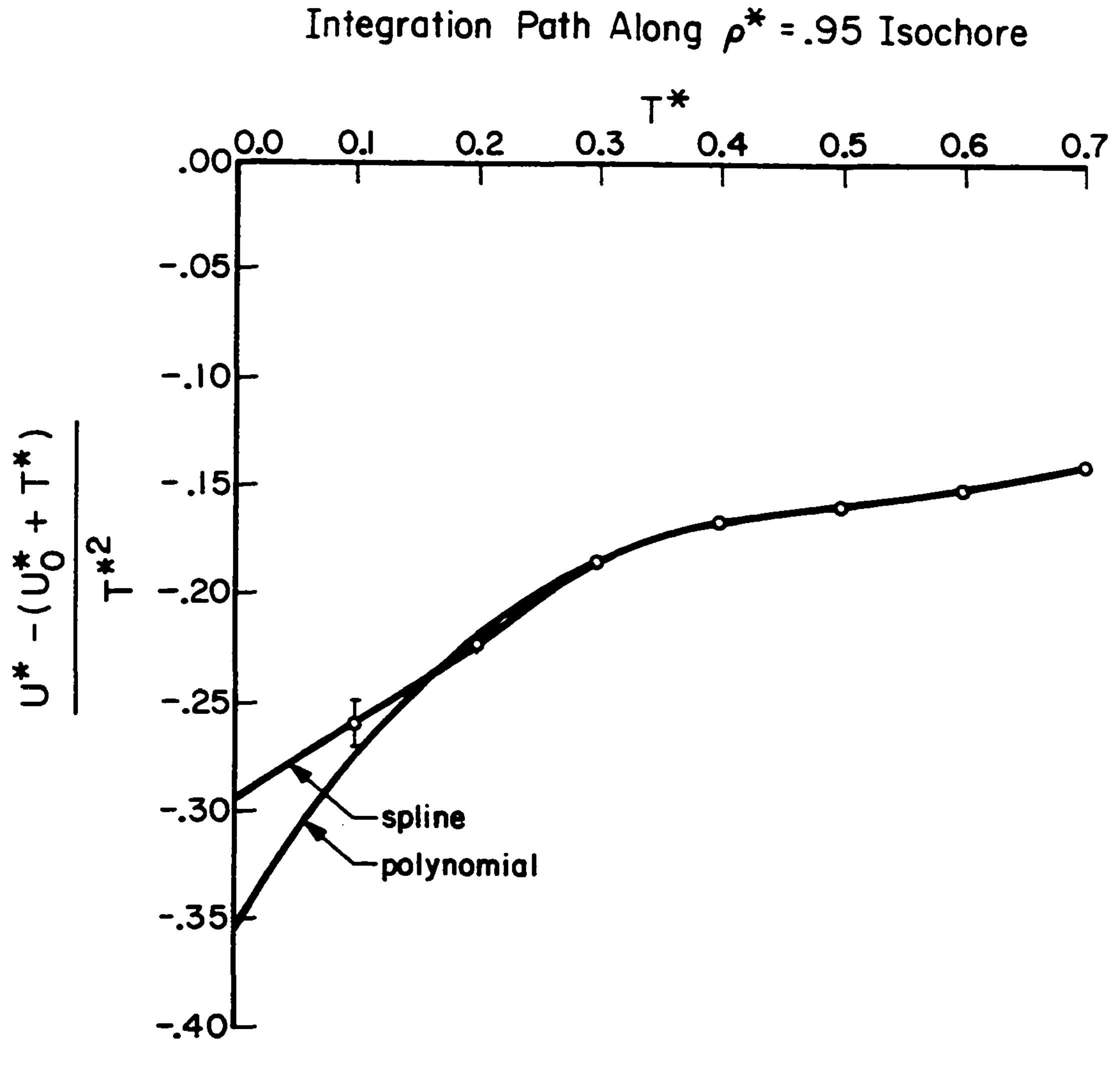


Figure 5.1 The polynomial and spline functions that fit the data points which form the integration path for the free energy along the $\rho^* = .95$ isochore. Only the data point at $T^* = .1$ has a statistical error that is large enough to be shown.

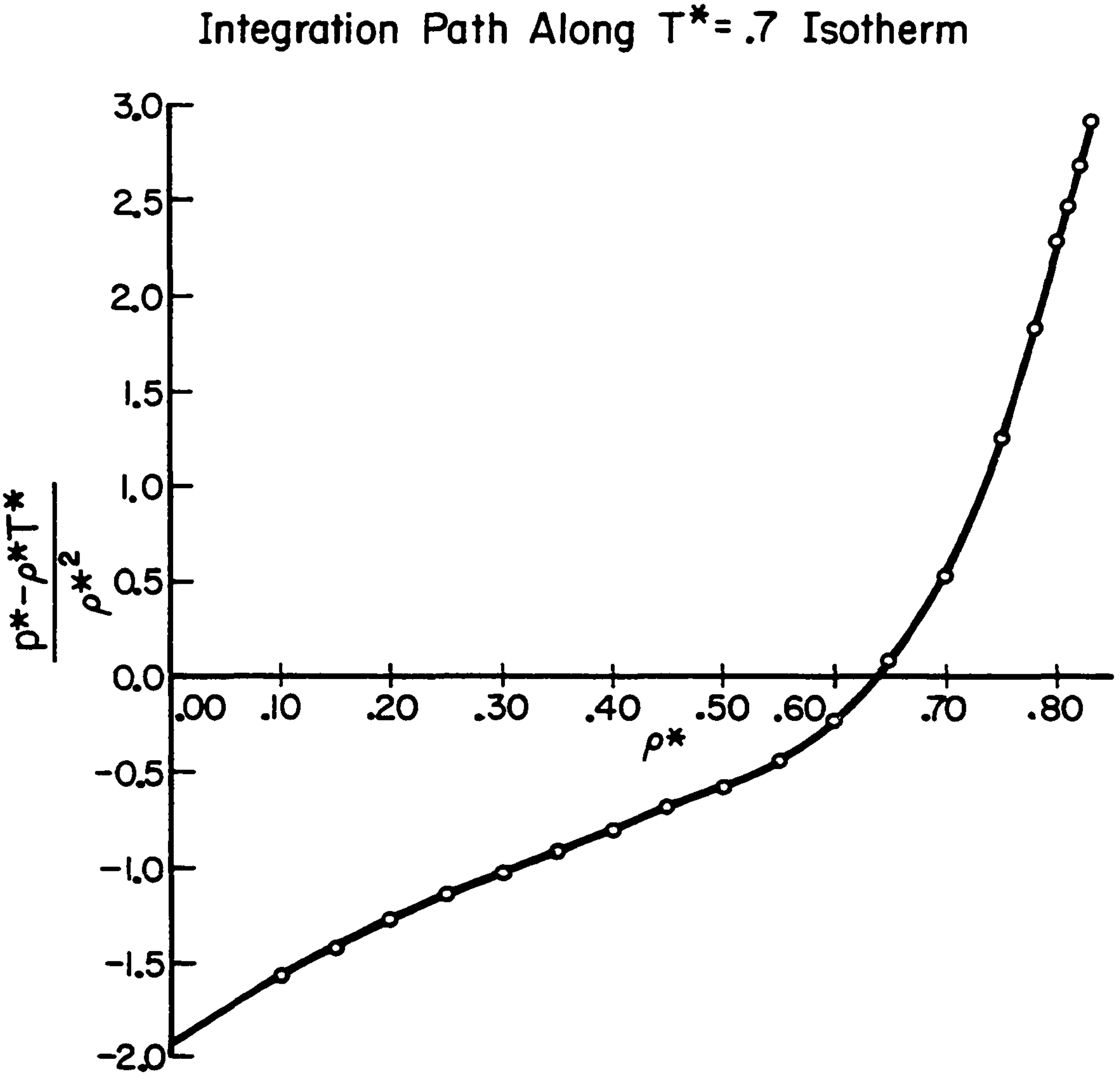


Figure 5.2 The spline function that fits the data points which form the integration path for the free energy along the $T^* = .7$ isotherm. The second virial coefficient gives the exact value of the data point at $\rho^* = 0$.

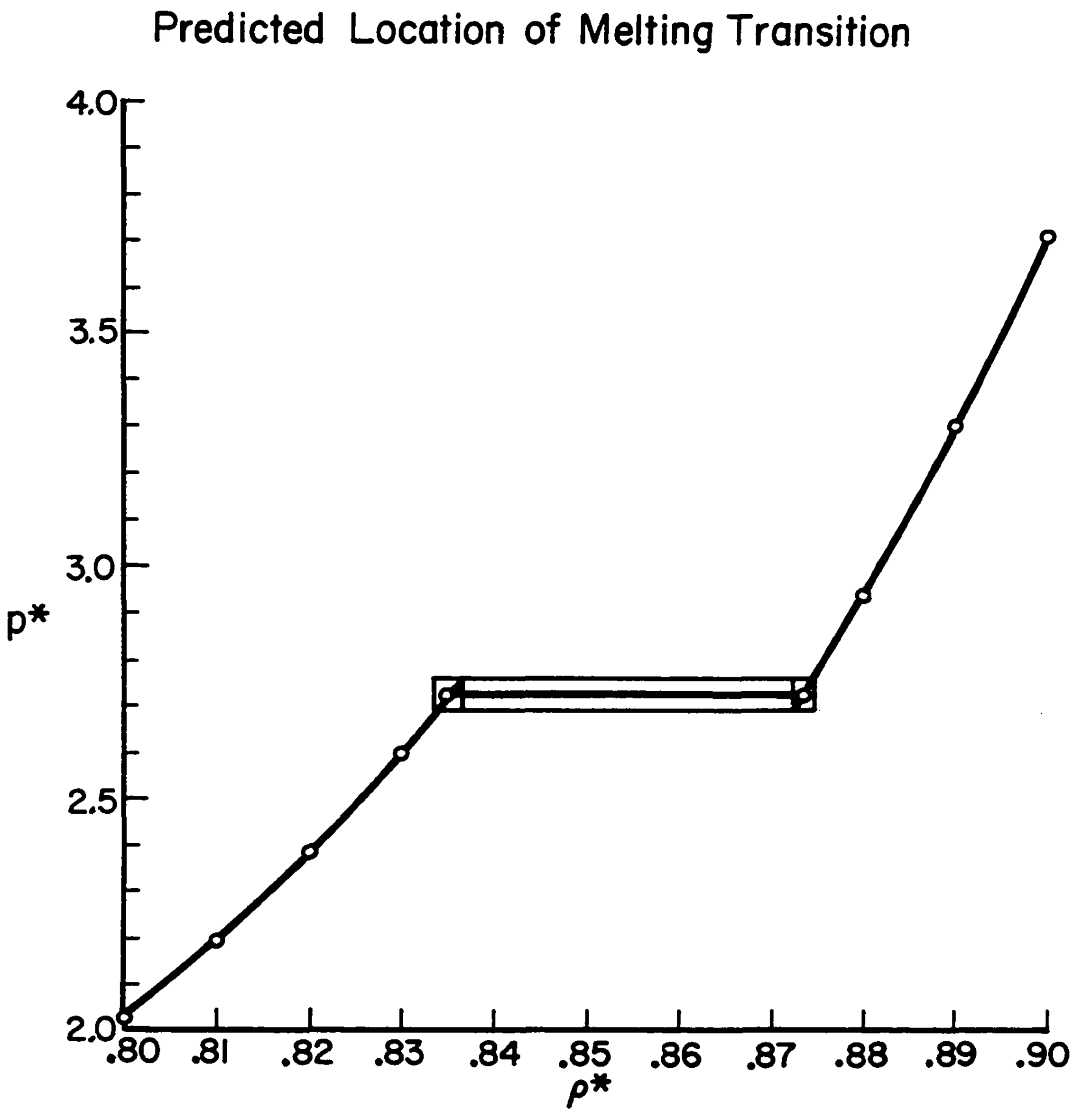


Figure 5.3 A pressure versus density plot showing the predicted location of the melting transition. Portions of the polynomials used to extrapolate the pressure data are shown. The two data points and the heavy line that connect the two polynomials locate ρ_f^*, ρ_s^*, and p_m^*, whose statistical errors are indicated by the lighter lines.

Chapter 6: Direct Simulation of Transition Region

6.1 Constant-Density Simulations

In Section 4.4 we reported that the two-dimensional Lennard-Jones system exhibited a van der Waals loop in its pressure as we thermalized the system at successively lower densities, indicating the presence of a phase transition. In this section we describe the results of simulating the system along the $T^* = .7$ isotherm at densities from $\rho^* = .84$ to $\rho^* = .87$, which seem to lie in the transition region. In the following section we discuss a sensitive method of locating the phase transition that involves simulating the system at constant pressure rather than at constant density. We will compare the results from both methods of directly simulating the system in the transition region to the predictions of the free energy analysis that we presented in the previous chapter.

The first indication of the presence of the transition region occurred when the pressure of the system with $N = 1024$ at $\rho^* = .84$ increased sharply after appearing to be in equilibrium. Figure 6.1 illustrates the rise in pressure in a plot of pressure versus Monte Carlo sweep number, in which each value of pressure is averaged over 400 sweeps. The horizontal lines indicate a range of one standard deviation about the melting pressure that the free energy analysis predicted. About 30,000 sweeps after the system seemed to be in equilibrium, the pressure started increasing towards a new value, which further simulation indicated was relatively stable. In addition to the higher pressure, we observed large fluctuations in the pressure on a time scale of thousands of sweeps. Fluctuations with such a long time scale are unique to the transition region and represent longer term correlations than are present in simulations outside the transition region. The simulations in the transition region typically used about an order of magnitude more Monte Carlo sweeps than other simulations because of the presence of such long time scales. Accordingly, we spent most of our computer time on the few simulations in the transition region that we report in this chapter.

After observing the increase in pressure at $\rho^* = .84$, we performed more update sweeps of the simulations at the higher densities that lie in the transition region. The pressure at $\rho^* = .87$, which was only slightly lower than the the pressure at $\rho^* = .84$, showed little change. At $\rho^* = .85$, the pressure eventually increased to a value near the pressure at $\rho^* = .84$, but the long-term fluctuations were even more pronounced, making an accurate measurement of the pressure impossible. Although the pressure at $\rho^* = .86$ was substantially lower than the pressure at $\rho^* = .84$, it showed little change as it was updated, except for one interesting anomaly. The plot of pressure versus sweep number in Figure 6.2 shows a sudden increase in pressure after about 40,000 sweeps. The system remained at the higher pressure for about 10,000 sweeps, before falling back to the lower pressure. Since the pressure then remained substantially lower than the pressure at $\rho^* = .84$, the van der Waals loop seemed to be stable, which indicated that finite-size effects were dominating the simulation.

In order to reduce the effects of the small size of the system, we decided to increase the number of particles in the system from 1024 to 4096. As described in Section 4.3, we used configurations from the smaller system to initialize simulations of the larger system since the periodic boundary conditions ensure that the edges fit together correctly. The simulations of the larger system exhibited the same type of long-term fluctuations in pressure as the smaller system. Thus, describing the results of the simulations by giving their mean pressures and standard deviations would be misleading for most of the densities. In Figures 6.3 to 6.6 we plot pressure versus sweep number for each of the densities that we simulated in the transition region. We note that although the vertical scales are the same in all of the figures, the horizontal scale is expanded in Figures 6.4 to 6.6 since we performed fewer total sweeps in those simulations. For comparison with the densities in the transition region, Figure 6.7 shows the same type of plot for $\rho^* = .83$, which is not in the transition region. Although the pressure at $\rho^* = .83$ does exhibit the long-term fluctuations, they are much less pronounced than for the simulations in the transition region. Similar plots showed that such fluctuations quickly disappear as the density decreases.

In contrast to the behavior of simulations in the solid and fluid regimes, the pressure at $\rho^* = .86$ in the 4096-particle system shows a considerable change from its behavior in the 1024-particle system, emphasizing that finite-size effects dominated the smaller system. We see from Figure 6.5 that the pressure quickly increased to approximately the same value as observed at other densities in the transition region. Thus, we see no evidence of a stable van der Waals loop in the system with $N = 4096$. However, an indication that the larger system still contains finite-size effects is that the long-term fluctuations in pressure are smaller in amplitude and longer in period, as the difference between Figure 6.1 and Figure 6.3 illustrates. The long-term fluctuations are clearly finite-size effects that using the 4096-particle system reduces but does not eliminate.

The simulation at $\rho^* = .87$, whose results are illustrated in Figure 6.6, appears somewhat anomalous because its long-term fluctuations are much smaller than those at other densities in the transition region. Since $\rho^* = .87$ is near the edge of the transition region, finite-size effects may have been dominating the simulation, even though its pressure is similar to that at the other densities.

The long-term fluctuations in pressure could be evidence of the proposed hexatic phase in which such fluctuations are expected because of critical slowing down near a higher-order transition. However, the fluctuations could also indicate a two-phase region in which channels of fluid separate regions of solid. We would then interpret the fluctuations as normal statistical fluctuations involving changes in the interface separating the solid and fluid regions. The fluctuations would be large because of the relatively small number of such regions and would decrease as the size of the system increases since the number of such regions would also increase.

Table 6.1

Residual Corrections for T^*=.7 and N=4096		
ρ^*	U^*_{cor}	p^*_{cor}
0.87	−.003704±.000043	−.00965±.00011
0.86	−.003309±.000054	−.00852±.00014
0.85	−.002864±.000053	−.00729±.00013
0.84	−.002575±.000068	−.00648±.00017

Since the simulations at $\rho^* = .84$ exhibit several of the long-term fluctuations in pressure, obtaining accurate values of their average pressures and standard deviations from the simulation data appears possible. Before presenting the averages, we report the residual corrections to the potential energy and pressure at each of the densities in the transition region, which we measured with the technique described in Section 4.2. Table 6.1 reports the residual corrections and Figure 6.8 adds the new values of the residual corrections to Figure 4.2. Using the last 80,000 sweeps of those shown in Figure 6.1, we obtained 2.746±0.017 for the corrected pressure of the system with $N = 1024$. For the system with $N = 4096$, we obtained 2.712±0.016 using all 120,000 sweeps shown in Figure 6.3. Although the difference in pressure may not be significant, we would anticipate a lower pressure in the 4096-particle system if the 1024-particle system were on the fluid side of a van der Waals loop that was significantly smaller in the 4096-particle system.

We see that the pressure obtained by directly simulating the 4096-particle system agrees closely with the melting pressure that the free energy analysis predicted. The close agreement with an analysis which assumed that only solid and fluid phases exist and the apparently constant pressure throughout the transition region give strong support to the interpretation that the melting transition is first-order. Although many of the observations, such as the long-term correlations, have ambiguous interpretations, none of the observations contradict interpreting the transition region as a two-phase region. Thus, we interpret the transition region to be a standard two-phase region and not the proposed hexatic phase.

6.2 Constant-Pressure Simulations

Although the results presented in the previous section agreed closely with the results of the free energy analysis, we could not accurately measure the pressure in most of the simulations in the transition region because of their long-term fluctuations. Accurate measurements of the pressure in the transition region would allow us to determine the width of the transition region in pressure and place bounds on the extent of any hexatic region. As we mentioned in Section 1.4, we cannot make an absolute determination of the order of the melting transition in any simulation or experiment. However, by placing bounds on the extent of the transition region, we can quantify the limits of our knowledge about the melting transition. In addition, finding a narrow transition region would give strong support to the interpretation that the melting transition is a first-order phase transition.

Instead of simulating the system at constant density as discussed in the previous section, we attempt to improve our results by simulating the system under conditions of constant pressure. Constant-pressure simulations have several advantages over constant-density simulations near a phase transition, especially if it is first-order. Along an isotherm, a first-order phase transition occurs over the range of densities between the densities of the solid and fluid phases at the transition pressure. In order to determine the width of the melting transition in pressure using constant-density simulations, we must measure small changes in the pressure, which the long-term fluctuations obscure. In contrast, we choose the pressure exactly in a constant-pressure simulation and measure the density, which exhibits a large change across the phase boundary. Thus, constant-pressure simulations should allow much more accurate measurements of the pressure and width of the melting transition.

However, constant-pressure simulations also have a major disadvantage over constant-density simulations since they are much more strongly metastable at a first-order phase boundary. As we discussed in Section 2.3, the source of the metastability is the periodic boundary conditions, which eliminate the surfaces that are necessary for a first-order melting transition to occur gradually. Without surfaces, the whole system must abruptly decrease in density as it crosses the phase boundary since the solid prevents local regions of fluid from forming. Such an abrupt change cannot occur by simply changing a single degree of freedom, such as the length scale, because the solid configuration is not a good approximation to a fluid configuration at the same pressure. Thus, crossing a first-order phase boundary at constant pressure involves changing nearly all of the degrees of freedom in the system, which makes the system strongly metastable.

In contrast to the strong metastability that occurs at the phase boundary in a constant-pressure simulation, the metastability is much weaker when the system enters the transition region in a constant-density simulation. In order for a solid to form a two-phase system by partially melting, the solid must break apart into several pieces so that channels of fluid can form. The solid becomes unstable to the formation of a two-phase system when breaking into domains of greater than average density reduces its potential energy by enough to offset the energy cost of the resulting domain boundaries. Since breaking into such domains mainly affects the particles near the domain boundaries, it involves relatively few degrees of freedom compared to melting the system at constant pressure. Thus, the metastability associated with forming a two-phase region is much weaker than that which occurs at the phase boundary of a first-order transition in a constant-pressure simulation.

By combining the two methods of simulation, we can significantly reduce the effects of metastability near the phase transition and exploit the instability of a two-phase configuration in a constant-pressure simulation. If the pressure that we choose for the simulation is slightly higher than the melting pressure, all of the fluid will freeze and the system will become completely solid. On the other hand, if the pressure is slightly lower than the melting pressure, all of the solid will melt and the system will become completely fluid. We avoid the strong metastability at the phase transition by starting with a configuration that contains both phases, so the proportion of each phase can change continuously during the simulation. If the melting transition is higher-order rather than first-order, the instability does not occur because the transition is not sharp, so the density should change smoothly as a function of pressure. However, we retain the advantage of setting the pressure exactly and looking for a relatively large change in the density across the transition region. Thus, initializing constant-pressure simulations with a configuration from the transition region should allow the location and width of the phase transition to be determined precisely.

Even though initializing the simulation with a two-phase configuration avoids the strong metastability associated with melting a completely solid configuration at constant pressure, it does not eliminate all of the long time scales from the simulation. As we mentioned in Section 4.3, a simulation reaches equilibrium much faster when the system's entropy increases during thermalization than it does when its entropy decreases. Thus, the rate at which a simulation approaches equilibrium is quite slow when its specified pressure is slightly higher than the melting pressure, since the entropy of the system decreases significantly as the fluid freezes into a crystal. Annealing the defects that form where domain boundaries meet involves many degrees of freedom, so it is a slow process.

In contrast, the simulation approaches equilibrium much faster when its specified pressure is slightly less than the melting pressure since the entropy of the system increases as the regions of solid melt. In order to make any increase or decrease in density easily noticeable, the initial configuration should be one that is in equilibrium near the middle of the transition region. Although we may not always be able to determine whether a simulation reaches equilibrium, we will at least know whether the density of the system is increasing or decreasing, which provides bounds on the width of the transition region.

Another source of long time scales that is present when the transition is first-order is the small difference in the Gibbs free energy between the two-phase configuration and the equilibrium state of the system. The difference in free energy from the state of minimum free energy, which defines the equilibrium system, is what drives the simulation towards equilibrium. When the specified pressure is very close to the melting pressure, the difference in free energy is small, so the system may take very long to change significantly from its initial configuration. As successive simulations get closer to the pressure at the phase transition, they will require more Monte Carlo sweeps to reach equilibrium.

In order to determine the location and width of the melting transition accurately, we used a configuration that appeared to be in equilibrium at $T^* = .7$ and $\rho^* = .86$ to initialize a series of constant-pressure simulations of the 4096-particle system at $T^* = .7$. Figures 6.9 to 6.12 illustrate the results from four of the simulations with plots of density versus Monte Carlo sweep number, in which we have plotted the average densities from blocks of 400 sweeps. In addition, the horizontal lines on each of the plots indicate ranges of one standard deviation about the values of ρ_s^* and ρ_f^* that the free energy analysis predicted. From Figure 6.9, we see that the simulation at $p^* = 2.71$ apparently reached an equilibrium density that is slightly higher than the predicted fluid density after about 30,000 sweeps. At $p^* = 2.72$ the simulation required nearly 40,000 sweeps before beginning a definite decrease in density and almost 60,000 sweeps before its density seemed to stop decreasing, as Figure 6.10 illustrates. Although the simulation may not be in equilibrium after 80,000 sweeps, its density clearly decreased to a value close to the predicted fluid density at the phase transition. Thus, we see strong evidence that the melting pressure at $T^* = .7$ is greater than $p^* = 2.72$.

In contrast, Figure 6.11 shows that the simulation at $p^* = 2.73$ neither increased nor decreased substantially in density after 60,000 sweeps. While the system may actually be in equilibrium near its initial density of $\rho^* = .86$, we find such an interpretation unconvincing since the simulation at $p^* = 2.72$ required almost as many updates before its density clearly began to change. We think a more reasonable interpretation is that the pressure is so close to the melting pressure that the free energy driving the simulation towards equilibrium is too small to be apparent in 60,000 sweeps. Regardless of which interpretation is correct, the melting pressure must be very close to $p^* = 2.73$.

From Figure 6.12 we see that the density in the simulation at $p^* = 2.75$ is slowly but definitely increasing. Since the simulation still appears to be approaching equilibrium after 60,000 sweeps, we cannot estimate its equilibrium density, but we expect that it is at least as high as the highest value attained during the simulation. Although the simulation has not reached equilibrium, its increasing density provides strong evidence that the melting pressure is less than $p^* = 2.75$.

Thus, the results of the constant-pressure simulations give further support to our interpretation that the melting transition at $T^* = .7$ is a first-order phase transition. While the simulations closest to the melting pressure are not all in equilibrium, they clearly indicate that a density change of at least $\Delta\rho^* = .025$ occurs between the pressures $p^* = 2.72$ and $p^* = 2.75$. The location of the melting transition is consistent with the results of both the free energy analysis in Chapter 5 and the constant-density simulations in the previous section. In addition, by bounding the region in which the transition occurs, we show that the melting transition is quite abrupt, which is characteristic of a first-order transition.

While we still cannot eliminate the possibility that melting is higher-order, we have placed limits on the width of any region of hexatic phase, unless the simulations have completely missed the hexatic phase due to critical slowing down. Although we cannot completely rule out the possibility that critical slowing down has dominated all of our simulations, such a situation seems very unlikely since the results of many types of simulations give consistent results.

Table 6.2

Densities for T^*=.7 and N=4096		
p^*	ρ^*	sweeps
2.5939	0.82927±0.00044	40000
2.6537	0.83443±0.00043	40000
2.6836	0.83722±0.00062	30000
2.7036	0.83841±0.00040	30000
2.7134	0.84141±0.00045	20000
2.7216	0.85865±0.00044	30000
2.7407	0.86662±0.00048	10000

Table 6.2 reports additional results from the constant-pressure simulations in which we have added the residual corrections to the simulation pressures in order to obtain a better value of the true pressure of each simulation. We calculated the residual corrections by using the measured density of the simulation and linearly interpolating between the values of p^*_{cor} in Tables 4.4 and 6.1. The entries in Table 6.2 also give the number of Monte Carlo sweeps used to calculate the average and standard deviation at each pressure.

For instance, at $p^* = 2.72$, whose corrected value is $p^* = 2.7134$, we used only the last 20,000 sweeps of the 80,000 sweeps illustrated in Figure 6.10 in order to avoid using the configurations are clearly not in equilibrium. Although the simulation at $p^* = 2.75$ does not appear to have reached equilibrium, we used its last 10,000 sweeps to calculate an average density in order to indicate how much its density increased during the simulation.

In Figure 6.13 we have expanded the transition region shown in Figure 5.3 and have turned the plot on its side. The polynomials that give the solid and fluid pressures and the predicted location of the transition and its statistical errors are identical to those in Figure 5.3. The data point at $\rho^* = .83$ with a horizontal error bar and the one at $\rho^* = .88$ are the pressures obtained from the constant-density simulations reported Chapter 4. The constant-density simulation of the 4096-particle system that was reported in the previous section obtained the pressure at $\rho^* = .84$ with the horizontal error bar in Figure 6.13. In addition, we have plotted the results of the constant-pressure simulations as they appear in Table 6.2. Even if some of the simulations did not reach their equilibrium values, they still indicate that the transition region is very narrow in pressure. We see some evidence that the transition is rounded, which could be due to the simulations' not being in equilibrium since the simulations approached equilibrium by moving away from their initial density of $\rho^* = .86$. While the rounding could indicate that the transition is higher-order and therefore continuous, a more likely explanation is that we are seeing finite-size rounding of a first-order transition. As we mentioned in Section 1.4, a first-order transition is discontinuous only in an infinite system and is otherwise rounded by an amount that depends on the number of particles in the system.

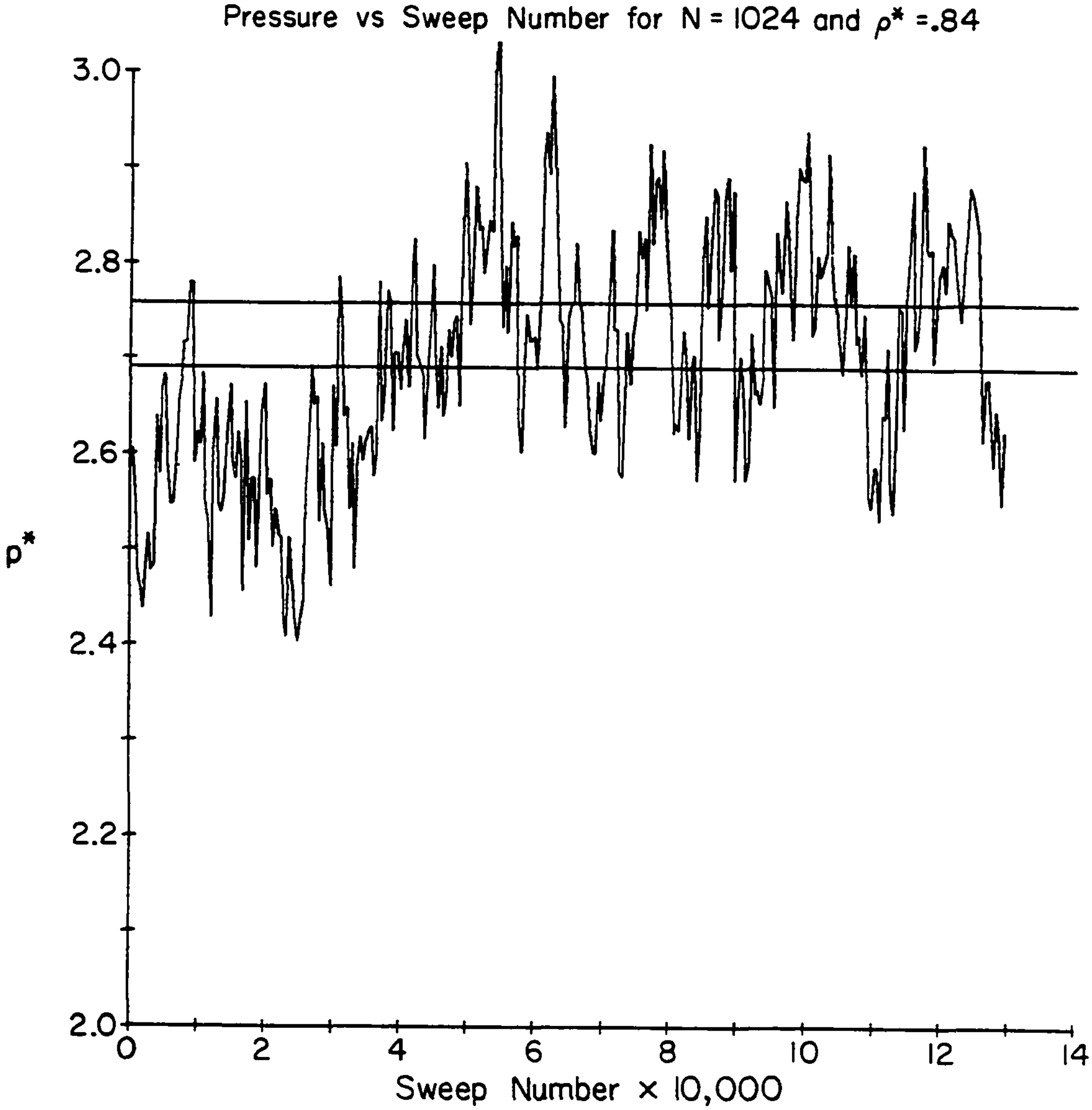


Figure 6.1 A plot of pressure versus sweep number for ρ^* =.84 and N = 1024. Each data point is the average pressure of a block of 400 sweeps. The horizontal lines indicate a range of one standard deviation about the melting pressure that the free energy analysis predicted.

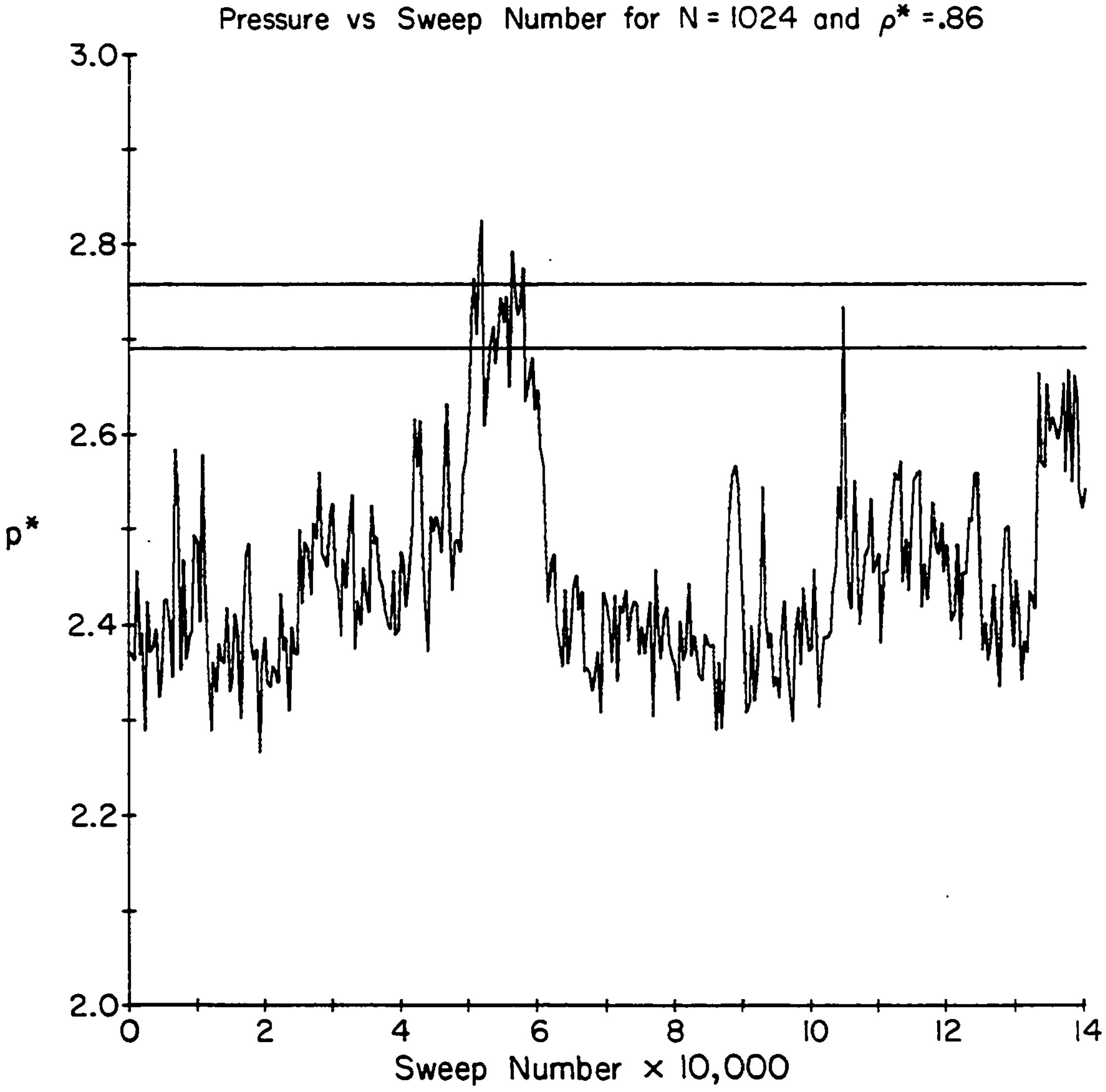


Figure 6.2 A plot of pressure versus sweep number for $\rho^* = .86$ and $N = 1024$. Each data point is the average pressure of a block of 400 sweeps. The horizontal lines indicate a range of one standard deviation about the melting pressure that the free energy analysis predicted.

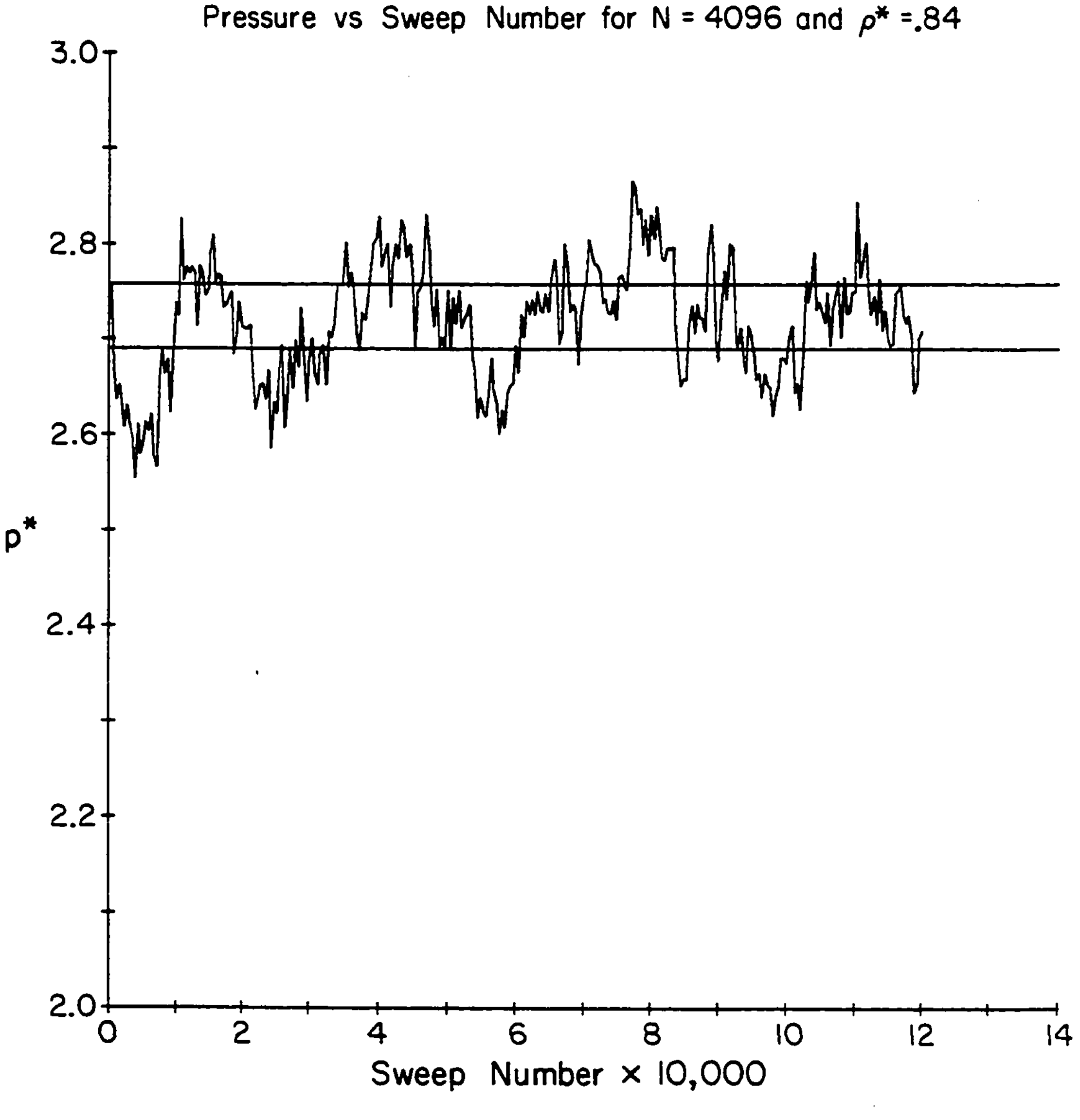


Figure 6.3 A plot of pressure versus sweep number for $\rho^* = .84$ and $N = 4096$. Each data point is the average pressure of a block of 400 sweeps. The horizontal lines indicate a range of one standard deviation about the melting pressure that the free energy analysis predicted.

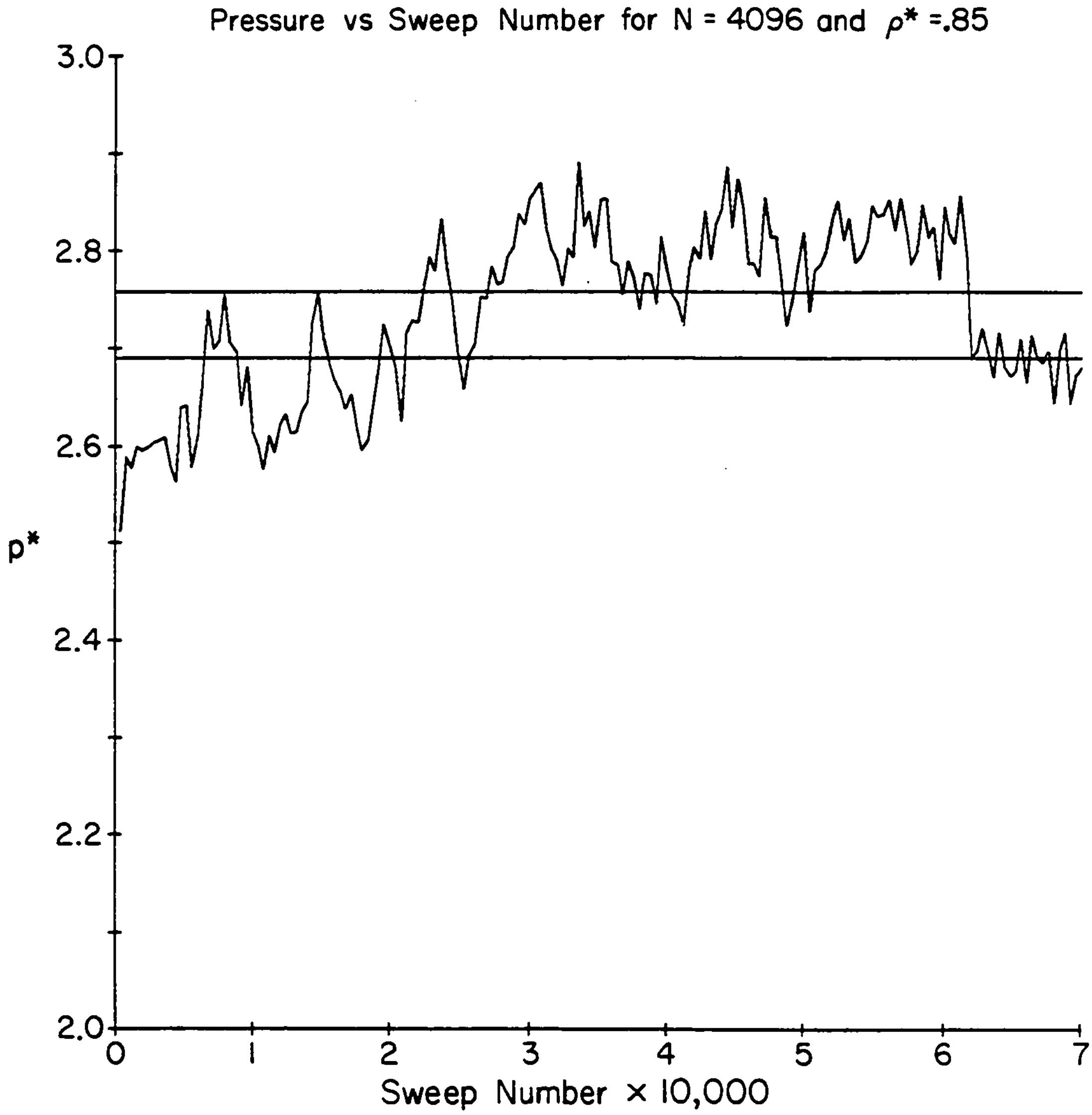


Figure 6.4 A plot of pressure versus sweep number for $\rho^* = .85$ and $N = 4096$. Each data point is the average pressure of a block of 400 sweeps. The horizontal lines indicate a range of one standard deviation about the melting pressure that the free energy analysis predicted.

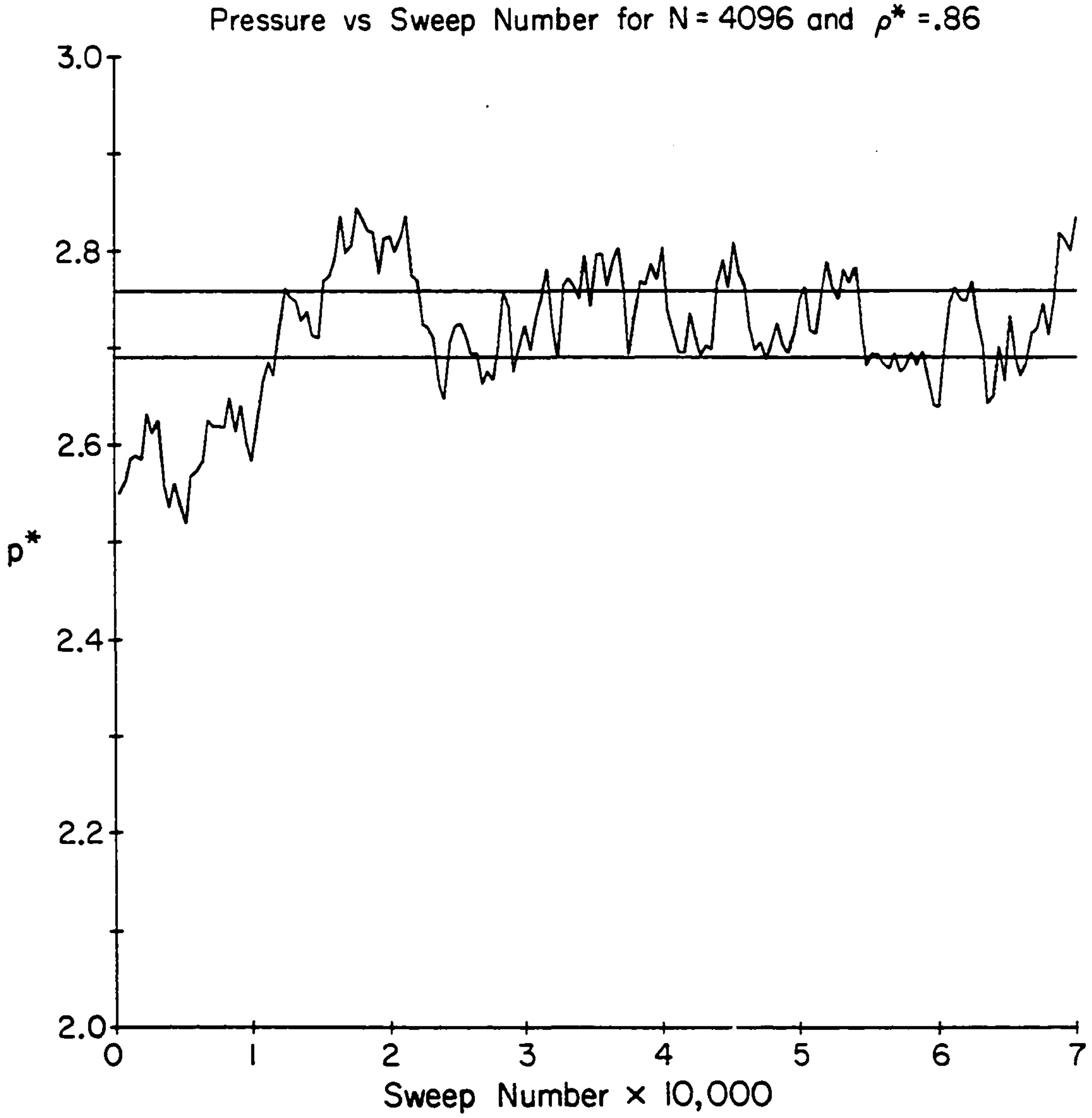


Figure 6.5 A plot of pressure versus sweep number for $\rho^* = .86$ and $N = 4096$. Each data point is the average pressure of a block of 400 sweeps. The horizontal lines indicate a range of one standard deviation about the melting pressure that the free energy analysis predicted.

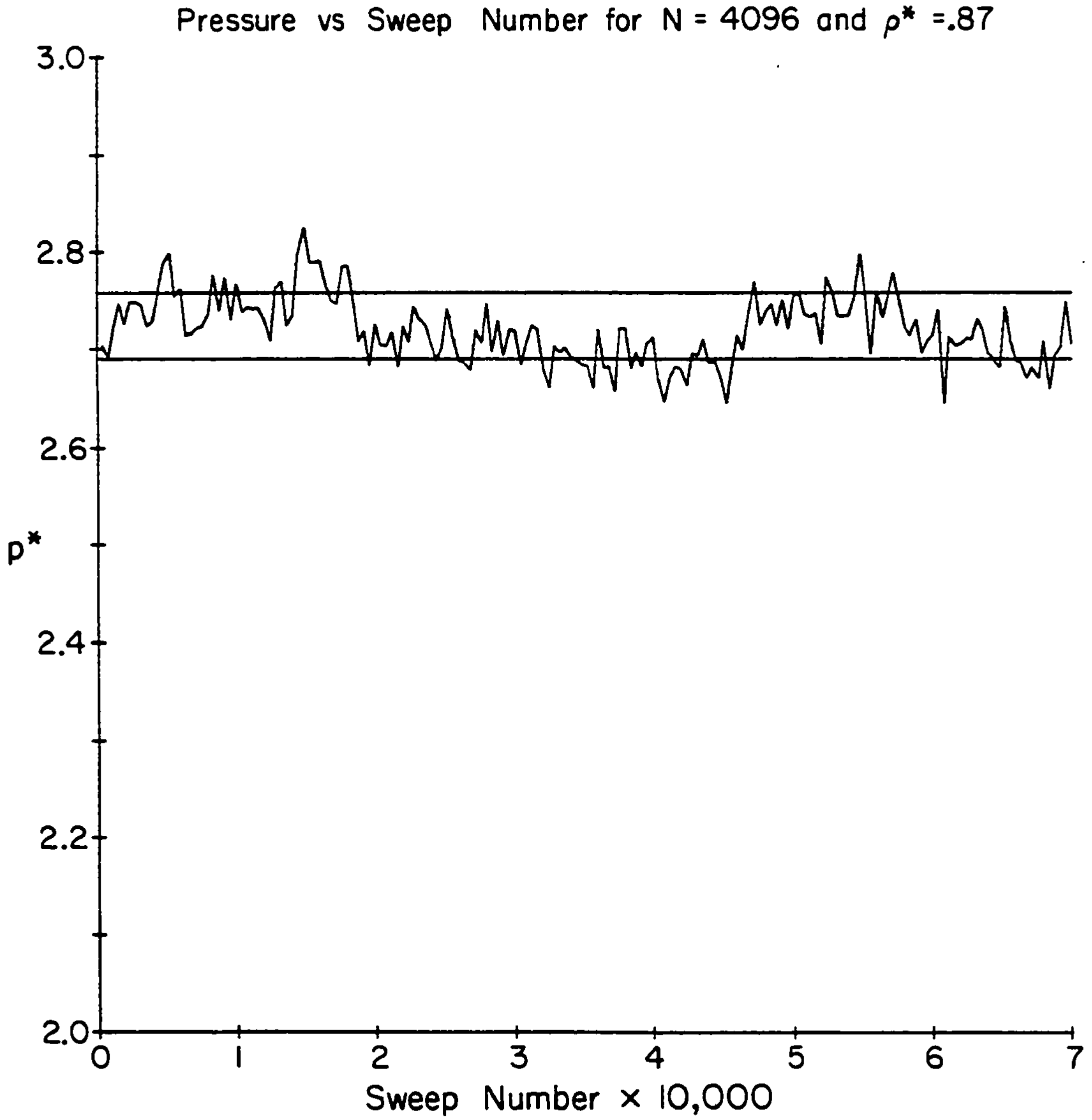


Figure 6.6 A plot of pressure versus sweep number for $\rho^* = .87$ and $N = 4096$. Each data point is the average pressure of a block of 400 sweeps. The horizontal lines indicate a range of one standard deviation about the melting pressure that the free energy analysis predicted.

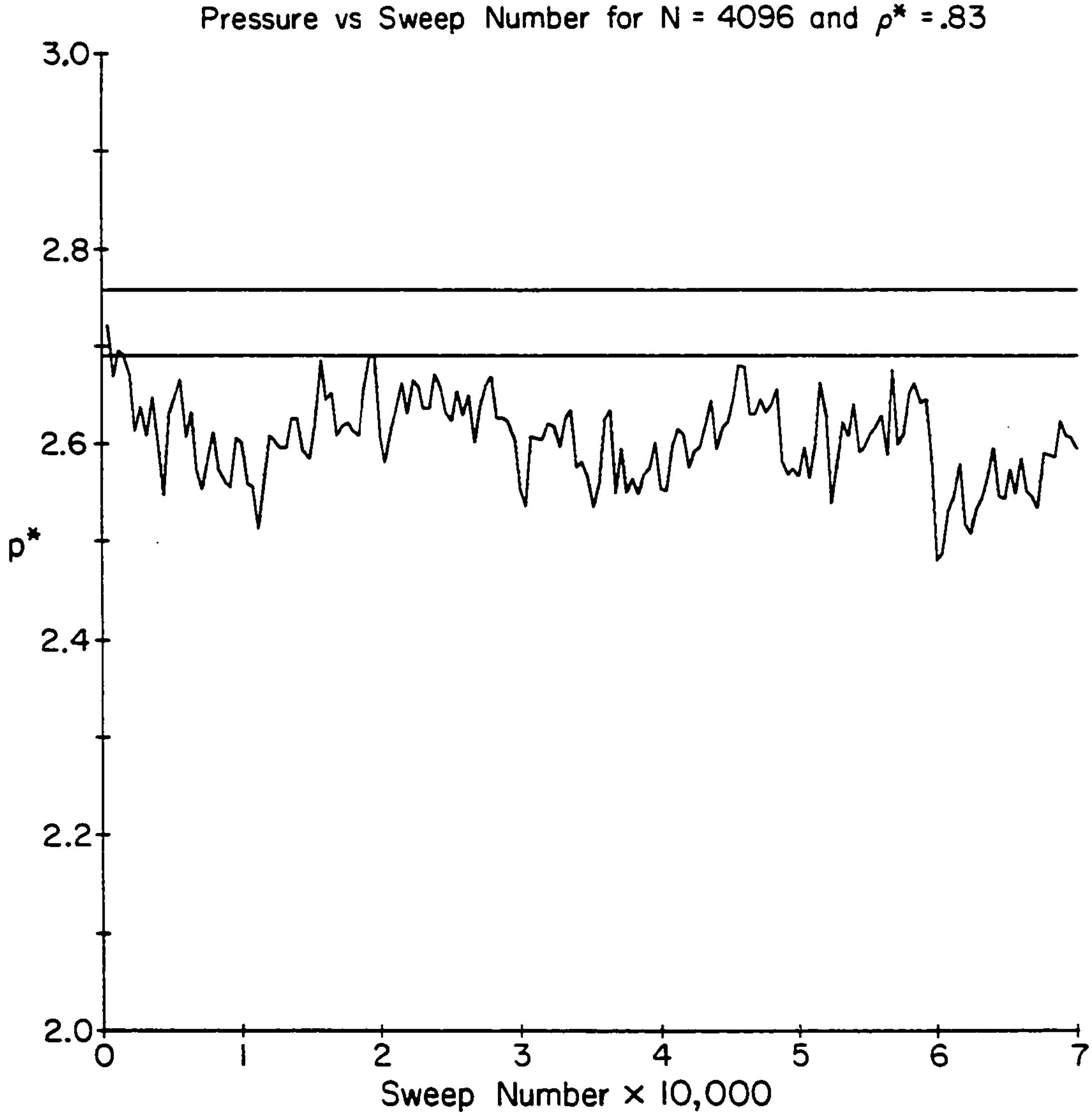


Figure 6.7 A plot of pressure versus sweep number for $\rho^* = .83$ and $N = 4096$. Each data point is the average pressure of a block of 400 sweeps. The horizontal lines indicate a range of one standard deviation about the melting pressure that the free energy analysis predicted.

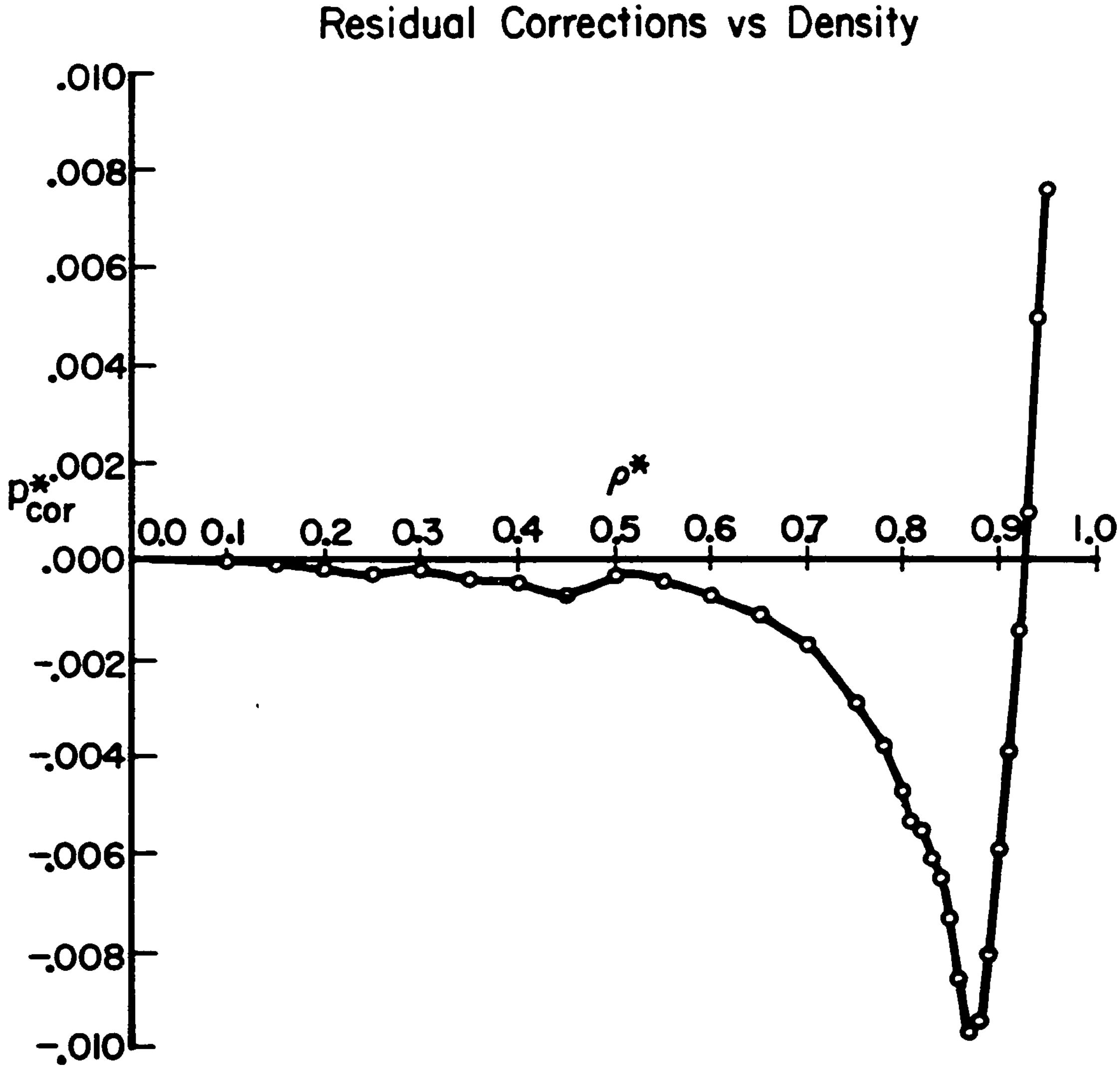


Figure 6.8 A plot of the residual corrections to the pressure versus density. Data points at densities that lie in the transition region have filled the gap that separated the solid and fluid regions in Figure 4.2.

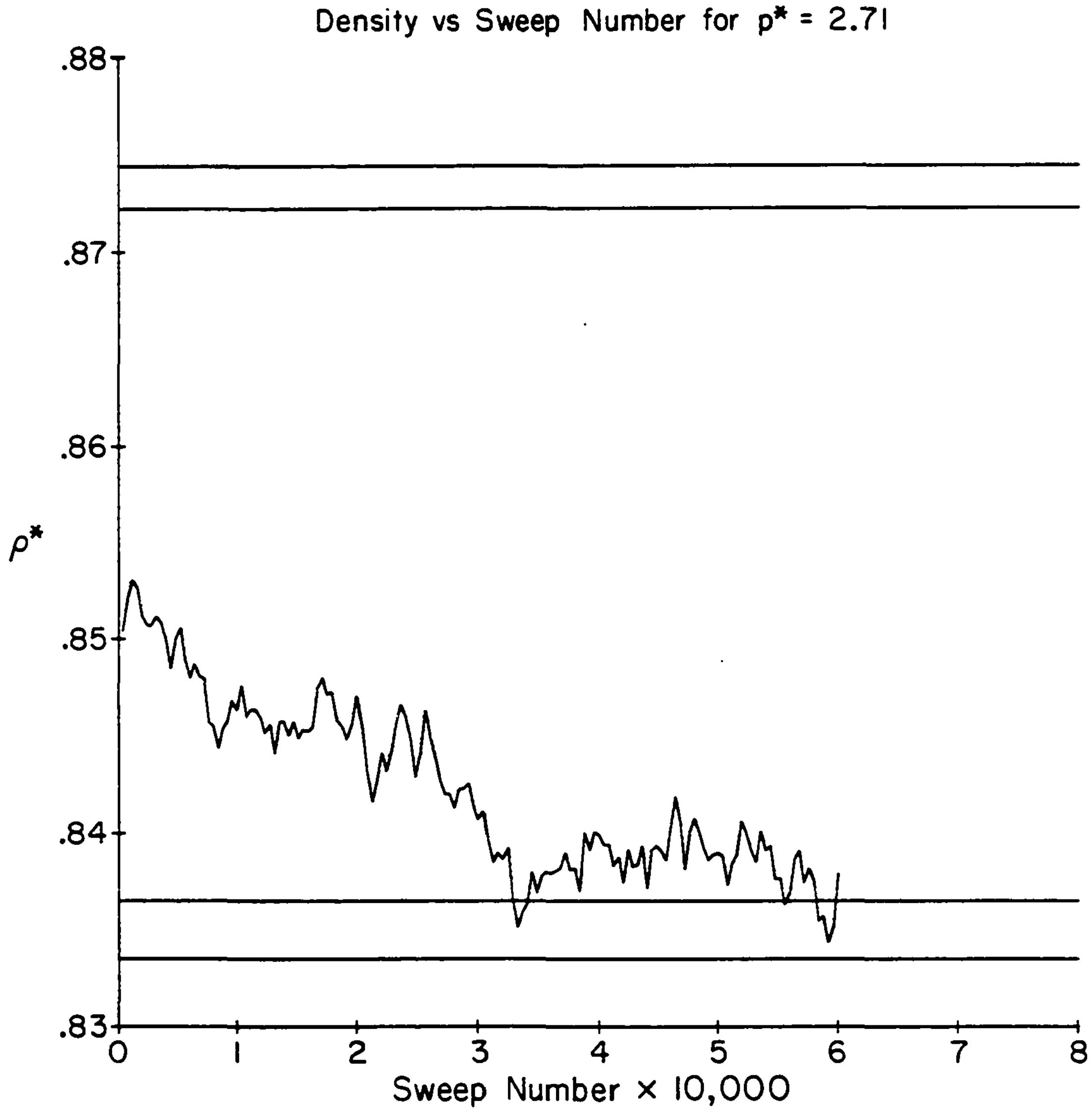


Figure 6.9 A plot of density versus sweep number for $p^* = 2.71$ and $T = .7$. Each data point is the average density of a block of 400 sweeps. The horizontal lines indicate a range of one standard deviation about the values of ρ_s^* and ρ_f^* that the free energy analysis predicted.

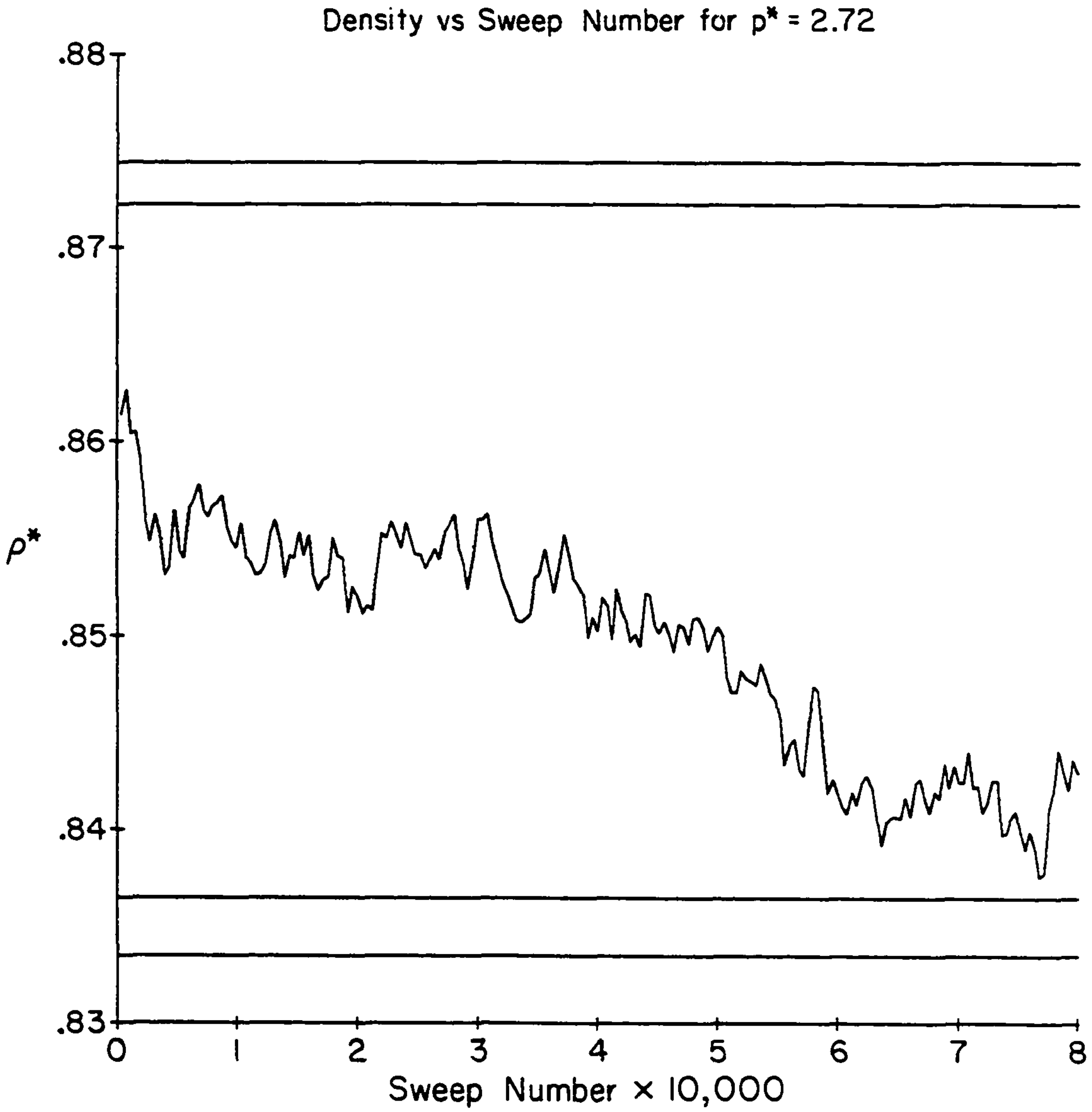


Figure 6.10 A plot of density versus sweep number for $p^* = 2.72$ and $T = .7$. Each data point is the average density of a block of 400 sweeps. The horizontal lines indicate a range of one standard deviation about the values of ρ_s^* and ρ_f^* that the free energy analysis predicted.

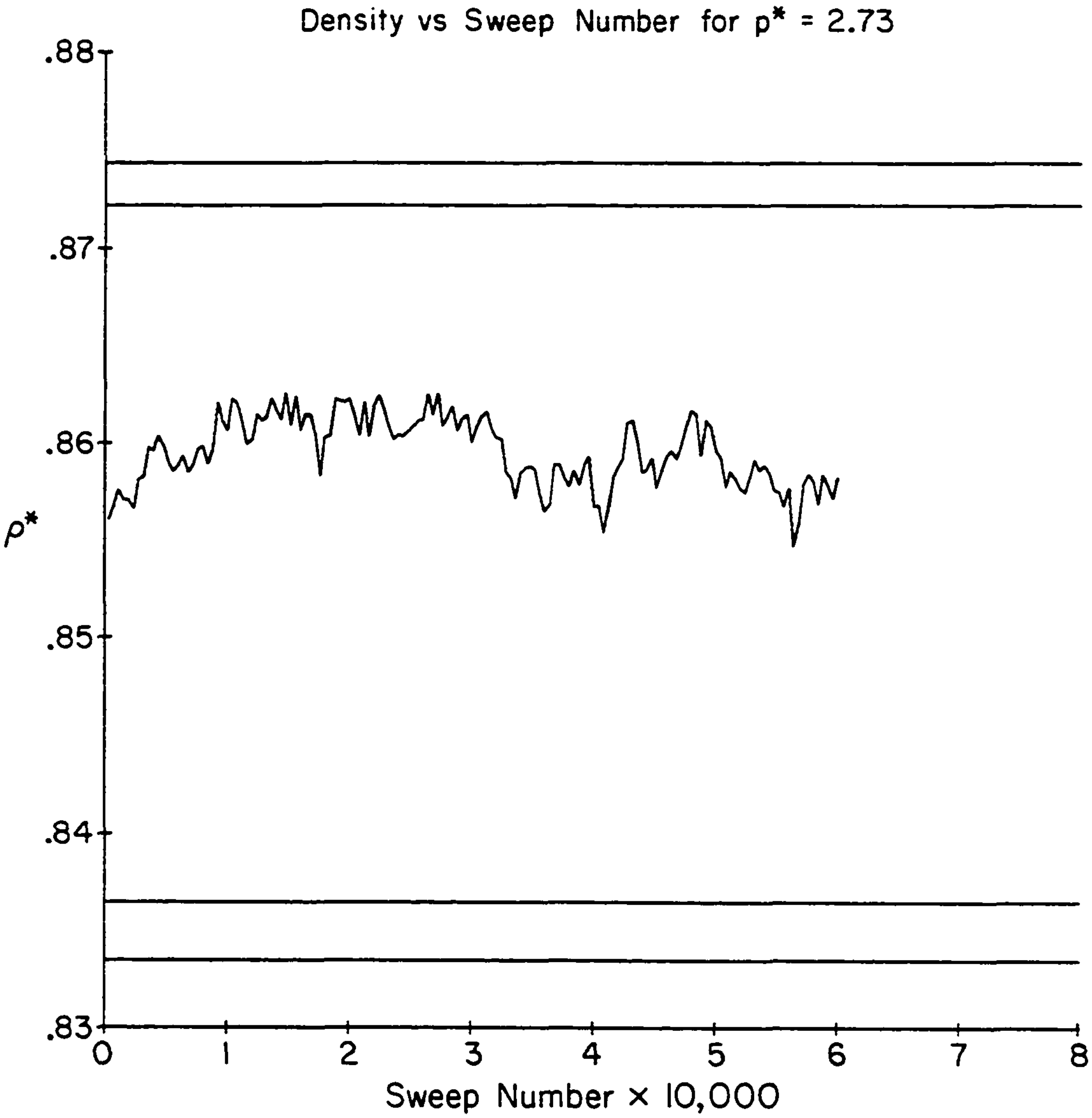


Figure 6.11 A plot of density versus sweep number for $p^* = 2.73$ and $T = .7$. Each data point is the average density of a block of 400 sweeps. The horizontal lines indicate a range of one standard deviation about the values of ρ_s^* and ρ_f^* that the free energy analysis predicted.

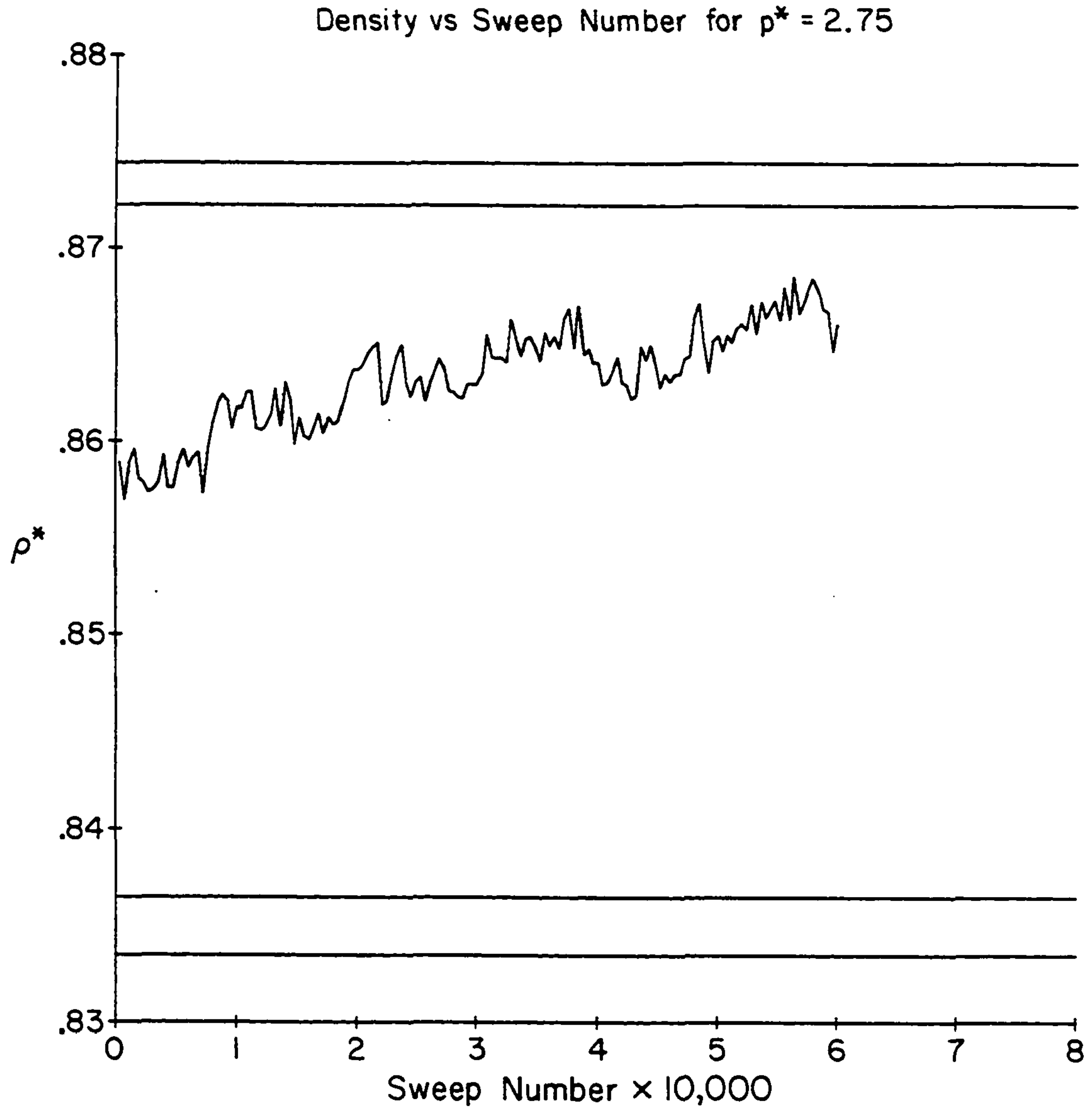


Figure 6.12 A plot of density versus sweep number for $p^* = 2.75$ and $T = .7$. Each data point is the average density of a block of 400 sweeps. The horizontal lines indicate a range of one standard deviation about the values of ρ_s^* and ρ_f^* that the free energy analysis predicted.

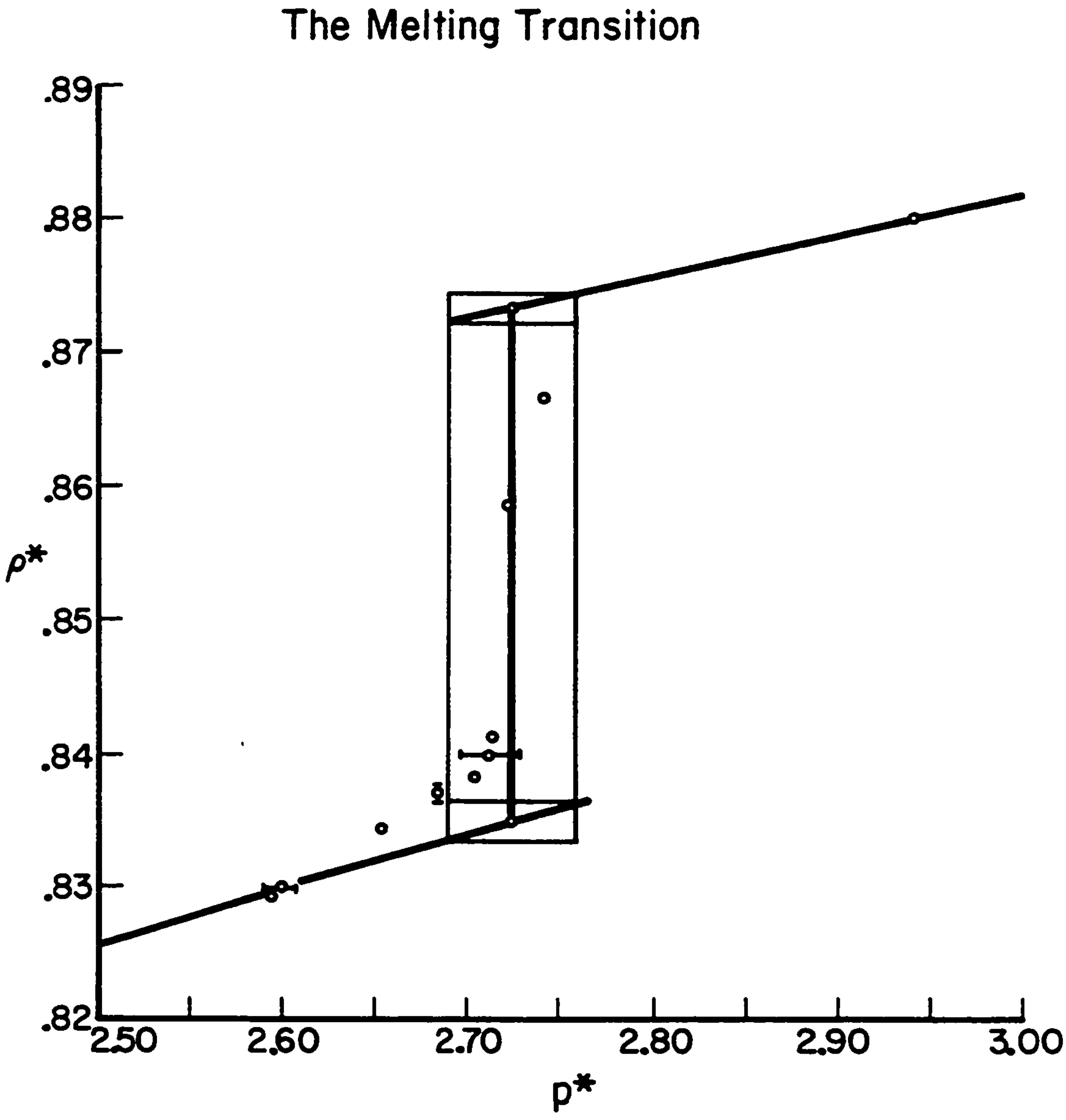


Figure 6.13 A density versus pressure plot that summarizes our results on the melting transition. Portions of the polynomials used to extrapolate the pressure data and the results of the free energy analysis are shown as in Figure 5.3. The data points at $\rho^* = .83$ and $\rho^* = .84$ with horizontal error bars and the data point at $\rho^* = .88$ were obtained in constant-density simulations. The other data points are the results of the constant-pressure simulations.

Chapter 7: Measurement of Elastic Constants

7.1 Derivation of Expressions for Elastic Constants

In the previous chapters we have reported only the results of measuring the potential energy and pressure of the system since we were interested in locating its melting transition. We now expand our investigation and measure the elastic properties of the two-dimensional Lennard-Jones system so that we can test the predictions of Kosterlitz and Thouless. In Section 1.2 we mentioned that Kosterlitz and Thouless[4] established a criterion that specifies the limit of the stability of a solid against the formation of free dislocations. When the Kosterlitz-Thouless K parameter reaches a value of 16π, free dislocations, which prevent the system from supporting a shearing stress, can form and K drops abruptly to zero. Converting the expression for K in (1.1) to reduced quantities gives

$$K = \frac{8}{\sqrt{3}\rho^* T^*} \frac{\mu^*(\mu^* + \lambda^*)}{2\mu^* + \lambda^*} \tag{7.1}$$

where μ and λ are the two Lamé coefficients. Thus, $K(T_m^*) = 16\pi$ defines the temperature T_m^* at which the solid loses its resistance to shear due to the formation of free dislocations. We note that the measured values of the Lamé coefficients are necessarily their renormalized values since the simulation correctly models any interaction between the pairs of thermally excited dislocations.

In order to test whether the two-dimensional Lennard-Jones system melts by becoming unstable to the creation of free dislocations, we will evaluate K near the melting transition. If K is substantially greater than 16π at the melting density along the $T^* = .7$ isotherm, we can conclude that the system melts before reaching the instability predicted by the model of Kosterlitz and Thouless. In such a situation, another mechanism would have to allow the system to melt before the pairs of thermally excited dislocations could unbind.

On the other hand, obtaining a value of $K = 16\pi$ at the melting density would suggest that melting may occur through two higher-order phase transitions that bound a region of hexatic phase. In Section 1.2 we mentioned that results of renormalization group calculations by Halperin and Nelson[7,8] showed that dislocation unbinding leads to the possibility of such a two-step melting process. By measuring K near the melting transition, we intend to determine whether the model of melting on which Halperin and Nelson based their analysis applies to the system that we are investigating.

We prepare to study the elastic properties of the system by deriving expressions that give the isothermal elastic constants of the system in terms of quantities that the Monte Carlo simulation can measure. In the following section we will use the elastic constants to obtain the Lamé coefficients, which we then use to evaluate K. In order to derive the elastic constants, we follow the procedure outlined by Squire et al.,[46] which starts with the partition function for the system at constant density. From Section 2.4 we obtain the partition function, which is

$$Z = \int_0^X dx_1 \cdots dx_N \int_0^Y dy_1 \cdots dy_N \, e^{-\beta\Sigma\phi(r)} \tag{7.2}$$

where $\phi(r)$ is the Lennard-Jones potential and the summation extends over all distinct pairs of particles. Substituting $x_i' = \frac{x_i}{X}$ and $y_i' = \frac{y_i}{Y}$ into (7.2) makes the volume dependence explicit and gives

$$Z = V^N \int_0^1 dx_1' \cdots dx_N' \int_0^1 dy_1' \cdots dy_N' \, e^{-\beta\Sigma\phi(r)} \ . \tag{7.3}$$

In terms of the Helmholtz free energy, which is $F = -kT \ln Z$ by (5.18) in Section 5.2, the definition of the isothermal elastic constants is[47]

$$C_{ijkl}^T \equiv \frac{1}{V} \left[\frac{\partial^2 F}{\partial \eta_{ij} \, \partial \eta_{kl}} \right]_{T\eta'} \tag{7.4}$$

where η_{ij} is the *Lagrangian strain tensor*, and the subscript η' indicates that all other η_j are held constant during differentiation.

Before we can differentiate F with respect to η_{ij}, we must express the distance between two particles in terms of the Lagrangian strain tensor. Deforming a system displaces every point by an amount that depends on the *displacement vector*, $\vec{u}$, which is

$$u_i = X_i - x_i \tag{7.5}$$

where the coordinates of the point are x_i in the undeformed system and X_i in the deformed system. From the displacement vector we define the *displacement gradients*

$$u_{ij} = \frac{\partial u_i}{\partial x_i} \ . \tag{7.6}$$

In our derivation of the elastic constants we will consider only the effects of homogeneous deformations, in which the u_{ij} are constant throughout the system.

If the vector connecting two points is $\vec{r}$ in the undeformed system and $\vec{R}$ in the deformed system, the squared distance between the points is

$$R_i^2 = (r_i + u_{ij} r_j)^2 = r_i^2 + 2u_{ij} r_i r_j + u_{ki} u_{kj} r_i r_j \tag{7.7}$$

where we have used the Einstein summation convention of summing over repeated indices. The Lagrangian strain tensor is defined as[48]

$$\eta_{ij} = \frac{1}{2}(u_{ij} + u_{ji} + u_{ki} u_{kj}) \tag{7.8}$$

where we have written $2u_{ij}$ in the explicitly symmetrical form, $u_{ij} + u_{ji}$. Thus, the distance between two points in the deformed system is

$$R_i^2 = r_i^2 + 2\eta_{ij} r_i r_j \ . \tag{7.9}$$

Having expressed the distance between two particles in terms of the Lagrangian strain tensor, we can obtain a derivative that we will use evaluate the derivatives in (7.4). We let $\vec{r}$ be the vector connecting two particles in the system and $r = |\vec{r}|$ be the distance between them and use (7.9) to obtain

$$\frac{\partial r}{\partial \eta_{ij}} = \frac{r_i r_j}{r} \tag{7.10}$$

in which the derivative is evaluated in the undeformed system. Using (7.10) and the chain rule, we can now differentiate F with respect to η_{ij} to obtain the elastic constants. For now, we ignore the *kinetic term* in the elastic constants, which arises from the factor of V^N in (7.3). Differentiating F once gives

$$\left[\frac{\partial F}{\partial \eta_{ij}}\right]_T = \langle \sum \frac{\phi'(r)}{r} r_i r_j \rangle \tag{7.11}$$

where we have written the derivative as an expectation value over the partition function. We differentiate (7.11) and obtain

$$\begin{aligned}\left[\frac{\partial^2 F}{\partial \eta_{ij} \partial \eta_{kl}}\right]_T &= \beta \langle \sum \frac{\phi'}{r} r_i r_j \rangle \langle \sum \frac{\phi'}{r} r_k r_l \rangle \\ &\quad - \beta \langle (\sum \frac{\phi'}{r} r_i r_j)(\sum \frac{\phi'}{r} r_k r_l) \rangle \\ &\quad + \langle \sum (\frac{\phi''}{r^2} - \frac{\phi'}{r^3}) r_i r_j r_k r_l \rangle \ .\end{aligned} \tag{7.12}$$

We now evaluate the contribution to the elastic constants from the factor of V^N in (7.3). In order to differentiate the additive term of $-kTN \ln V$ in the free energy, we first need the derivative of V with respect to η_{ij}. We note that the Lagrangian strain tensor is symmetric, so an appropriate rotation of the coordinate axes will transform it into diagonal form.[48] The two mutually perpendicular basis vectors in the rotated coordinate system are its *principal axes*, along which the deformation is only pure compression or pure extension. Thus, the elements of length along the principal axes, which are dx_i in the undeformed system, become $dx_i' = \sqrt{1 + 2\eta^{(i)}}dx_i$ in the deformed system, where $\eta^{(i)}$ are the *principal values* of η_{ij}. The volume element, which is $dV = dx_1 dx_2$ in the undeformed system, becomes

$$dV' = \sqrt{1 + 2\eta^{(1)} + 2\eta^{(2)} + 4\eta^{(1)}\eta^{(2)}}\, dV \tag{7.13}$$

in the deformed system.

We obtain the two principal values of the Lagrangian strain tensor by solving

$$\det \begin{bmatrix} a - \eta^{(i)} & b \\ b & c - \eta^{(i)} \end{bmatrix} = 0 \tag{7.14}$$

where $a = \eta_{11}$, $b = \eta_{12} = \eta_{21}$, and $c = \eta_{22}$. Solving (7.14) gives

$$\eta^{(i)} = \tfrac{1}{2}(a + c) \pm \tfrac{1}{2}\sqrt{(a - c) + 4b^2}\ . \tag{7.15}$$

Substituting the principal values given by (7.15) into (7.13) gives the volume element in the deformed system, which is

$$dV' = dV[1 + 2(a + c) + 4(ac - b^2)]\ . \tag{7.16}$$

In terms of η_{ij}, (7.16) becomes

$$dV' = dV[1 + \eta_{ii} - \eta_{ij}^2 + \tfrac{1}{2}\eta_{ii}^2] \tag{7.17}$$

where η_{ij}^2 is the sum of the squared components and η_{ii}^2 is the squared sum of the diagonal components.

Using (7.17) we can now calculate the derivative of V' with respect to η_{ij} and obtain

$$\frac{dV'}{d\eta_{ij}} = V\,(\delta_{ij} - 2\eta_{ij} + \delta_{ij}\eta_{kk}) \qquad (7.18)$$

in which the derivative is evaluated in the undeformed system. Differentiating $-kTN \ln V'$ to obtain the kinetic contribution to the elastic constants gives

$$-kTN\frac{\partial \ln V'}{\partial \eta_{ij}} = -kTN\,\frac{1}{V'}V\,(\delta_{ij} - 2\eta_{ij} + \delta_{ij}\eta_{kk}) \ . \qquad (7.19)$$

Differentiating a second time gives

$$-kTN\frac{\partial^2 \ln V'}{\partial \eta_{ij}\partial \eta_{kl}} = kTN(\delta_{ik}\,\delta_{jl} + \delta_{il}\,\delta_{jk}) \qquad (7.20)$$

where we have written the result in explicitly symmetric form. Thus, the final form of the elastic constants is

$$\begin{aligned} C^T_{ijkl} &= \frac{1}{kTV}\langle\sum \frac{\phi'}{r}r_i r_j\rangle\langle\sum \frac{\phi'}{r}r_k r_l\rangle \\ &- \frac{1}{kTV}\langle(\sum \frac{\phi'}{r}r_i r_j)\,(\sum \frac{\phi'}{r}r_k r_l)\rangle \\ &+ \frac{1}{V}\langle\sum\,(\frac{\phi''}{r^2} - \frac{\phi'}{r^3})\,r_i r_j r_k r_l\rangle + \frac{kTN}{V}(\delta_{ik}\,\delta_{jl} + \delta_{il}\,\delta_{jk}) \ . \end{aligned} \qquad (7.21)$$

7.2 The Effective Lamé Coefficients

The elastic constants form a fourth rank tensor, C_{ijkl}, which has $2^4 = 16$ components in two dimensions, but not all 16 of the components are independent. From the definition of the elastic constants in (7.4) and the symmetry of the Lagrangian strain tensor, we see that the elastic constants have complete Voigt symmetry. The Voigt symmetry means that the elastic constants are invariant when the ij or kl indices are exchanged and when the pairs ij and kl are exchanged.[47] For instance, we see that

$$C_{ijkl} = C_{jikl} = C_{ijlk} = C_{klij} \tag{7.22}$$

Applying the Voigt symmetry reduces the number of independent components to six. By exploiting the symmetries of the triangular lattice, we will show that the number of nonzero, independent components is only two.

We first use the inversion symmetry of the triangular lattice to show that two of the elastic constants are zero. Since the triangular lattice is invariant under the transformations $x \rightarrow -x$ and $y \rightarrow -y$, C_{ijkl} must also be invariant under the same transformations. The elements of a tensor transform like the product of the corresponding coordinates,[49] so the components of C_{ijkl} in which x or y appear an odd number of times change sign under at least one of the transformations and must be zero. Thus, we see that $C_{xxxy} = C_{xyyy} = 0$.

Using the rotational symmetry of the triangular lattice, we will now show that only two of the remaining four elastic constants are independent. Under rotation the elements of a fourth rank tensor transform as[50]

$$C_{ijkl} = a_{im}\, a_{jn}\, a_{ko}\, a_{lp}\, C_{mnop} \tag{7.23}$$

where a_{ij} is the orthogonal rotation matrix of the transformation. A triangular lattice is invariant under a rotation of $\frac{\pi}{3}$, so the elastic constants must also be invariant under such a rotation. The orthogonal matrix that describes a rotation of $\frac{\pi}{3}$ is

$$a_{ij} = \begin{bmatrix} \cos\frac{\pi}{3} & \sin\frac{\pi}{3} \\ -\sin\frac{\pi}{3} & \cos\frac{\pi}{3} \end{bmatrix} . \tag{7.24}$$

Transforming the C_{xxxx} component of the elastic tensor with the rotation matrix (7.24) and applying the invariance of the elastic constants give

$$C_{xxxx} = c^4 C_{xxxx} + 2c^2 s^2 C_{xxyy} + 4c^2 s^2 C_{xyxy} + s^4 C_{yyyy} \tag{7.25}$$

where we have used $c = \cos\frac{\pi}{3}$ and $s = \sin\frac{\pi}{3}$ for convenience. Transforming the C_{yyyy} component in the same way gives

$$C_{yyyy} = s^4 C_{xxxx} + 2c^2 s^2 C_{xxyy} + 4c^2 s^2 C_{xyxy} + c^4 C_{yyyy} \ . \tag{7.26}$$

By subtracting (7.26) from (7.25) we obtain

$$C_{xxxx} - C_{yyyy} = (c^4 - s^4)(C_{xxxx} - C_{yyyy}) \tag{7.27}$$

which gives $C_{xxxx} = C_{yyyy}$ since $\cos^4\frac{\pi}{3} - \sin^4\frac{\pi}{3} \neq 1$. Substituting $C_{xxxx} = C_{yyyy}$ into (7.25) and using $\cos^2\theta + \sin^2\theta = 1$ gives

$$C_{xxxx} = C_{yyyy} = C_{xxyy} + 2C_{xyxy} \ . \tag{7.28}$$

Thus, two elastic constants completely describe the elastic properties of the two-dimensional hexagonal crystal. Since two elastic constants also describe an isotropic solid, we see that the elastic properties of the two-dimensional hexagonal crystal are isotropic. Using (7.21) to obtain expressions for the two independent elastic constants gives

$$C^T_{xyxy} = -\frac{1}{kTV}\left[\left\langle\left(\sum \frac{\phi'}{r} xy\right)^2\right\rangle - \left\langle\sum \frac{\phi'}{r} xy\right\rangle^2\right]$$

$$+ \frac{1}{V}\left\langle\sum\left(\frac{\phi''}{r^2} - \frac{\phi'}{r^3}\right) x^2 y^2\right\rangle + \frac{kTN}{V} \tag{7.29}$$

and

$$C^T_{xxyy} = -\frac{1}{kTV}[\langle(\Sigma\,\frac{\phi'}{r}x^2)\,(\Sigma\,\frac{\phi'}{r}y^2)\rangle - \langle\Sigma\,\frac{\phi'}{r}x^2\rangle\langle\Sigma\,\frac{\phi'}{r}y^2\rangle]$$

$$+\frac{1}{V}\langle\Sigma\,(\frac{\phi''}{r^2}-\frac{\phi'}{r^3})\,x^2y^2\rangle\ . \qquad (7.30)$$

We now derive expressions for the Lamé coefficients in terms of the elastic constants in (7.29) and (7.30) so that we can evaluate the Kosterlitz-Thouless K parameter. Landau and Lifshitz[6] define the Lamé coefficients, μ and λ, as

$$\frac{F}{V} = \frac{F_0}{V} + \frac{1}{2}\lambda\eta_{ii}^2 + \mu\eta_{ij}^2 \qquad (7.31)$$

where F_0 is the Helmholtz free energy of the undeformed system and η_{ij} is the Lagrangian strain tensor of the deformation. Since (7.31) is a power series expansion of the free energy about the undeformed crystal, it contains no linear term in η_{ij}. Performing a similar expansion in terms of the elastic constants using (7.4) gives

$$\frac{F}{V} = \frac{F_0}{V} + \frac{1}{2}C^T_{xxxx}\ \eta_{xx}^2\ + \frac{1}{2}C^T_{yyyy}\,\eta_{yy}^2 + 2C^T_{xyxy}\ \eta_{xy}^2\ + C^T_{xxyy}\ \eta_{xx}\ \eta_{yy} \qquad (7.32)$$

which (7.28) reduces to

$$\frac{F}{V} = \frac{F_0}{V} + \frac{1}{2}C^T_{xxyy}\ \eta_{ii}^2 + C^T_{xyxy}\ \eta_{ij}^2\ . \qquad (7.33)$$

Equating the corresponding coefficients in (7.31) and (7.33) gives $\lambda = C^T_{xxyy}$ and $\mu = C^T_{xyxy}$.

However, we do not measure the elastic constants in an undeformed crystal since the pressure of the Lennard-Jones solid at $T^* = .7$ is nonzero. Thus, the effective Lamé coefficients that we need to evaluate the Kosterlitz-Thouless K parameter are the values of the Lamé coefficients at finite pressure. Instead of expanding the free energy about the undeformed crystal as in (7.31), we expand the free energy about the equilibrium state of the crystal at the temperature and density of the simulation. Thus, the linear term in η_{ij} does not vanish and we get

$$\frac{F}{V} = \frac{F_0}{V} + \tau'_{ij}\eta_{ij} \tag{7.34}$$

where $\tau'_{ij} = \frac{1}{V}\frac{\partial F}{\partial \eta_{ij}}$ is the stress tensor[47] that represents the externally applied stress that maintains the system at the specified density. Differentiating τ'_{ij} to obtain the second-order terms in the expansion of the free energy gives

$$\frac{F}{V} = \frac{F_0}{V} + \tau_{ij}\eta_{ij} + \tfrac{1}{2}B^T_{ijkl}\,\eta_{ij}\eta_{kl} \tag{7.35}$$

where we define $B^T_{ijkl} \equiv \left[\frac{\partial \tau'_{ij}}{\partial \eta_{kl}}\right]_T$. In (7.35) both τ_{ij} and B^T_{ijkl} are evaluated in the equilibrium state of the crystal, where $\eta_{ij} = 0$.

Wallace[47] derives the stress-strain derivatives B^T_{ijkl} by expanding the stress tensor for one configuration in terms of the stress tensor for another configuration and the Lagrangian strain tensor that relates the two configurations. The stress-strain derivatives that he obtains are

$$B^T_{ijkl} = \tfrac{1}{2}(\tau_{il}\delta_{jk} + \tau_{jl}\delta_{ik} + \tau_{ik}\delta_{jl} + \tau_{jk}\delta_{il} - 2\tau_{ij}\delta_{kl}) + C^T_{ijkl} \tag{7.36}$$

which differ from the isothermal elastic constants by terms involving the applied stress. Since τ_{ij} is a symmetric tensor, B^T_{ijkl} has the same symmetry properties as C^T_{ijkl}, so (7.35) leads an equation that is similar to (7.33) with the stress-strain derivatives in place of the elastic constants. Since the two-dimensional hexagonal crystal is isotropic, the externally applied stress is the same as that on a fluid at pressure p, which is

$$\tau_{ij} = -p\delta_{ij} \ . \qquad (7.37)$$

Expressing the Lamé coefficients in terms of B^T_{ijkl} and substituting (7.37) into (7.36) give

$$\lambda = B^T_{xxyy} = C^T_{xxyy} + p \qquad (7.38)$$

and

$$\mu = B^T_{xyxy} = C^T_{xyxy} - p \qquad (7.39)$$

for the effective Lamé coefficients at nonzero pressure.

7.3 Simulation Results

In order to obtain the values of C^T_{xyxy} and C^T_{xxyy} from the simulation of the two-dimensional Lennard-Jones system, we measured the four sums that appear in (7.29) and (7.30) for many configurations. After every tenth Monte Carlo sweep we calculated $\sum \frac{\phi'}{r}xy$, $\sum \frac{\phi'}{r}x^2$, $\sum \frac{\phi'}{r}y^2$, and $\sum [\frac{\phi''}{r^2} - \frac{\phi'}{r^3}]x^2y^2$. Although the Monte Carlo update algorithm provides the sums that enter the expressions for the potential energy and pressure, it does not provide the sums needed to calculate the elastic constants. Thus, we must calculate the four sums over all pairs of particles whose separation is less than the range of the truncated potential. Since successive configurations in the simulation are highly correlated, we lose little information by calculating the sums for one configuration out of ten and we slow the simulation only slightly.

Using the values of the four measurements, we can construct the expectation values that enter (7.29) and (7.30). Following the example of Squire et al,[46] we report the contributions of the various terms separately. The first terms in (7.29) and (7.30) have the forms of a variance and a correlation, respectively, so both of the terms arise from fluctuations in the data and are known as *fluctuation terms*. Squire et al. refer to the second term as the

Born term since it is the term derived by Born,[51] who confined his results to the case of perfect crystals at absolute zero. The third term in (7.29), which is absent in (7.30), is the *kinetic term*, which enters through the V^N term in the partition function. Although calculating the Born and kinetic terms is simple, evaluating the fluctuation term is more difficult since the correlation of successive measurements affects its average value, not just its standard deviation as in the other measurements.

When we calculate the fluctuation terms by using all of the measurements together, we obtain the best estimates of the terms, but we have no information about their standard deviations. We can measure their standard deviations by grouping the measurements into blocks and averaging the values of the fluctuation terms calculated from each block. Although we obtain both average values and their standard deviations when we average the results from each block, inaccurate values of the fluctuation terms result if the blocks contain too few measurements. The averages that enter the fluctuation terms in (7.29) and (7.30) minimize the size of the fluctuations since the mean of a normal distribution is the value that minimizes its standard deviation. Thus, the blocks must contain enough measurements to ensure that the correlation of successive measurements does not bias the averages and thereby reduce the magnitude of the fluctuation terms.

In order to determine the effect of the correlations, we calculated the fluctuation terms using different numbers of measurements per block. As the size of the blocks increased, the fluctuation terms also increased in magnitude until they reached their true values. However, the number of blocks decreases with increasing block size, so the error estimates of the fluctuation terms become less precise. Table 7.1 reports the averages and standard deviations of the fluctuation terms, denoted by f_{xyxy} and f_{xxyy} , which we calculated with 500 measurements at $T^* = .7$ and $\rho^* = .95$ using the indicated blocking sizes. We see that the fluctuation terms apparently reached their limiting values when each block contained about 125 measurements, so the 500 measurements appear to give accurate

Table 7.1

Fluctuation Terms from 5,000 MC sweeps

with ρ^*=.95, T^*=.7, N=4096

block size	number	f_{xyxy}	f_{xxyy}
2	250	−7.26±0.60	−7.1±1.6
4	125	−11.55±0.80	−10.2±1.9
5	100	−11.75±0.83	−10.3±2.0
10	50	−14.03±0.79	−14.2±2.4
20	25	−15.02±0.85	−15.4±2.1
25	20	−15.09±0.93	−15.2±1.9
50	10	−15.59±1.04	−16.6±2.2
100	5	−15.86±0.83	−17.6±2.4
125	4	−16.06±1.17	−18.4±3.4
250	2	−16.08±0.87	−18.2±2.4
500	1	−16.08	−18.1

values of the fluctuation terms. Other simulations of the solid phase at various densities gave similar results for the behavior of the fluctuation terms.

In addition to measuring the elastic constants in the solid phase, we also measured them at one density in the transition region and one density in the fluid region. In the transition region we measured the elastic constants at $\rho^* = .84$ since it was the only simulation in the transition region that exhibited several of its long-term fluctuations in the number of sweeps that we could perform. In contrast to the behavior of the fluctuation

Table 7.2

Fluctuation Terms from 50,000 MC sweeps with ρ^*=.84, T^*=.7, N=4096

block size	number	f_{xyxy}	f_{xxyy}
500	10	−26.99±0.86	−17.9±3.0
625	8	−27.21±0.65	−15.9±2.2
1000	5	−27.37±1.04	−20.8±2.2
1250	4	−27.31±0.65	−19.2±2.2
2500	2	−27.63±1.34	−23.1±4.2
5000	1	−27.65	−26.2

terms in the solid region, Table 7.2 shows that the 5,000 measurements that were obtained from 50,000 sweeps at $\rho^* = .84$ may not be enough to give an accurate value of f_{xxyy} . While f_{xyxy} nearly reached its limiting value with 1,000 measurements per block, f_{xxyy} exhibited a substantial change when the number of measurements per block increased from 2,500 to 5,000, so we have no indication that f_{xxyy} has reached its limiting value. In the simulation of the fluid phase at $\rho^* = .80$, the fluctuation terms seemed to reach their limiting values more slowly than in the simulations of the solid phase, but the 500 measurements still appeared to give accurate values.

In Table 7.3 we summarize our measurements of the elastic constants along the $T^* = .7$ isotherm in the system with $N = 4096$. In addition to the fluctuation terms, we give the Born term and the kinetic term. We corrected the Born term for the effects of the

Table 7.3

Elastic Constants for T^*=.7 and N=4096

ρ^*	f_{xyxy}	f_{xxyy}	Born	kinetic	C^T_{xyxy}	C^T_{xxyy}
0.95	−16±1	−18±2	53.107±.024	0.665	38±1	35±2
0.94	−15±1	−17±2	49.505±.032	0.658	35±1	33±2
0.93	−15±1	−18±2	46.092±.024	0.651	32±1	28±2
0.92	−16±1	−21±3	43.062±.039	0.644	28±1	22±3
0.91	−16±1	−19±2	40.167±.031	0.637	25±1	21±2
0.90	−16±1	−19±2	37.655±.038	0.630	22±1	19±2
0.89	−15±1	−19±2	35.184±.021	0.623	21±1	16±2
0.88	−15±1	−23±3	33.032±.037	0.616	19±1	10±3
0.84	−28±1	−26±2	29.576±.078	0.588	2±1	4±2
0.80	−21±1	−14±4	24.297±.057	0.560	4±1	10±4

truncated potential by taking advantage of the isotropy of the elastic properties in the hexagonal crystal and the fluid to obtain $\frac{x^2y^2}{r^4} = \frac{1}{8}$ by averaging over orientations. We then used the results of the uniform density approximation and the measurements of the residual corrections to calculate the corrections to the Born term. We cannot correct the fluctuation terms with any such static corrections since the corrections would simply cancel in the variance and correlation that comprise the terms. However, correcting the fluctuation terms for the effects of the truncated potential is completely unnecessary since their errors are so large. The last two columns of Table 7.3 report the values of the two independent elastic constants with their standard deviations, which arise entirely from the fluctuation terms.

Table 7.4

Lamé Coefficients and Isothermal Bulk Modulus for T^*=.7 and N=4096

ρ^*	p^*	C^T_{xyxy}	C^T_{xxyy}	μ^*	λ^*	B^*_T
0.95	6.5990±.0037	38±1	35±2	31±1	42±2	73±2
0.94	5.8960±.0020	35±1	33±2	29±1	39±2	68±2
0.93	5.2505±.0039	32±1	28±2	27±1	33±2	60±2
0.92	4.6725±.0032	28±1	22±3	23±1	27±3	50±3
0.91	4.1603±.0031	25±1	21±2	21±1	25±2	46±2
0.90	3.7110±.0051	22±1	19±2	18±1	23±2	41±2
0.89	3.3024±.0031	21±1	16±2	18±1	19±2	37±2
0.88	2.9396±.0022	19±1	10±3	16±1	13±3	29±3
0.84	2.712 ±.016	2±1	4±2	−1±1	7±2	6±2
0.80	2.0368±.0053	4±1	10±4	2±1	12±4	14±4

In Table 7.4 we report the results of using (7.38) and (7.39) to convert the elastic constants into the Lamé coefficients, which we give in reduced units. We obtained the pressures from Tables 4.2 and 4.4, except for the pressure at $\rho^* = .84$, which we gave in Section 6.1. In addition to the Lamé coefficients, we report the reduced isothermal bulk modulus, B^*_T, which is related to the Lamé coefficients. Separating the second-order contributions to the Helmholtz free energy in (7.31) into hydrostatic compression and pure shearing stress gives

$$\frac{1}{2}\lambda\eta_{ii}^2 + \mu\eta_{ij}^2 = \frac{1}{2}(\lambda + \frac{2}{d}\mu)\eta_{ii}^2 + \mu(\eta_{ij} - \frac{1}{d}\delta_{ij}\eta_{kk})^2 \tag{7.40}$$

where d is the dimensionality of the system. Since the bulk modulus is twice the coefficient of the hydrostatic compression, we obtain

$$B_T = \lambda + \mu \tag{7.41}$$

in the two-dimensional system. In addition, we see from (7.40) that the shear modulus, which is the coefficient of the pure shearing stress, is identical to the Lamé coefficient μ.

From Table 7.4 we see that both the shear modulus and the bulk modulus of the solid decrease with decreasing density. At $\rho^* = .84$ and $\rho^* = .80$ the shear modulus is close to zero, indicating that the system is no longer a solid. Although the bulk modulus in the solid and fluid regions near the phase transition is large, it is considerably smaller at $\rho^* = .84$, which is consistent with the sharpness of the melting transition that Figure 6.13 illustrates. Using an alternate definition of the bulk modulus, which we will present, we can show that the isothermal bulk modulus is exactly zero in the two-phase region of an infinite system, since the pressure of the system remains constant as its density changes. While the bulk modulus is not zero at $\rho^* = .84$ to within its errors, f_{xxyy} may not have reached its limiting value, which would result in a value for the bulk modulus that is too large. In addition, the true bulk modulus may not be exactly zero due to finite-size rounding of the melting transition.

The values of the elastic constants also suggest another interpretation of the long-term fluctuations in the potential energy and pressure that we discussed in Section 6.1. The long-term fluctuations, which are unique to the transition region, are the same fluctuations that make the fluctuation terms in the elastic constants, especially C_{xxyy} , become sufficiently large to nearly cancel the Born term. Thus, the long-term fluctuations indicate that the system's bulk modulus is very small in the transition region. Such an interpretation in no way contradicts the previous interpretation that the fluctuations are finite-size effects that involve changes in the interface separating the solid and fluid regions of a two-phase system. Taken together, the two interpretations indicate that the changes in the solid-fluid interface reflect the behavior of the pressure in the two-phase region and vice versa.

Instead of using the values of λ^* that appear in Table 7.4 to evaluate the Kosterlitz-Thouless K parameter, we use the values of μ^* in Table 7.4 and the values of B_T^* that we obtain by a different method which gives smaller errors. Besides using (7.41) to express the isothermal bulk modulus in terms of the Lamé coefficients, we can write B_T as

$$B_T \equiv -V(\frac{\partial p}{\partial V})_T \ . \tag{7.42}$$

Converting (7.42) to reduced units and writing it in terms of density rather than volume give

$$B_T^* = \rho^*(\frac{\partial p^*}{\partial \rho^*})_{T^*} \ . \tag{7.43}$$

Using the previously derived expressions that relate V to η_{ij} and p to τ_{ij}, we can easily verify that (7.43) and the definition of B_T in terms of the Lamé coefficients are equivalent.

We now obtain the bulk modulus by fitting a third-degree polynomial to the pressure data in Table 4.2 using the same algorithm as we used in Chapter 5. Differentiating the resulting polynomial gives the derivative in (7.43), which we evaluate at each of the simulation densities in the solid region. We propagate the statistical errors in the data using the method described in Section 5.3, which involves repeating the analysis many times with data sets generated from the original data. Fitting a polynomial to the measured pressures allows us to avoid measuring second derivatives of the free energy directly, as we did to calculate the elastic constants. Thus, we avoid the large statistical errors in the fluctuation terms and obtain values of the isothermal bulk modulus whose errors are about an order of magnitude smaller. The third column of Table 7.5 gives the results of calculating B_T^* by fitting a polynomial to the pressure data. The results of the two methods of obtaining B_T^* are generally consistent, and the large error estimates for the values in Table 7.4 appear to be reasonable.

Table 7.5

Kosterlitz-Thouless K Parameter for T^*=.7 and N=4096

ρ^*	μ^*	B_T^*	K	$\frac{K}{16\pi}$
0.95	31±1	71.00±.42	150±3	2.98±.06
0.94	29±1	63.46±.19	140±3	2.79±.06
0.93	27±1	56.48±.09	130±3	2.59±.06
0.92	23±1	50.05±.11	113±3	2.25±.06
0.91	21±1	44.16±.12	103±3	2.05±.06
0.90	18±1	38.79±.08	[illegible]±3	1.79±.06
0.89	18±1	33.94±.14	87±3	1.73±.06
0.88	16±1	29.58±.33	78±3	1.55±.06

Using the values of μ^* and B_T^* reported in Table 7.5, we evaluate (7.1) to obtain the Kosterlitz-Thouless K parameter, which we report in the fourth column of Table 7.5. In addition, we divide K by 16π, which is the value of K at which the system becomes unstable to the formation of free dislocations. Although K decreases with decreasing density, its value is still about 50% above the limit of stability at $\rho^* = .88$, which is very close to the density of the solid at the melting transition. Unless K drops very sharply between $\rho^* = .88$ and the density of the solid at the melting transition, it would remain significantly larger than 16π when the system melts. Thus, the values of K seem to indicate that the melting transition occurs by some other mechanism than the unbinding of dislocation pairs that Kosterlitz and Thouless proposed. Since the renormalization group analysis of Halperin and Nelson[7,8] is based on the mechanism of melting that Kosterlitz and Thouless proposed, their results do not seem to apply to melting in the Lennard-Jones system at $T^* = .7$. In contrast, none of the measurements presented in this chapter weaken our previous interpretation that the melting transition is a conventional first-order phase transition.

Chapter 8: Topological Defects and Angular Correlations

8.1 Topological Defects

In this chapter we attempt to test some of the predictions that the Halperin-Nelson theory of melting in two dimensions makes about the topological order of the system. We first examine the topological defects in configurations that simulations at various densities produced and compare the behavior of the defects to the predictions of the theory. In the following section we extract a measure of the angular correlation of the bonds that connect a particle to its nearest neighbors. We then compare the behavior of the angular correlation function to that predicted by the Halperin-Nelson theory.

At low temperatures the two-dimensional system contains only bound pairs of thermally excited dislocations. According to the Halperin-Nelson theory,[7,8] both the number of such pairs and the average length of their bonds increase with increasing temperature until the dislocations unbind at the temperature T_m, which is defined by $K(T_m) = 16\pi$. The resulting system cannot support a shearing stress since it contains free dislocations, but Halperin and Nelson show that disclinations may remain bound in pairs of opposite disclinicity, giving the system a sixfold anisotropy. According to their theory, a region of hexatic phase, which is orientationally ordered but translationally disordered, separates the normal solid and fluid phases. At a higher temperature, T_i, the pairs of disclinations unbind and the hexatic phase becomes an isotropic fluid.

In order to test the sequence of events that the Halperin-Nelson theory predicts for the melting transition, we examine the defect structure of configurations at densities in and around the transition region. Although we vary the density of the system while holding its temperature constant instead of varying its temperature, we still expect to see the predicted behavior of the defects if the Halperin-Nelson theory accurately describes the system. Before we can extract the defect structure of a configuration, we need a precise definition of a particle's nearest neighbors, which the Voronoi polygon construction[21] provides. The Voronoi polygon that surrounds a particle is the polygon that encloses all of the points closer to the particle than any other particle. The sides of the Voronoi polygon are the perpendicular bisectors of the lines that connect the particle to its nearest neighbors. Tobochnik and Chester[21] show that a particle with a coordination number of n, which is its number of nearest neighbors, is a disclination of strength $m = 6 - n$. They also note that in a triangular lattice an adjacent pair of disclinations of strengths $+1$ and -1 form a dislocation whose Burgers vector is perpendicular to the line connecting the two disclinations.

In Figure 8.1 we show the topological defects in a configuration at $T^* = .7$ and $\rho^* = .88$, where the system is a solid. The dots indicate the positions of the particles and the additional symbols identify the defects according to the key at the top of the figure. With the exception of one cluster of six disclinations, we see that all of the defects consist of clusters of four disclinations, which represent a tightly bound pair of dislocations whose Burgers vectors point in opposite directions. In the case of the cluster of six disclinations, the Burgers vectors of the three dislocations also sum to zero due to the symmetry of the lattice. We note that the thermally excited pairs of dislocations apparently show no tendency to cluster together and that their effects on the crystal lattice are local.

At $\rho^* = .87$, which is in the transition region, the number of defects is larger and a new type of defect appears. Figure 8.2 shows two instances of a disclination of strength -2 that is tightly bound to two others of strength $+1$. Closer inspection of the figure reveals that nearby vacancies in the lattice are responsible for both such defects. Although the system is at the measured transition pressure and below the estimated density of the solid at the phase transition, we see no evidence that free dislocations can exist in the system. In fact, only one pair of dislocations has separated by more than a single lattice spacing.

Figure 8.3 shows the defects in a configuration at $\rho^* = .86$, which contains substantially more defects than were present in the previous figures. While many of the isolated defects are bound pairs of dislocations, a few dislocations do not seem to be bound. However, a strong tendency for the defects to cluster is evident, making a determination of the interactions between the defects difficult. On the other hand, the disclinations appear to be tightly bound since we see no disclination that is more than two lattice spacings from a disclination of opposite sign. Thus, the predictions of the Halperin-Nelson theory seem to describe the defect structure in Figure 8.3. However, we could also explain the defects by considering the transition region to be a two-phase region in which channels of fluid separate domains of solid. Since we would expect lattice defects to be clustered around the domain boundaries in such a situation, we see that the defect structure in Figure 8.3 also agrees with the interpretation that the melting transition is first-order.

In Figures 8.4 and 8.5 we illustrate the defect structure of configurations at $\rho^* = .85$ and $\rho^* = .84$, respectively. As in Figure 8.3, we see that the defects exhibit a strong tendency to cluster. Since the concentration of defects continues to increase as the density of the system decreases, the distinction between bound dislocations and free ones gradually disappears. Still, we see no evidence of free disclinations existing in any of the configurations. Figure 8.6 shows the defect structure of a configuration at $\rho^* = .83$, where the system is an ordinary fluid. We see that the defect structure of the fluid appears to be similar to that of the system at $\rho^* = .84$, which is in the transition region, except that its concentration of defects is higher. We cannot determine whether any of the disclinations are free in the fluid since the large number of defects obscures their interactions.

Although the defect structures of the configurations in Figures 8.1 to 8.6 clearly exhibit some of the behaviors that the Halperin-Nelson theory predicts, we can also interpret them in terms of a first-order melting transition. Thus, we cannot reliably distinguish between the proposed hexatic phase and a two-phase region. We note only that none of the observed features of the lattice defects weaken our previous interpretation that the melting transition is first-order at $T^* = .7$.

8.2 Angular Correlation Function

Using the precise definition of a particle's nearest neighbors that the Voronoi polygon construction provides, we can easily calculate a measure of the angular correlation in the crystal lattice. Nelson and Halperin[8] define an order parameter for bond orientations in a triangular lattice as

$$\psi(r) = e^{i6\theta(r)} \tag{8.1}$$

where $\theta(r)$ is the orientation of the bond relative to some fixed axis. The factor of six in the definition of ψ arises from the symmetry of the triangular lattice. In terms of ψ the angular correlation function is

$$g_6(r) = \langle \psi^*(r)\psi(0) \rangle \tag{8.2}$$

where the asterisk denotes the complex conjugate.

Halperin and Nelson show that when melting occurs due to the unbinding of dislocation pairs, the resulting fluid retains an orientational order since the pairs cf disclinations remain bound. The angular correlation function of the anisotropic fluid decays algebraically rather than approaches a constant as it does in a solid or decays exponentially as it does in an isotropic fluid. Thus, they predict that g_6 will have the form

$$g_6(r) = a\, r^{-\eta_6(T)} \tag{8.3}$$

where η_6 is always less than $\frac{1}{4}$, except at the disclination unbinding temperature where $\eta_6(T_i) = \frac{1}{4}$. Tobochnik and Chester[21] note that η_6 cannot exceed $\frac{1}{4}$ as long as macroscopic elastic theory correctly describes the disclinations, since the system would then be unstable to the formation of free disclinations. The presence of free disclinations makes the fluid isotropic and causes the angular correlation function to decay exponentially instead of algebraically.

Rather than use the definition of ψ in (8.1), we have used a slightly modified definition, which considerably reduces the amount of computation required to obtain the angular correlation function. By averaging over the bonds that connect a particle to its n nearest neighbors, we obtain

$$\psi(r) = \frac{1}{n}\sum_{1}^{n} e^{i6\theta(r)} \tag{8.4}$$

which is similar to the definition of ψ that Frenkel and McTague[14] used. Actually, the only difference between (8.4) and (8.1) is that in (8.4) r refers to the location of the particle instead of the center of the bond as it does in (8.1).[21] Thus, we see oscillations in our results of the angular correlation function, which are caused by correlations in the particles' positions, but the oscillations do not obscure the form of the correlation function. After calculating ψ for each of the particles in the system, we evaluate $g_6(r)$ by averaging $\psi^*(r_i)\psi(r_j)$ over all distinct pairs of particles and using $r = |r_i - r_j|$.

In Figure 8.7 we plot the angular correlation function of a configuration at $T^* = .7$ and $\rho^* = .88$ versus r, using a logarithmic scale for both axes. The range of r over which we plot g_6 extends from $r = \sigma$ to $r = 33\sigma$, which is nearly half the minimum dimension of the periodic system. We see that g_6 quickly approaches the constant value of $g_6 = .624$, which is consistent with the system's being a solid at $\rho^* = .88$. Figure 8.8 is a similar plot of g_6 at $\rho = .87$ in which we averaged the angular correlation function over four configurations. The correlation function at $\rho^* = .87$ approaches the constant value of $g_6 = .556$. Since we did not calculate g_6 for many configurations, we cannot accurately estimate the statistical error of the asymptotic value, but it appears to be about ±.002. Table 8.1 reports the asymptotic values of g_6 for several densities at $T^* = .7$.

Table 8.1

Asymptotic Values of g_6 at $T^* = .7$	
ρ^*	g_6
0.90	0.675
0.89	0.638
0.88	0.624
0.87	0.556

Starting with the density $\rho^* = .86$, we were able to fit the angular correlation function to a function of the form given in (8.3) with a value of η_6 that was nonzero. In Figures 8.9 to 8.11 we plot g_6 versus r for the densities $\rho^* = .86$, $\rho^* = .85$, and $\rho^* = .84$, respectively, using a logarithmic scale for both axes. In addition, we plot the function of the form given in (8.3) that best fits the data. While the angular correlation functions in Figures 8.9 and 8.10 seem to fit the predicted function quite well, the fit is somewhat poorer in Figure 8.11. We believe that the upward deviation from the straight line for large r is due to the periodic boundary conditions of the system, which allow the effect of a particle to wrap around the system in both directions. At each of the densities we averaged g_6 over four configurations to obtain the function that we plotted. We observed large fluctuations in the angular correlation functions of the configurations, which make the errors at the densities in the transition region considerably larger than those at higher densities. Due to the large number of Monte Carlo sweeps associated with the long-term fluctuations, we could not accurately determine the statistical errors of the correlation functions. In Table 8.2 we report the values of a and η_6 for each of the densities at which g_6 exhibited a nonzero slope.

Table 8.2

Values of a and η_6 at T^*=.7

ρ^*	a	η_6
0.86	0.461	0.037
0.85	0.516	0.315
0.84	0.727	0.831

At densities of $\rho^* = .83$ and lower, the angular correlation function appeared to decay exponentially rather than algebraically. However, accurate measurement of the correlation length was impossible due to large fluctuations in g_6 as it approached zero.

In our measurements of the angular correlation function we saw each of the behaviors that Halperin and Nelson predicted. In the solid regime and at the highest density in the transition region, g_6 clearly approached a constant value. At the densities in the transition region other than $\rho = .87$, g_6 fit the functional form in (8.3) quite well over a substantial range of r. We also observed exponential decay of the correlation function at densities that lie in the fluid regime. While our data agree with some of their predictions, the values of η_6 at $\rho^* = .85$ and $\rho^* = .84$ are considerably above the maximum value of $\frac{1}{4}$

that they predict. Although at $\rho^* = .85$ the value of η_6 is close enough to $\frac{1}{4}$ so that we could dismiss the difference as error due to the long-term fluctuations in g_6, the same is not true at $\rho^* = .84$. Thus, the behavior of the angular correlation function in the transition region argues against interpreting the transition region as a region of hexatic phase.

Tobochnik and Chester[21] note that the angular correlation function might exhibit an algebraic decay in a two-phase configuration, arising from the competing effects of its solid and fluid components. While g_6 would approach a constant in the domains of solid, it would decay exponentially to zero in the regions of fluid. Averaging the two behaviors over the whole system might produce an algebraic decay in g_6 or at least a behavior that is sufficiently similar to an algebraic decay that we could not distinguish between the two in our measurements. Thus, the measurements of the angular correlation function support interpreting the transition region as a two-phase region rather than as a region of hexatic phase.

Lattice Defects at $\rho^* = .88$

symbol	n	#	%
⊙	5	25	.6
·	6	4046	98.8
◬	7	25	.6

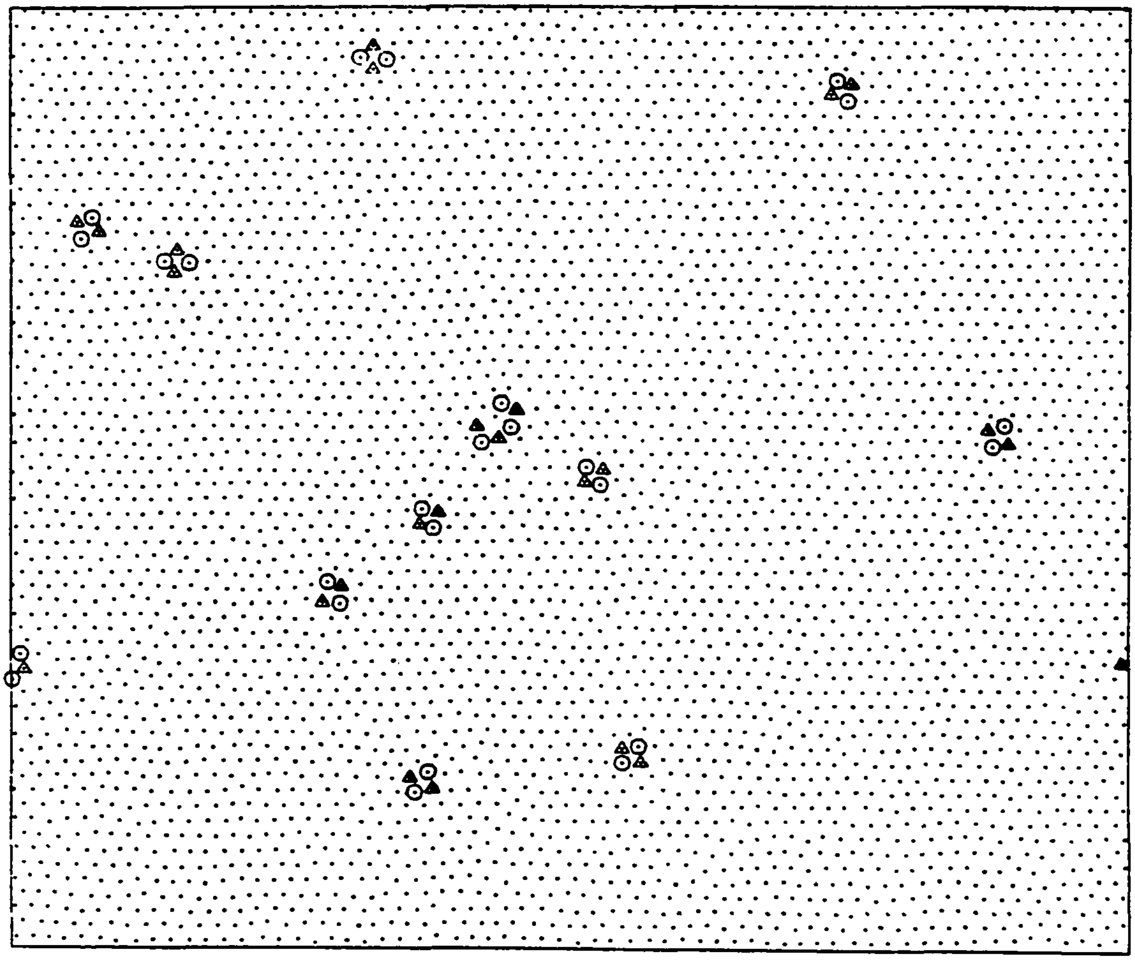


Figure 8.1 The topological defects in a configuration at $T^* = .7$ and $\rho^* = .88$. The dots indicate the positions of the particles and the additional symbols identify the defects according to the key at the top of the figure. In the key n is the coordination number and # and % are the number and percentage of such particles in the configuration.

Lattice Defects at $\rho^* = .87$

symbol	n	#	%
⊙	5	49	1.2
·	6	4000	97.7
▲	7	45	1.1
⊡	8	2	.0

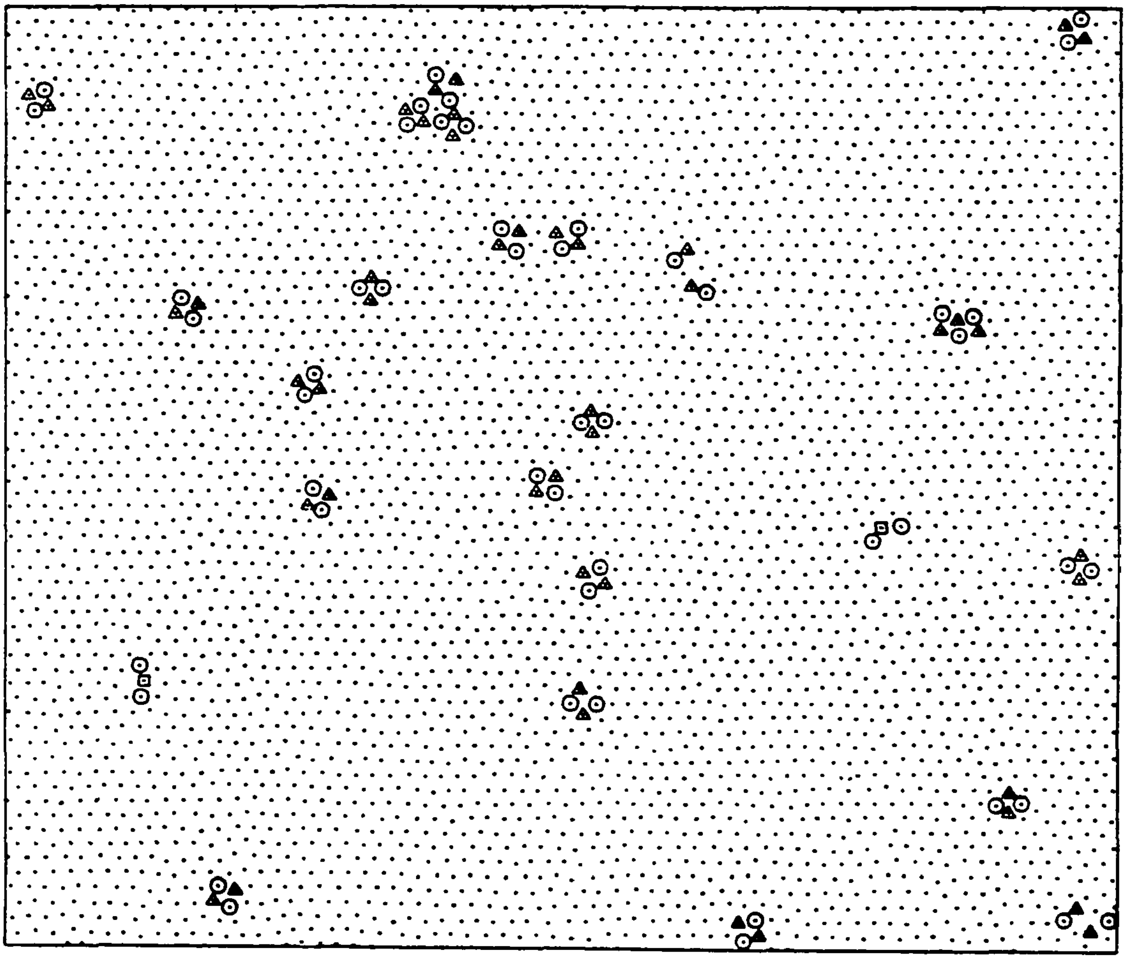


Figure 8.2 The topological defects in a configuration at $T^* = .7$ and $\rho^* = .87$. The dots indicate the positions of the particles and the additional symbols identify the defects according to the key at the top of the figure. In the key n is the coordination number and # and % are the number and percentage of such particles in the configuration.

Lattice Defects at ρ^* =.86

symbol	n	#	%
⊙	5	187	4.6
·	6	3732	91.1
▲	7	168	4.1
⊡	8	9	.2

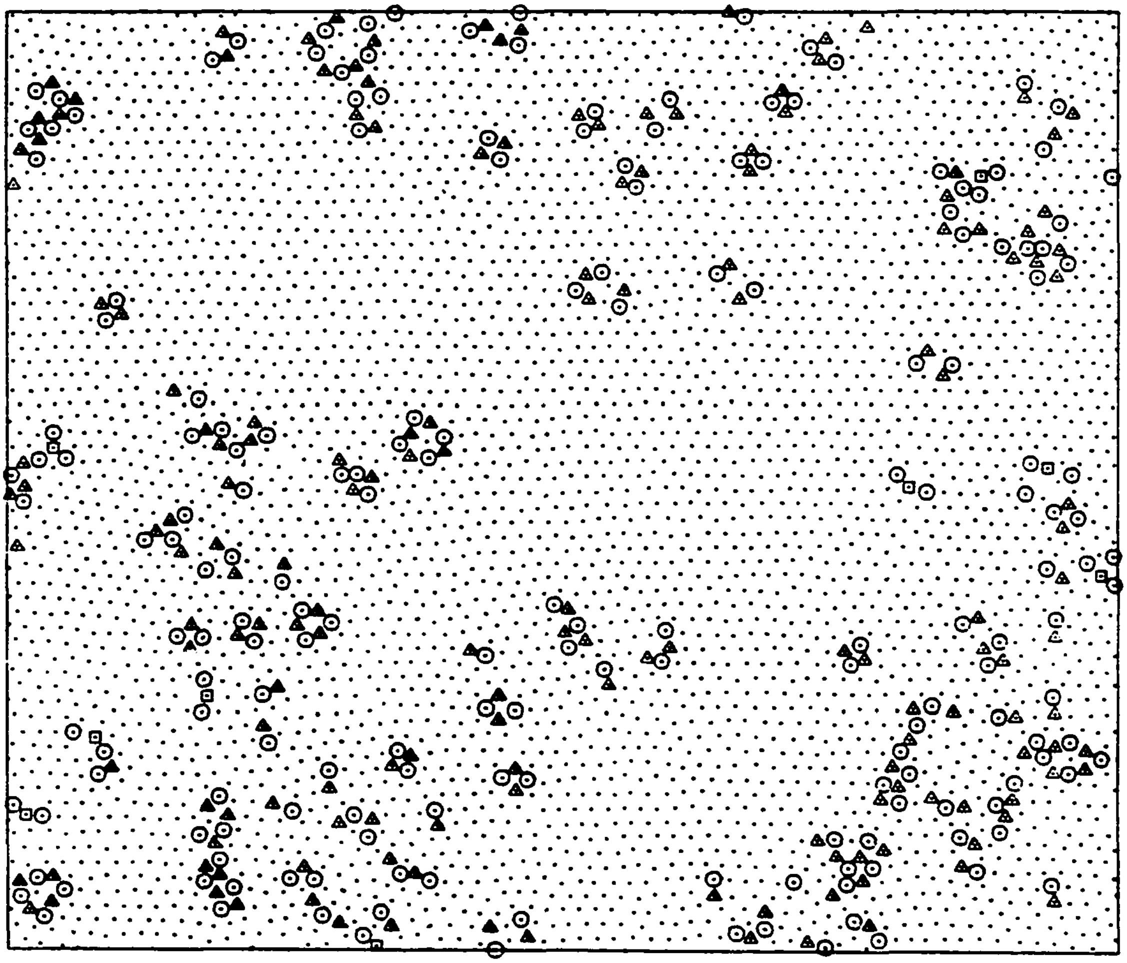


Figure 8.3 The topological defects in a configuration at $T^*=.7$ and $\rho^*=.86$. The dots indicate the positions of the particles and the additional symbols identify the defects according to the key at the top of the figure. In the key n is the coordination number and # and % are the number and percentage of such particles in the configuration.

Lattice Defects at $\rho^* = .85$

symbol	n	#	%
⊙	5	280	6.8
·	6	3543	86.5
△	7	266	6.5
▣	8	7	.2

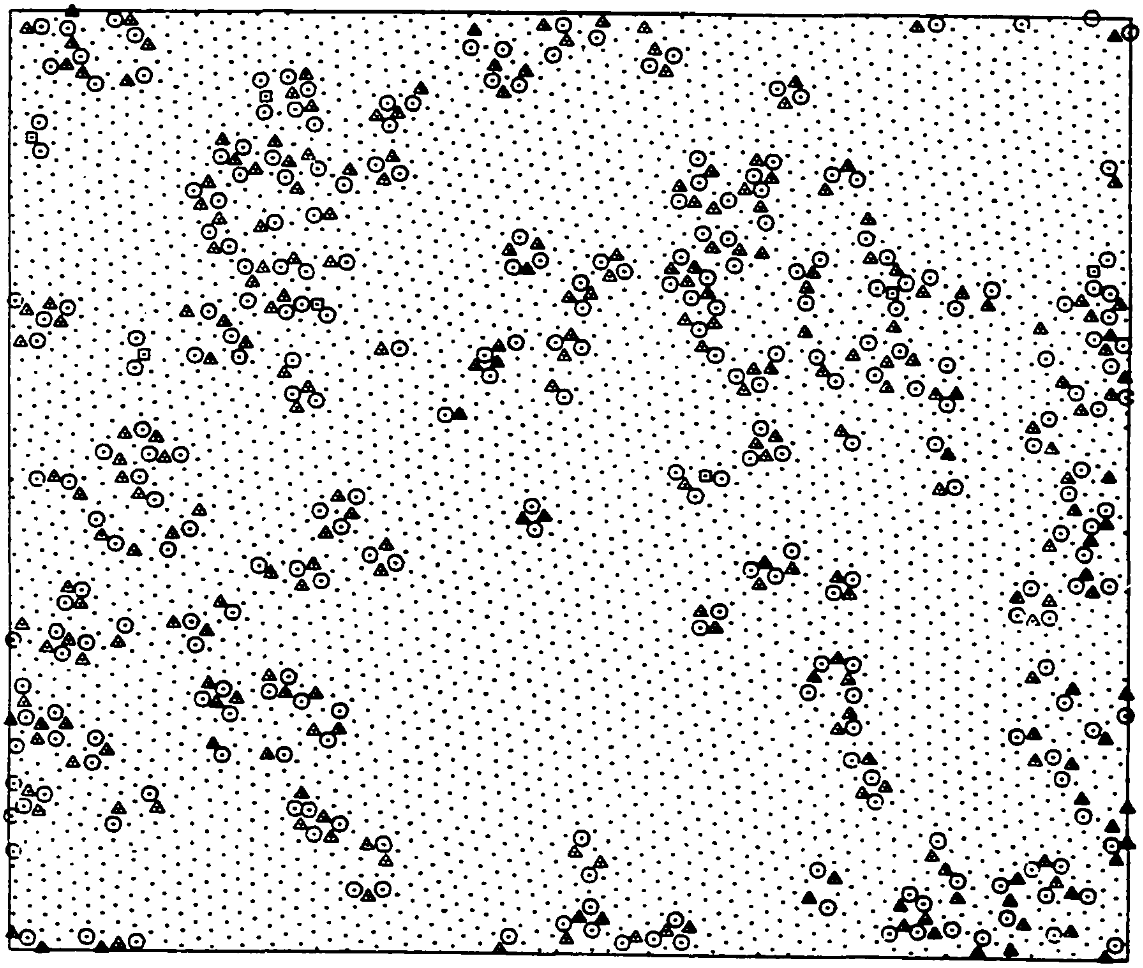


Figure 8.4 The topological defects in a configuration at $T^* = .7$ and $\rho^* = .85$. The dots indicate the positions of the particles and the additional symbols identify the defects according to the key at the top of the figure. In the key n is the coordination number and # and % are the number and percentage of such particles in the configuration.

Lattice Defects at $\rho^* = .84$

symbol	n	#	%
◈	4	2	.0
⊙	5	426	10.4
·	6	3249	79.3
▲	7	408	10.0
⊡	8	11	.3

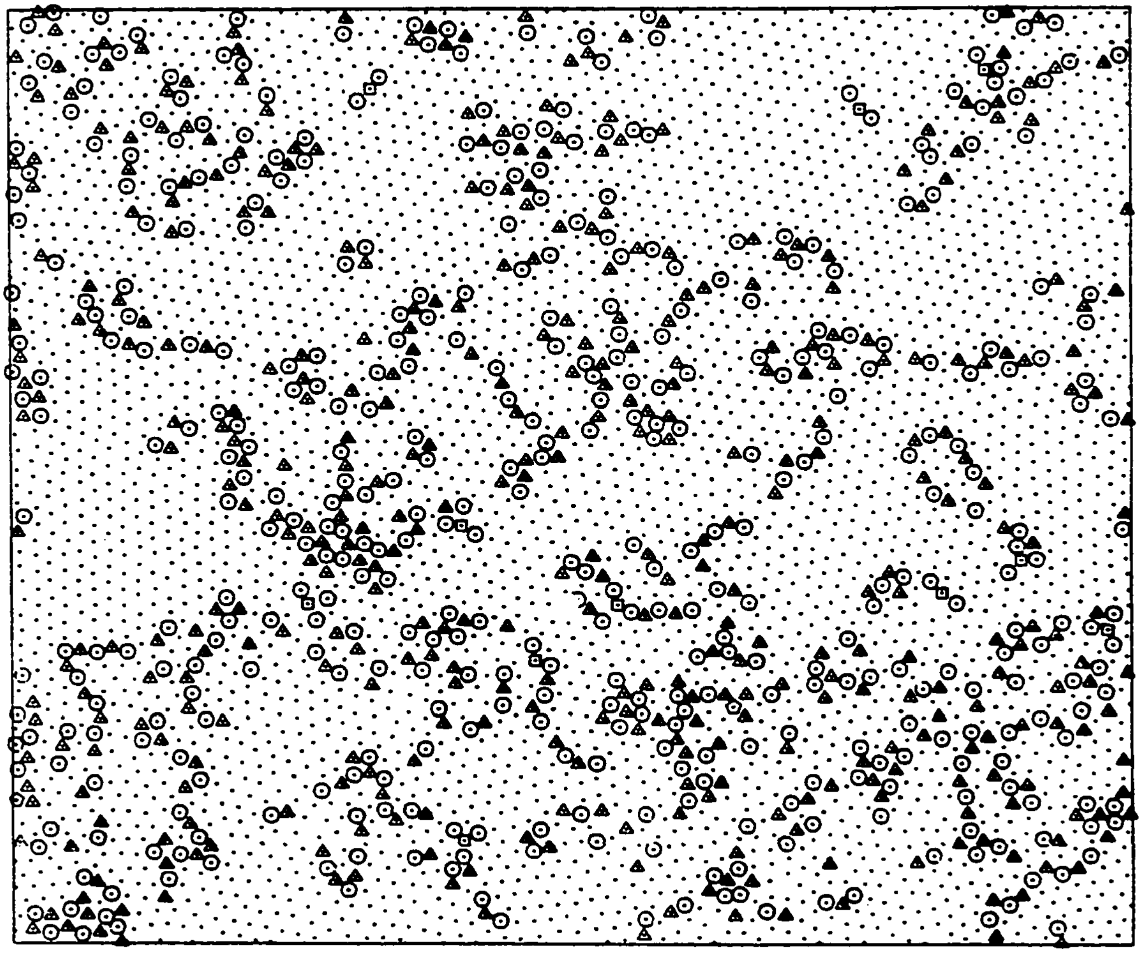
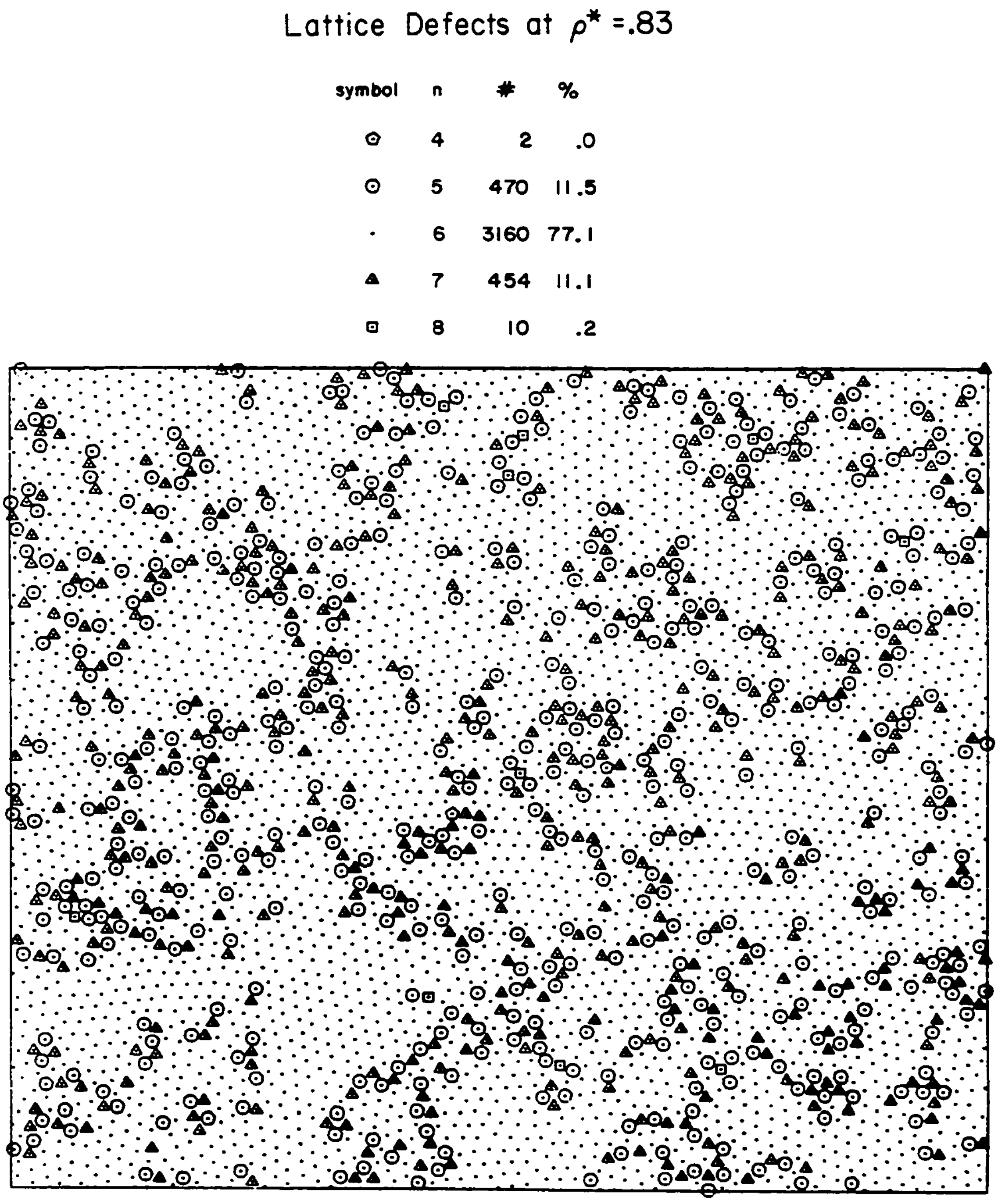


Figure 8.5 The topological defects in a configuration at $T^* = .7$ and $\rho^* = .84$. The dots indicate the positions of the particles and the additional symbols identify the defects according to the key at the top of the figure. In the key n is the coordination number and # and % are the number and percentage of such particles in the configuration.



symbol	n	#	%
⬡	4	2	.0
⊙	5	470	11.5
·	6	3160	77.1
▲	7	454	11.1
▣	8	10	.2

Figure 8.6 The topological defects in a configuration at $T^* = .7$ and $\rho^* = .83$. The dots indicate the positions of the particles and the additional symbols identify the defects according to the key at the top of the figure. In the key n is the coordination number and # and % are the number and percentage of such particles in the configuration.

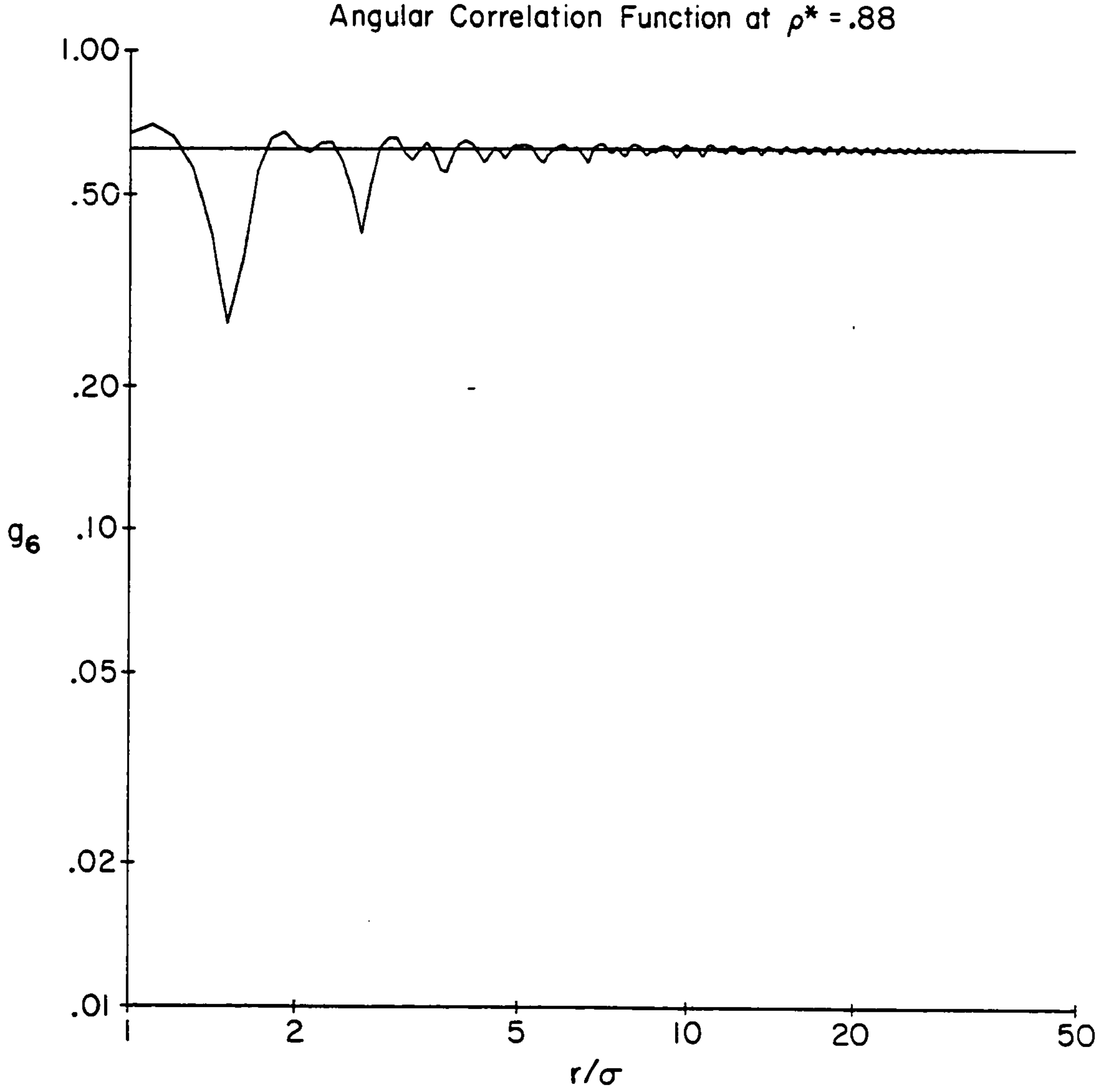


Figure 8.7 The angular correlation function of a configuration at $T^*=.7$ and $\rho^*=.88$. A logarithmic scale is used for both axes. The horizontal line indicates the asymptotic value of g_6=.624.

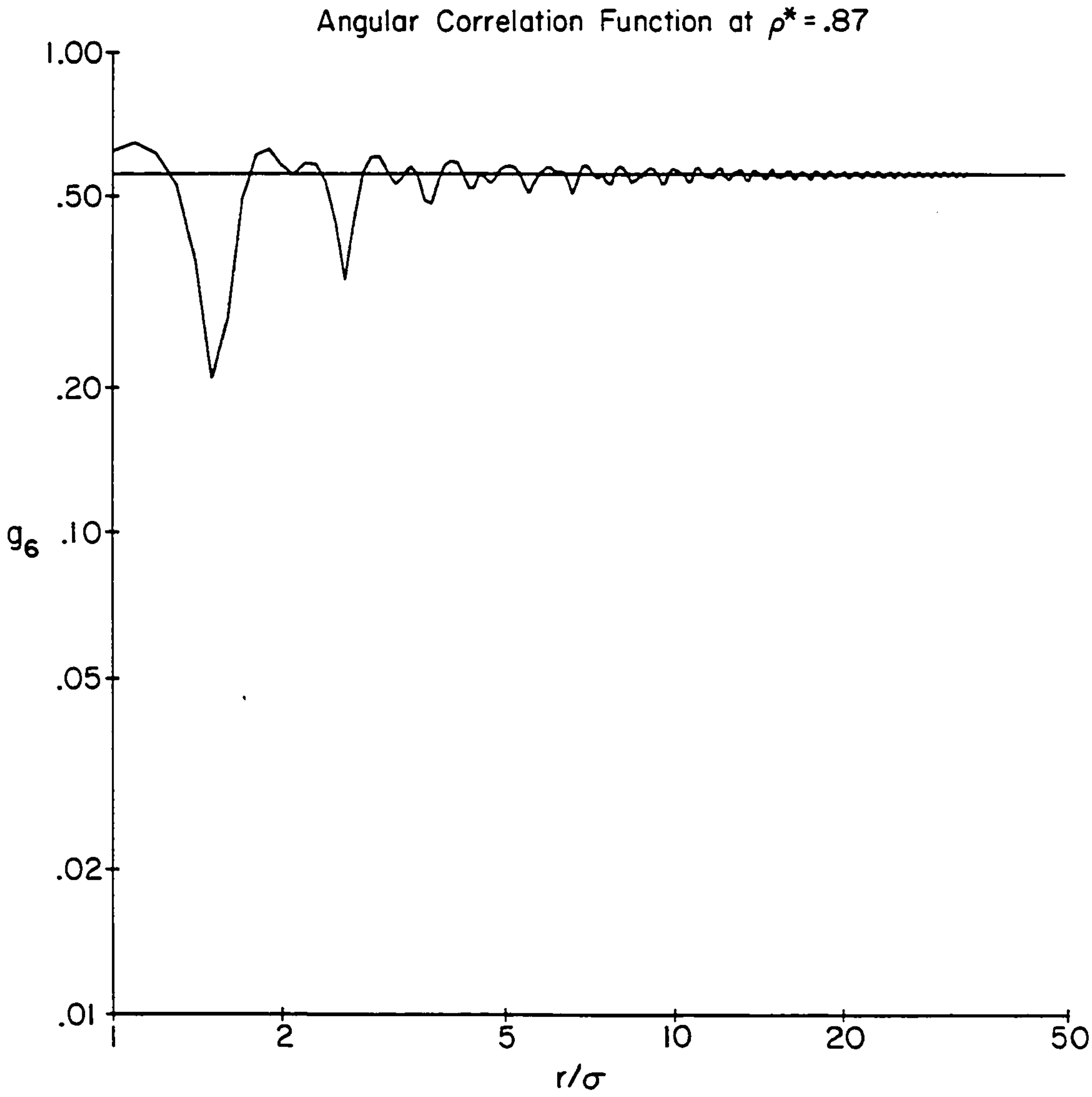


Figure 8.8 **The angular correlation function averaged over four configurations at $T^* = .7$ and $\rho^* = .87$. A logarithmic scale is used for both axes. The horizontal line indicates the asymptotic value of $g_6 = .556$.**

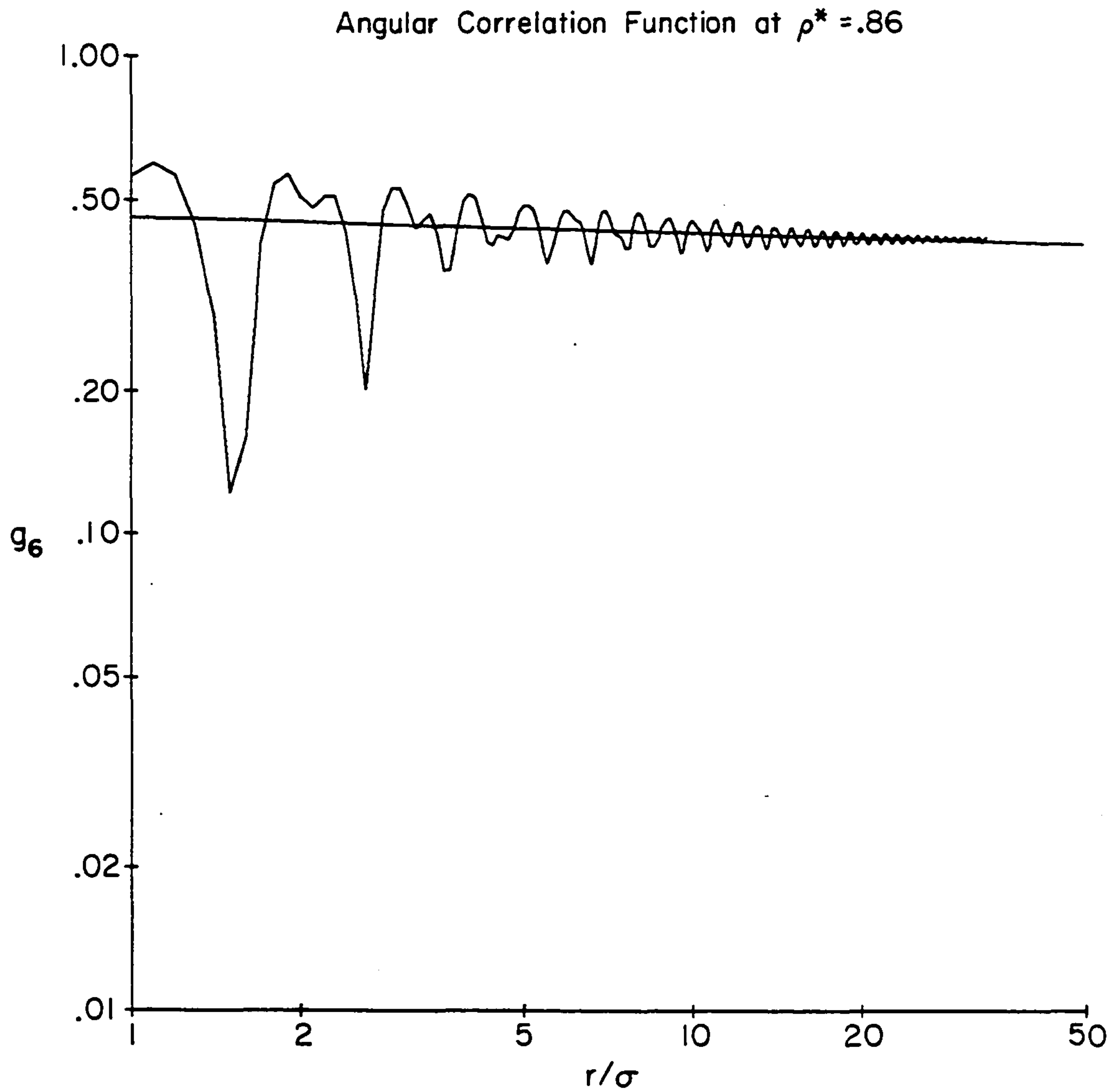


Figure 8.9 The angular correlation function averaged over four configurations at $T^* = .7$ and $\rho^* = .86$. A logarithmic scale is used for both axes. We also plot $g_6(r) = a\, r^{-\eta_6(T)}$ for $a = 0.461$ and $\eta_6 = 0.037$, which was fit to the correlation function.

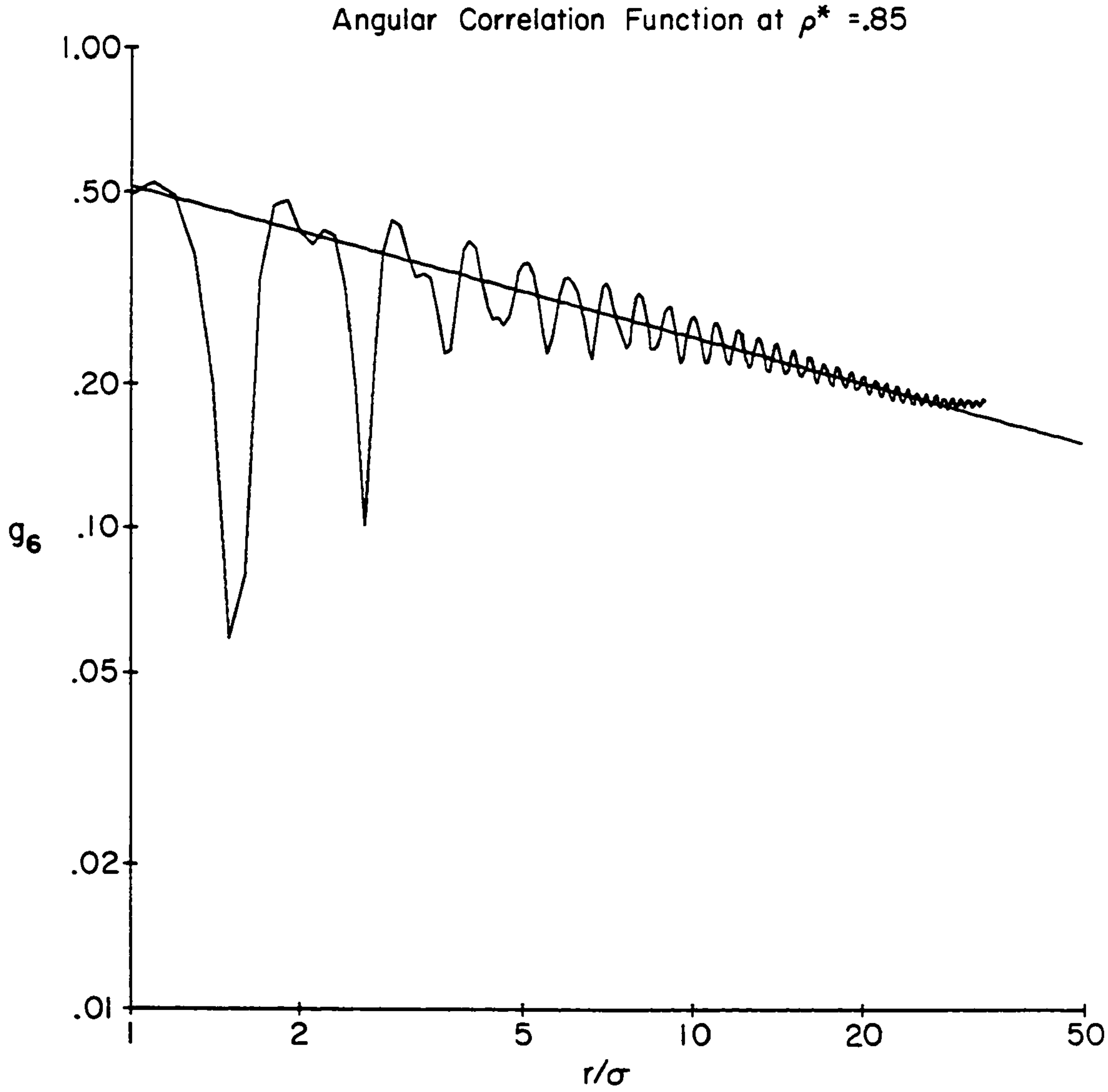


Figure 8.10 The angular correlation function averaged over four configurations at $T^* = .7$ and $\rho^* = .85$. A logarithmic scale is used for both axes. We also plot $g_6(r) = a\,r^{-\eta_6(T)}$ for $a = 0.516$ and $\eta_6 = 0.315$, which was fit to the correlation function.

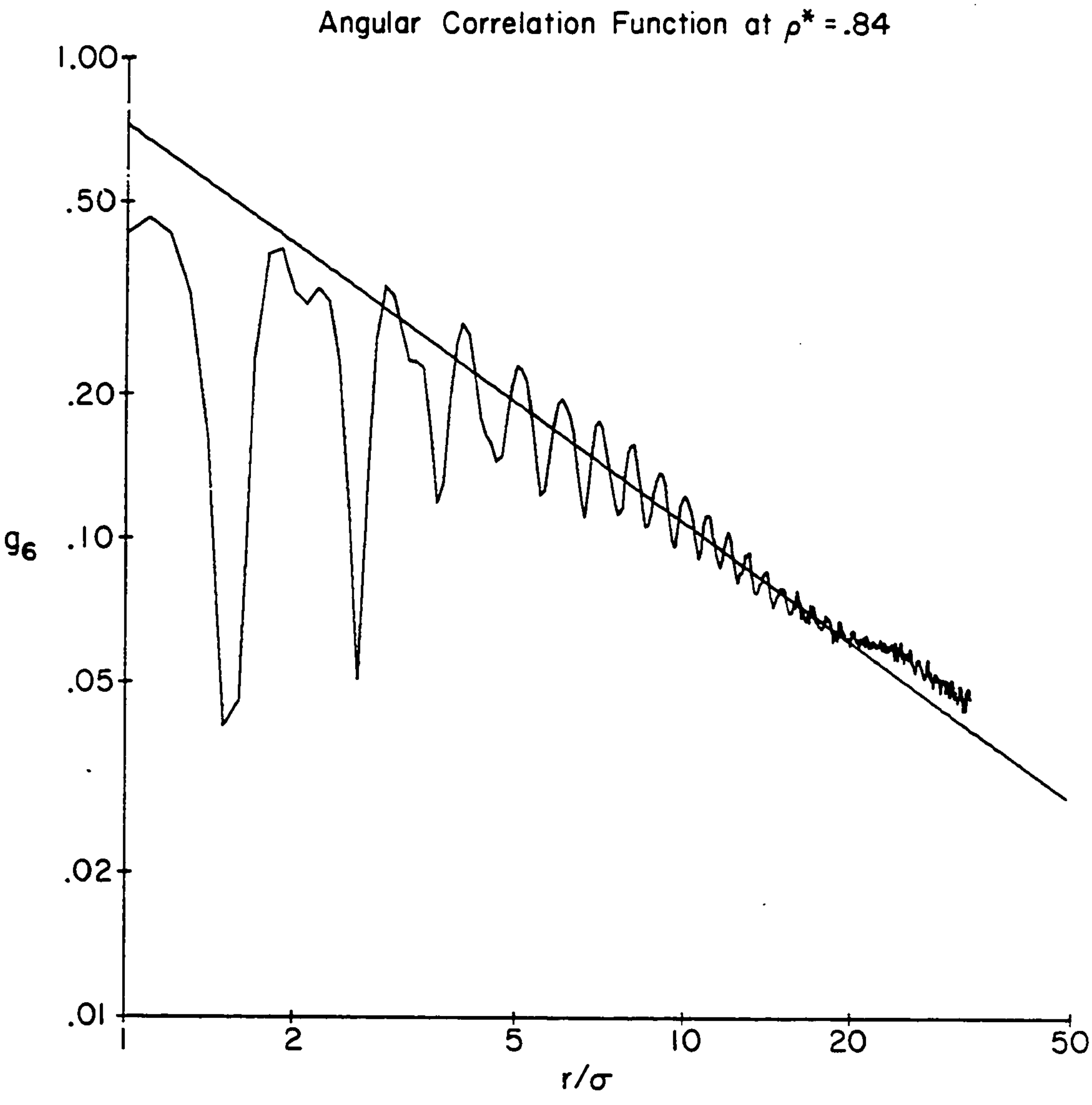


Figure 8.11 The angular correlation function averaged over four configurations at $T^* = .7$ and $\rho^* = .84$. A logarithmic scale is used for both axes. We also plot $g_6(r) = a\, r^{-\eta_6(T)}$ for $a = 0.727$ and $\eta_6 = 0.831$, which was fit to the correlation function.

Chapter 9: Conclusions

After presenting a detailed description of the Monte Carlo method in Chapter 2, we discussed the implementation of the simulation on a concurrent processor and analyzed its performance in Chapter 3. The simulation of the Lennard-Jones system was the first application that we implemented on the Caltech concurrent processor, which required a more general communication environment than the Crystalline Operating System provided. The irregular nature of the algorithm required that we implement an interrupt-driven communication system, which allowed the nodes to run asynchronously during the simulation. In order to maintain the necessary sequential ordering of the updates, we developed a technique that guarantees a consistent ordering of conflicting updates without drastically decreasing the efficiency of the update algorithm. Although we designed the implementation for a concurrent processor, the implementation would also perform very well on a sequential computer since we used geometric hashing to reduce the complexity of the algorithm to order N. Thus, we could easily use our simulation program to investigate larger systems when faster computers become available.

In Chapter 4 we reported measurements of the potential energy and pressure of the Lennard-Jones system along integration paths in the solid and fluid regions. While our results are not quite the same as those of other investigators, such as Barker et al.,[18] finite-size effects might be significantly affecting their systems, most of which are substantially smaller than ours. In Chapter 5 we used the measurements of the potential energy and pressure to calculate the free energy in the solid and fluid regions, allowing us to predict the location of the melting transition. The results of our free energy analysis, which predicted a melting pressure of $p_m^* = 2.724 \pm 0.034$, differ from the results of Barker et al., who reported a predicted pressure of $p_m^* = 2.85$ at the same temperature of $T^* = .7$.

In Chapter 6 we presented the results of directly simulating the system in the transition region. The measured values of the pressure in the constant-density simulations were near the value predicted by our free energy analysis and were consistent with the pressure's being constant across the transition region. We interpreted the transition region as a two-phase region, which identifies the melting transition as being first-order. In addition, we demonstrated that finite-size effects are significant in the transition region in the 1024-particle system. Thus, we believe that most of the previous simulations of the transition region were completely dominated by finite-size effects. We also observed long-term fluctuations of the pressure that are unique to the transition region, which make obtaining accurate expectation values in the transition region difficult.

In addition, we also presented the results of a series of constant-pressure simulations of the transition region. Since melting a solid without surfaces is strongly metastable in a constant-pressure simulation, we initialized the simulations with a configuration from a constant-density simulation of the transition region. We showed that such constant-pressure simulations can provide bounds on the width in pressure of the transition region and that the Lennard-Jones system has a very sharp melting transition. We believe that the constant-pressure simulations provided the strongest evidence that the melting transition is a first-order transition. In addition, we noted that the results of the free energy analysis, the constant-density simulations, and the constant-pressure simuiations are all consistent.

In Chapter 7 we presented measurements of the elastic constants of the system that allowed us to test the predictions of the Kosterlitz and Thouless. Although our statistical errors were rather large, the results indicated that the system melted before reaching the Kosterlitz-Thouless instability temperature. Thus, we do not believe that the two-dimensional Lennard-Jones system at $T^* = .7$ melts through the formation of free dislocations.

We directly examined the topological characteristics of the system in Chapter 8 by illustrating the topological defects and by plotting the angular correlation function. At high densities along the $T^* = .7$ isotherm we observed bound pairs of dislocations that had no noticeable tendency to cluster. In the transition region the number of defects was large enough to obscure their interaction, but we did notice a definite tendency of the defects to cluster. Since we could interpret the characteristics of the defects in terms of either a hexatic phase or a two-phase region, we could not distinguish reliably between the two by examining the lattice defects.

The angular correlation function clearly approached a constant at densities where the system was a solid and at the highest density in the transition region. At densities where the system was a fluid, the angular correlation function seemed to decay exponentially, but long-term fluctuations prevented an accurate measurement of its correlation length. The angular correlation function appeared to decay algebraically in the transition region, as predicted,by Halperin and Nelson. However, at the two lowest densities in the transition region it decayed faster than the results of the Halperin-Nelson theory allow. Thus, the angular correlation function provides evidence that the Halperin-Nelson theory does not correctly describe melting in the system.

The finite-size effects and long-term fluctuations that we observed in the transition region indicate that future simulations of the transition region should perform tens of thousands of Monte Carlo sweeps on systems containing at least 4096 particles. Although we do not believe that the Lennard-Jones system at $T^* = .7$ melts through the mechanism that Kosterlitz and Thouless proposed, the order of the transition may be different at other temperatures. Investigating the change in entropy across the transition at different temperatures would provide an indication of whether the transition becomes continuous at some temperature.

Potentials other than the Lennard-Jones potential could also be studied in an attempt to determine whether a two-dimensional system of particles ever melts through the Kosterlitz-Thouless mechanism. An interesting system for further study consists of particles interacting with a repulsive $\frac{1}{r}$ potential, such as that investigated by Morf.[52] The techniques of constant-pressure simulation that we used in Chapter 7 provide a way of placing bounds on the width of the transition region, which is our most sensitive indicator of the nature of the transition. In addition, our method of using constant-pressure simulations appears to be sufficiently sensitive that direct measurements of the finite-size rounding of first-order transitions could be made. Such measurements could be compared to the behavior of the finite-size rounding that Privman and Fisher predict.[53] As more powerful computers become available, simulations could reflect actual experimental conditions more accurately. In particular, the simulation could account for the interaction of the substrate with the atoms adsorbed onto its surface. Effects of multiple layers and the transition to a full three-dimensional system could then be investigated.

Our investigations in concurrent computing were some of the first that used an actual machine. Having several years of experience programming our concurrent processor, we believe that we understand the major issues involved in the Crystalline Operating System that we briefly described in Section 3.4. Since CrOS is now relatively stable, we present a detailed discussion of its current version in a book[54] that we are writing about concurrent computing. While we have designed and implemented a high-performance interrupt-driven communication system for the Monte Carlo simulation program,[28] many important issues, such as the user interface, still need to be resolved. At a deeper level, many questions about the correct programming model for concurrent computing remain.

Appendix A: Calculating Pressure from the Partition Function

As we mentioned in Section 4.1, we will calculate the pressure of a two-dimensional system of N interacting particles by using the partition function that describes the system in a constant volume V. In Section 2.2 we gave the partition function of the system at temperature T as

$$Z = \frac{1}{h^{2N} N!} \int_{-\infty}^{\infty} d^2p_1 \cdots d^2p_N \int_V d^2q_1 \cdots d^2q_N \, e^{-\beta E} \tag{2.2}$$

in which $\beta = (kT)^{-1}$, and E is the total energy of the system. In (2.2) p_i and q_i are the momentum and position of particle i, respectively. Splitting total energy into its kinetic and potential components gives

$$E = \sum_{i=1}^{N} \frac{p_i^2}{2m} + U(q_1 \cdots q_N) \ . \tag{A.1}$$

In Section 5.1 we derived an expression for the pressure of the system in terms of its Helmholtz free energy, F, which is

$$\left(\frac{\partial F}{\partial V} \right)_T = -p \ . \tag{5.8}$$

Using $F = -kT \ln Z$ from (5.18) in Section 5.2, we write the pressure in terms of Z as

$$p = kT \frac{1}{Z} \left(\frac{\partial Z}{\partial V} \right)_T \ . \tag{A.2}$$

Since limits of the integrals over the particles' coordinates in (2.2) depend on the volume of the system, we must change variables to make the volume appear explicitly before differentiating with respect to V. Dividing each coordinate by the length of the system in its direction gives $d^2q'_i = \frac{d^2q_i}{V}$, which we substitute into (2.2) to obtain

$$Z = \frac{V^N}{h^{2N}N!} \int_{-\infty}^{\infty} d^2p_1 \cdots d^2p_N \int_0^1 d^2q'_1 \cdots d^2q'_N \, e^{-\beta E} \ . \tag{A.3}$$

Substituting (A.3) into (A.2) and differentiating with respect to V gives

$$p = \frac{kTN}{V} - \langle \frac{\partial U}{\partial V} \rangle \tag{A.4}$$

where $\langle A \rangle$ is the expectation value of A at the temperature T and volume V, as defined in Section 2.2.

Using the general result in (A.4), we now consider the case of the Lennard-Jones system, following the same procedure as in Section 4.1. Converting (A.4) to reduced units gives

$$p = \frac{T^*}{V^*} - \langle \frac{\partial U^*}{\partial V^*} \rangle \ . \tag{A.5}$$

In order to differentiate with respect to the Lennard-Jones length parameter instead of V, we use $V^* = \frac{V}{N\sigma^2}$ to obtain

$$\frac{\partial V^*}{\partial \sigma} = -2\frac{V^*}{\sigma} \ . \tag{A.6}$$

Substituting (A.6) into (A.5) with the chain rule and using $\rho^* = \frac{1}{V^*}$ give

$$p = \rho^* T^* + \tfrac{1}{2}\rho^* \langle \sigma \frac{\partial U^*}{\partial \sigma} \rangle \ . \tag{A.7}$$

Since the expectation value of $\frac{\partial U^*}{\partial \sigma}$ is its average over many configurations in a Monte Carlo simulation, we can obtain the pressure of each configuration by evaluating (A.7) for each configuration. In Section 4.1 we gave the potential energy of the Lennard-Jones system in reduced units as

$$U^* = 4[\sigma^{12}\langle r^{-12}\rangle - \sigma^6\langle r^{-6}\rangle] \tag{4.1}$$

where (2.14) expresses $\langle r^{-12}\rangle$ and $\langle r^{-6}\rangle$ as a sum of the indicated quantity over all pairs of particles, divided by N. By substituting (4.1) into (A.7) and differentiating U^* with respect to σ, we obtain an expression for the pressure of the Lennard-Jones system, which is

$$p^* = \rho^* T^* + 12\rho^* [2\sigma^{12}\langle r^{-12}\rangle - \sigma^6\langle r^{-6}\rangle] \ . \tag{A.8}$$

We note that (A.8) is identical to (4.6), which was obtained by a different argument.

Appendix B: Calculating Pressure from the Virial Theorem

As we mentioned in Section 4.1, we will calculate the pressure of a two-dimensional system of N interacting particles by using the virial theorem. The virial theorem is[55]

$$2\overline{K} = -\overline{\sum_{i=1}^{N} \vec{r}_i \cdot \vec{F}'_i} \tag{B.1}$$

where K is the kinetic energy of the system and $\vec{F}'_i$ is the total force acting on particle i, which is located at $\vec{r}_i$. Since the virial theorem is statistical in nature, it uses the time averages of the relevant quantities, where the average of a quantity A over the interval τ is

$$\overline{A} = \frac{1}{\tau}\int_0^\tau A\,dt \quad . \tag{B.2}$$

The virial theorem applies to any system of particles whose coordinates and momenta remain finite, such as the Lennard-Jones system.

In order to derive the pressure of the system using the virial theorem, we begin by separating the total force acting on the particles into internal forces and forces of constraint. In the Lennard-Jones system the forces of constraint, which maintain the system at its specified density, consist of a hydrostatic compression that can be described as a uniform pressure over the unit cell of the periodic system. Writing the forces of constraint as an integral over S, the surface of the unit cell, in terms of the pressure p gives

$$\sum_{i=1}^{N} \vec{r}_i \cdot \vec{F}'_i = \sum_{i=1}^{N} \vec{r}_i \cdot \vec{F}_i - \oint_S p(\vec{r}\cdot\hat{n})\,dA \tag{B.3}$$

where $\hat{n}$ is the unit vector that is normal to the surface. The internal force on particle i is $\vec{F}_i$, which is the force due to all of the other particles but not the pressure. We use

$\int\limits_V \vec{\nabla}\cdot\vec{r}\,dV = \oint\limits_S (\vec{r}\cdot\hat{n})\,dA$ to convert from the integral over a closed surface in (B.3) to an integral over the volume of the system. In addition, we use $\vec{\nabla}\cdot\vec{r} = d$, where d is the dimensionality of the system, to obtain

$$\sum_{i=1}^{N} \vec{r}_i \cdot \vec{F}'_i = \sum_{i=1}^{N} \vec{r}_i \cdot \vec{F}_i - dpV \ . \tag{B.4}$$

The equipartition theorem gives the average kinetic energy of a d-dimensional system that contains N particles, which has dN kinetic degrees of freedom, as

$$\overline{K} = \tfrac{1}{2} dNkT \ . \tag{B.5}$$

We obtain an expression for the average pressure of the system by substituting (B.4) and (B.5) into (B.1), which gives

$$p = \frac{1}{dV}\overline{\sum_{i=1}^{N} \vec{r}_i \cdot \vec{F}_i} + \frac{NkT}{V} \ . \tag{B.6}$$

Keeping in mind that the thermodynamic pressure of the system is the time average of the pressure, we will drop the time averages in the remaining discussion and refer to the pressure of an individual configuration. In order to perform the sum in (B.6), we write $\vec{F}_i$ as the sum of the forces due to each of the other particles, which gives

$$\sum_{i=1}^{N} \vec{r}_i \cdot \vec{F}_i = \sum_{i=1}^{N} \sum_{j\neq i} \vec{r}_i \cdot \vec{f}_{ij} \tag{B.7}$$

where the force on particle i due to particle j is $\vec{f}_{ij}$. We obtain $\vec{f}_{ij} = -\vec{f}_{ji}$ from Newton's Third Law of Motion and substitute it into (B.7), giving

$$\sum_{i=1}^{N} \vec{r}_i \cdot \vec{F}_i = \sum_{\langle ij \rangle} (\vec{r}_i - \vec{r}_j)\cdot \vec{f}_{ij} \tag{B.8}$$

where $\langle ij \rangle$ indicates all distinct pairs of particles.

We now assume that the forces between particles arise from an interaction potential ϕ so that $|\vec{f}_{ij}| = -\frac{\partial \phi}{\partial r_{ij}}$, in which $r_{ij} = |\vec{r}_i - \vec{r}_j|$. In terms of the potential (B.8) becomes

$$\sum_{i=1}^{N} \vec{r}_i \cdot \vec{F}_i = -\sum_{\langle ij \rangle} r_{ij} \frac{\partial \phi}{\partial r_{ij}} \ . \qquad \text{(B.9)}$$

Substituting (B.9) into (B.6) gives

$$p = -\frac{1}{dV} \sum_{\langle ij \rangle} r_{ij} \frac{\partial \phi}{\partial r_{ij}} + \frac{NkT}{V} \qquad \text{(B.10)}$$

which is a general expression for the pressure of a system of particles that interact through an arbitrary pairwise potential.

We now consider the Lennard-Jones potential, which in Section 2.3 we gave as

$$\phi = 4\epsilon [(\frac{\sigma}{r})^{12} - (\frac{\sigma}{r})^{6}] \ . \qquad \text{(2.12)}$$

Substituting (2.12) into (B.10) and differentiating ϕ give

$$p = \frac{24\epsilon N}{dV} [2\sigma^{12}\langle r^{-12}\rangle - \sigma^{6}\langle r^{-6}\rangle] + \frac{NkT}{V} \qquad \text{(B.11)}$$

in which $\langle r^{-n} \rangle = \frac{1}{N} \sum_{\langle ij \rangle} r^{-n}$. Converting to reduced units and setting $d = 2$ give the final result for the pressure of the two-dimensional Lennard-Jones system, which is

$$p^{*} = 12\rho^{*} [2\sigma^{12}\langle r^{-12}\rangle - \sigma^{6}\langle r^{-6}\rangle] + \rho^{*} T^{*} \ . \qquad \text{(B.12)}$$

We note that the result that we obtained by deriving the pressure from the virial theorem is identical to the results in Section 4.1 and Appendix A, which were derived by different methods.

Appendix C: Normal Mode Analysis

In order to calculate $\frac{1}{N}\sum_{i=1}^{M}\ln\omega_i^*$, which appears in (5.29), we must solve for the normal mode frequencies of the hexagonal crystal that consists of N interacting particles. We begin by deriving the equations of motion for the N particles and solving them in terms of a general periodic function. After enumerating the wavevectors that the periodic boundaries of the system allow, we express the potential energy of the system in terms of the Lennard-Jones potential and solve for the frequencies of the normal modes. Finally, we report the results of the sum that enters (5.29) for several values of N and for two types of periodic boundary conditions.

Equations of Motion

We derive the equations of motion for the N particles using Feynman's notation,[37] except that we will consider only one particle per unit cell of the lattice. We label a particle in the crystal with $\vec{N}$, where $\vec{N} = n_1\vec{a}_1 + n_2\vec{a}_2$ in the two-dimensional system and the $\vec{a}_\alpha$ are the basis vectors of the crystal lattice. A convenient pair of basis vectors for the hexagonal crystal is[56]

$$\begin{aligned} \vec{a}_1 &= a\hat{x} \\ \vec{a}_2 &= \frac{a}{2}\hat{x} + \frac{\sqrt{3}a}{2}\hat{y} \end{aligned} \tag{C.1}$$

where a is the lattice constant and $\hat{x}$ and $\hat{y}$ are the unit vectors in the x and y directions, respectively. We define the basis vectors of the reciprocal lattice in terms of the basis vectors of the direct lattice as[57]

$$\vec{a}_\alpha \cdot \vec{b}_\beta = 2\pi\delta_{\alpha\beta} \ . \tag{C.2}$$

Substituting the basis vectors of the direct lattice from (C.1) into (C.2) gives

$$\vec{b}_1 = \frac{2\pi}{a}\hat{x} - \frac{2\pi}{\sqrt{3}a}\hat{y}$$
$$\vec{b}_2 = \frac{4\pi}{\sqrt{3}a}\hat{y} \tag{C.3}$$

which shows that the reciprocal lattice differs from the direct lattice by having a different lattice constant and being rotated by $\frac{\pi}{6}$.

We denote the displacement of the α coordinate of the particle at $\vec{N}$ from its equilibrium position as $Z_{\alpha,\vec{N}}$. Since we consider only small oscillations of the particles about their equilibrium positions in the analysis of normal modes, we can expand the potential energy of the crystal in powers of the displacement of the particles, which gives

$$U = U_0 + \sum_{\alpha,\vec{N}} \frac{\partial U}{\partial Z_{\alpha,\vec{N}}} Z_{\alpha,\vec{N}} + \frac{1}{2} \sum_{\alpha,\beta,\vec{N},\vec{M}} \frac{\partial^2 U}{\partial Z_{\alpha,\vec{N}} \partial Z_{\beta,\vec{N}+\vec{M}}} Z_{\alpha,\vec{N}} Z_{\beta,\vec{N}+\vec{M}} + \cdots \tag{C.4}$$

in which all of the derivatives are evaluated at $Z_{\alpha,\vec{N}} = 0$. Since U_0 is simply the potential energy of the static lattice, it cannot affect the dynamics of the particles, so we will ignore it. The first-order term vanishes since we evaluate the derivatives about the equilibrium positions. Considering only small oscillations of the particles allows us to ignore all but the first nonzero term, which leaves only the second-order term in the potential energy.

In terms of U we write the total force on the particle $\vec{N}$ due to the other particles as

$$\vec{F}_{\vec{N}} = -\frac{\partial U}{\partial \vec{Z}_{\vec{N}}} \ . \tag{C.5}$$

Substituting the second-order term of U from (C.4) into (C.5) gives

$$F_{\alpha,\vec{N}} = -\sum_{\beta,\vec{M}} C^{\vec{M}}_{\alpha\beta} Z_{\beta,\vec{N}+\vec{M}} \tag{C.6}$$

where we define

$$C^{\vec M}_{\alpha\beta} \equiv \left[\frac{\partial^2 U}{\partial Z_{\alpha,\vec N} \partial Z_{\beta,\vec N+\vec M}} \right]_{\vec z=0} . \tag{C.7}$$

Substituting $F_{\alpha,\vec N}$ into Newton's Second Law of Motion gives the equation of motion for each particle in the crystal, which is

$$m \, \ddot Z_{\alpha,\vec N} = -\sum_{\beta,\vec M} C^{\vec M}_{\alpha\beta} \, Z_{\beta,\vec N+\vec M} \tag{C.8}$$

where m is the mass of the particle.

In order to solve the equations of motion, we assume a general periodic function as the solution,

$$Z_{\alpha,\vec N} = a_\alpha(\vec k) \, e^{i\,(\vec k\cdot\vec N\, -\, \omega t)} . \tag{C.9}$$

Substituting (C.9) into (C.8) and differentiating give

$$m\omega^2 a_\alpha = \sum_{\beta,\vec M} C^{\vec M}_{\alpha\beta} \, a_\beta \, e^{i\vec k\cdot\vec M} . \tag{C.10}$$

Enumeration of Allowed Wavevectors

Before we can obtain the frequencies of the normal modes by solving (C.10), we must enumerate the allowed values of $\vec k$, which depend on the way the periodic boundaries are specified. Periodic boundary conditions mean that

$$\vec Z_{\vec N} = \vec Z_{\vec N+\vec M} \tag{C.11}$$

for certain values of $\vec M$. By substituting (C.9) into (C.11), we see that the periodic boundary conditions allow only the values of $\vec k$ that satisfy

$$\vec{k}\cdot\vec{M} = 2\pi m \tag{C.12}$$

where m is an integer.

A standard method of specifying periodic boundary conditions on a triangular lattice uses a 60° rhombus as a unit cell. Since the unit cell is periodic along two axes, we obtain two vectors that describe the periodicity of the lattice, which are

$$\begin{aligned} \vec{M}_1 &= M_1\vec{a}_1 \\ \vec{M}_2 &= M_2\vec{a}_2 \end{aligned} \tag{C.13}$$

where the $\vec{a}_\alpha$ are the basis vectors of the lattice. Substituting both vectors in (C.13) into (C.12) gives

$$\vec{k} = \frac{m_1}{M_1}\vec{b}_1 + \frac{m_2}{M_2}\vec{b}_2 \tag{C.14}$$

as the allowed wavevectors, in which the $\vec{b}_\alpha$ are the basis vectors of the reciprocal lattice and m_1 and m_2 are both integers.

As we mentioned in Section 2.3, using periodic boundary conditions with a unit cell that is a 60° rhombus has a disadvantage since the vectors in (C.13) are not mutually perpendicular. A more convenient specification of the boundary conditions uses a rectangular unit cell, which allows the simulation program to determine the boundary of the unit cell with less computation. The pair of vectors that describes the periodicity of the rectangular unit cell is

$$\begin{aligned} \vec{M}_1 &= M_1\vec{a}_1 \\ \vec{M}_2 &= M_2(\vec{a}_2 - \tfrac{1}{2}\vec{a}_1) \end{aligned} \tag{C.15}$$

where $\vec{M}_1$ and $\vec{M}_2$ express the periodicity in the x and y directions, respectively. As before, we substitute the two vectors that describe the periodicity of the system into

(C.12) to obtain the allowed wavevectors, which are

$$\vec{k} = \frac{m_1}{M_1}\vec{b}_1 + (\frac{m_2}{M_2} + \frac{m_1}{2M_1})\vec{b}_2 \qquad \text{(C.16)}$$

where m_1 and m_2 are again both integers.

We note that the two different types of periodic boundary conditions lead to different sets of allowed wavevectors. However, any effect that the details of the boundary conditions have on the system is a finite-size effect by definition. Thus, we expect that any differences due to the boundary conditions in the contribution of the normal modes to the free energy of the crystal will decrease as the number of particles in the system increases.

In (C.14) and (C.16) we express the allowed wavevectors in terms of the integers m_1 and m_2, but we did not specify the range of the integers. Since $\vec{k}\cdot\vec{M}$ defines a phase angle in (C.9), which has a period of 2π, only certain wavevectors correspond to distinct normal modes of the crystal. A convenient specification of such wavevectors is that they lie in the first Brillouin zone,[57] which is the region of space about a point in the reciprocal lattice which is closer to that point than any other point in the reciprocal lattice. Any wavevectors that lie on the boundary of the first Brillouin zone are weighted by a factor of $\frac{1}{2}$. In addition, we can further reduce the number of wavevectors that we use to calculate the frequencies of the crystal's normal modes by exploiting the symmetry of the hexagonal lattice. We calculate the unique frequencies by using the wavevectors that lie in $\frac{1}{12}$ of the first Brillouin zone and then weight them according to the number of times that they occur in the full zone.[38] We easily obtain the correct weighting for each frequency by initially giving it a weight of 12 and then dividing the weight by a factor of two for each boundary of the sector of the first Brillouin zone on which it lies.

Solving for the Normal Mode Frequencies

In order to calculate the frequencies of the crystal's normal modes, we solve (C.10) for each of the wavevectors that we discussed in the previous section. We assume that the total potential energy is the sum of a pairwise interaction potential over all distinct pairs of particles in the crystal, which we write as

$$U = \sum_{\langle \vec{N},\vec{N}+\vec{M}\rangle} \phi(r_{\vec{N},\vec{N}+\vec{M}}) \ . \tag{C.17}$$

Using (C.17) for the potential energy and referring to (C.7), we see that the only nonzero contributions to $C_{\alpha\beta}^{\vec{M}}$ come from pairs of particles that include the particle at $\vec{N}$. Each such pair makes two contributions to $C_{\alpha\beta}^{\vec{M}}$, one where $\vec{M} = 0$ and one where $\vec{M} \neq 0$. In addition, we can further simplify $C_{\alpha\beta}^{\vec{M}}$ by using

$$\frac{\partial}{\partial Z_{\alpha,\vec{N}+\vec{M}}} = -\frac{\partial}{\partial Z_{\alpha,\vec{N}}} \tag{C.18}$$

where $\vec{M} \neq 0$. Substituting (C.17) and (C.18) into (C.10) gives

$$m\omega^2 a_\alpha = \sum_{\beta,\vec{M}\neq 0} a_\beta (1 - e^{i\vec{k}\cdot\vec{M}}) \left[\frac{\partial^2 \phi(r_{\vec{M}})}{\partial Z_{\alpha,\vec{N}} \partial Z_{\beta,\vec{N}}} \right]_{\vec{Z}=0} \ . \tag{C.19}$$

Since the contributions to the sum in (C.19) from $\vec{M}$ and $-\vec{M}$ differ only by the sign of their exponent, we combine such pairs in the sum in order to eliminate the imaginary exponent and obtain

$$m\omega^2 a_\alpha = \sum_{\beta,\vec{M}} a_\beta 2(1 - \cos\vec{k}\cdot\vec{M}) \left[\frac{\partial^2 \phi(r_{\vec{M}})}{\partial Z_{\alpha,\vec{N}} \partial Z_{\beta,\vec{N}}} \right]_{\vec{Z}=0} \tag{C.20}$$

in which the sum includes only the values of $\vec{M}$ that lie in the upper half-plane, excluding $\vec{M} = 0$.

In order to solve (C.20) and obtain the normal mode frequencies, we define the following three quantities for convenience:

$$A \equiv \sum 2(1-\cos \vec{k}\cdot\vec{M})\frac{\partial^2\phi}{\partial x^2} \tag{C.21}$$

$$B \equiv \sum 2(1-\cos \vec{k}\cdot\vec{M})\frac{\partial^2\phi}{\partial x\partial y} \tag{C.22}$$

$$C \equiv \sum 2(1-\cos \vec{k}\cdot\vec{M})\frac{\partial^2\phi}{\partial y^2}\ . \tag{C.23}$$

Substituting (C.21), (C.22), and (C.23) into (C.20) and writing the result in matrix form give

$$\begin{bmatrix} A - m\omega^2 & B \\ B & C - m\omega^2 \end{bmatrix}\begin{bmatrix} a_x \\ a_y \end{bmatrix} = 0\ . \tag{C.24}$$

We solve (C.24) by setting

$$\det\begin{bmatrix} A - m\omega^2 & B \\ B & C - m\omega^2 \end{bmatrix} = 0\ . \tag{C.25}$$

The solution of the quadratic equation that results from expanding (C.25) is

$$m\omega^2 = \tfrac{1}{2}(A+C) \pm \tfrac{1}{2}\sqrt{(A-C)^2+4B^2}\ . \tag{C.26}$$

We calculated the frequency of the normal mode corresponding to each of the allowed wavevectors by summing A, B, and C separately for each value of $\vec{k}$ and substituting the results into (C.26). We then obtained the contribution of the normal modes to the free energy of the crystal by calculating $\frac{1}{N}\sum \ln\omega_i^*$, which appears in (5.29), after weighting each value of ω appropriately. In Table C.1 we report the contribution of the normal modes to the free energy at $\rho^* = .95$ for several values of N and for both types of periodic boundary conditions. The columns labeled *rhombus* and *rectangle* give the results for the boundary conditions in (C.13) and (C.15), respectively. We note that the difference in the results due to the specific form of the boundary conditions disappears as N increases. In addition, we observe that the contribution to the free energy approaches a limiting value as N increases.

Table C.1

Normal Mode Contribution to F^* at $\rho^* = .95$		
N	rhombus	rectangle
64	5.26046	5.26033
256	5.21523	5.21512
1024	5.19997	5.19993
4096	5.19514	5.19514
16384	5.19369	5.19368
65536	5.19326	5.19326
262144	5.19314	5.19314

Appendix D: Calculation of the Second Virial Coefficient

In this appendix we will derive an expression for the second virial coefficient of a system of N particles that interact through an arbitrary pairwise interaction potential.[58] We then consider the case of the Lennard-Jones potential and evaluate the resulting integral to obtain the second virial coefficient.[44,45] In Section 5.4 we introduced the virial expansion,[43] which is

$$p = kT(\rho + B_2\rho^2 + B_3\rho^3 + \cdots) \tag{5.49}$$

in which the virial coefficients, $B_2, B_3, \cdots$, depend on temperature but not on density.

In order to calculate the second virial coefficient, B_2, we express the pressure of the system in terms of the partition function, which we expand in powers of V^{-1}. In Section 5.1 we derived an expression for the pressure of the system in terms of its Helmholtz free energy, F, which is

$$\left(\frac{\partial F}{\partial V}\right)_T = -p \ . \tag{5.8}$$

Using $F = -kT \ln Z$ from (5.18) in Section 5.2, we write the pressure in terms of Z as

$$p = kT \left(\frac{\partial \ln Z}{\partial V}\right)_T \ . \tag{D.1}$$

In Section 2.2 we gave the partition function of the system at temperature T and volume V as

$$Z = \frac{1}{h^{2N} N!} \int_{-\infty}^{\infty} d^2p_1 \cdots d^2p_N \int_V d^2q_1 \cdots d^2q_N \, e^{-\beta E} \tag{2.2}$$

in which $\beta = (kT)^{-1}$, and E is the total energy of the system. In (2.2) p_i and q_i are the momentum and position of particle i, respectively.

We can ignore all factors in (2.2) that do not depend on V since they will cancel in (D.1). In addition, we rewrite the integrand in (2.2) as

$$Z = \int_V d^2q_1 \cdots d^2q_N \left[1 + \left(e^{-\beta U} - 1\right)\right] \tag{D.2}$$

so that we can expand $\ln Z$ in a power series. Using the series expansion of $\ln(1+x)$ gives

$$\ln Z = N \ln V + \sum_{m=1}^{\infty} \frac{-1^{m+1}}{m} \left[\frac{1}{V^N} \int_V d^2q_1 \cdots d^2q_N \left(e^{-\beta U} - 1\right)\right]^m . \tag{D.3}$$

Since the integral in (D.3) still contains different powers of V, we must also expand the exponential in a power series, which gives

$$\ln Z = N \ln V + \sum_{m=1}^{\infty} \frac{-1^{m+1}}{m} \left[\frac{1}{V^N} \sum_{n=1}^{\infty} \frac{(-\beta)^n}{n!} \int_V d^2q_1 \cdots d^2q_N \, U^n\right]^m . \tag{D.4}$$

In order to obtain the second virial coefficient, we must collect all of the terms in (D.4) that contain V^{-1}. We will assume that the potential energy can be written as

$$U = \sum_{\langle ij \rangle} \phi(r_{ij}) \tag{D.5}$$

where the sum extends over all distinct pairs of particles. In (D.5) ϕ is the pairwise interaction potential, and r_{ij} is the distance between particles i and j. If the integrand of (D.4) contained no interaction potential, the integral over each particle would contribute a factor of V, which would cancel the factor of $\frac{1}{V^N}$. However, since the index n must be at least one, the interaction potential must act between at least one pair of particles. When

ϕ acts between a single pair of particles and $m = 1$, the integrals over the particles contribute a factor of V^{N-1}, which gives us the desired V^{-1} terms in the expansion.

Keeping only the terms that contain the interaction of a single pair of particles in (D.4) and performing the integrals over the other particles give

$$\ln Z = N \ln V + \frac{1}{V^N} \sum_{n=1}^{\infty} \frac{(-\beta)^n}{n!} V^{N-1} \frac{N(N-1)}{2} \int_V d^2q \; \phi^n \tag{D.6}$$

where we integrate over the distance between the two interacting particles. The factor of $\frac{N(N-1)}{2}$ in (D.6) arises from the number of interacting pairs in the system. We will replace the factor with $\frac{N^2}{2}$ in subsequent equations because the difference between the two factors represents a finite-size effect, which disappears in large systems.

By performing the sum in (D.6) and converting the integration variables to polar coordinates, we obtain

$$\ln Z = N \ln V + \frac{N^2}{2V} \int_0^{2\pi} \int_0^{\infty} \left(e^{-\beta\phi} - 1\right) r \, dr \, d\theta \quad . \tag{D.7}$$

Substituting (D.7) into (D.1) and differentiating give

$$p = kT\rho - \tfrac{1}{2} kT\rho^2 \int_0^{2\pi} \int_0^{\infty} \left(e^{-\beta\phi} - 1\right) r \, dr \, d\theta \quad . \tag{D.8}$$

By comparing (5.49) and (D.8), we identify the second virial coefficient as

$$B_2 = -\pi \int_0^{\infty} \left(e^{-\beta\phi} - 1\right) r \, dr \tag{D.9}$$

where we have performed the integration over θ.

We now integrate (D.9) to obtain the second virial coefficient using the Lennard-Jones potential, which we gave in Section 2.3 as

$$\phi = 4\epsilon [(\frac{\sigma}{r})^{12} - (\frac{\sigma}{r})^{6}] \ . \tag{2.12}$$

Using $x = \frac{r}{\sigma}$, we convert (D.9) to reduced units, giving

$$B_2^* = -\pi \int_0^\infty (e^{-\frac{\phi^*}{T^*}} - 1) \, x \, dx \ . \tag{D.10}$$

In terms of x the Lennard-Jones potential is

$$\phi^* = 4(x^{-12} - x^{-6}) \ . \tag{D.11}$$

In order to make the numerical integration easier, we analytically integrate (D.10) in the limits, $x \rightarrow 0$ and $x \rightarrow \infty$. In the former limit, the exponential in the integrand of (D.10) approaches 0 since ϕ^* is large and positive. We denote the resulting integral as I_1, which is

$$I_1 = -\pi \int_0^\delta -x \, dx = \tfrac{1}{2}\pi\delta^2 \tag{D.12}$$

where δ is small enough that the exponential in (D.10) is always 0 to within the precision of the numerical integration that follows.

In the limit of $x \rightarrow \infty$ we expand the integrand in (D.10) in powers of x^{-6}, which gives

$$I_3 = -\pi \int_L^\infty [\frac{4}{T^*} x^{-6} - (\frac{4}{T^*} - \frac{8}{T^{*2}}) x^{-12}] \, x \, dx \tag{D.13}$$

where we have dropped terms of order x^{-18} and higher. Integrating (D.13) gives

$$I_3 = -\frac{\pi}{T^* L^4} + \frac{2\pi}{5T^* L^{10}} - \frac{4\pi}{5T^{*2} L^{10}} \ . \tag{D.14}$$

We determine an appropriate value for L by ensuring that the contribution of the terms that we dropped is less than the precision of the following numerical integration.

We evaluate the remaining integral,

$$I_2 = -\pi \int_{\delta}^{L} \left(e^{-\frac{\phi^*}{T^*}} - 1\right) x \, dx \tag{D.15}$$

by an appropriate numerical method, such as Simpson's rule. Combining the results of I_1, I_2, and I_3 for $T^* = .7$, $\delta = .5$, and $L = 5$ gives

$$B_2^* = -2.761223 \tag{D.16}$$

when we use a step size of .0001 in Simpson's rule. Our value of the second virial coefficient agrees closely with the value of –2.7613 that Morrison and Ross obtained.[44]

REFERENCES

1. R. E. Peierls, *Helv. Phys. Acta.* 7 Suppl II (1934) 81.

2. L. D. Landau, *Collected Papers of L. D. Landau*, ed. D. ter Haar (Pergamon, Oxford, 1965) Ch. 29.

3. N. D. Mermin, *Phys. Rev.* **158** (1967) 383.

4. J. M. Kosterlitz and D. J. Thouless, *J. Phys. C.* **6** (1973) 1181.

5. F. F. Abraham, *Phys. Rep.* **80** (1981) 339.

6. L. Landau and E. M. Lifshitz, *Theory of Elasticity* (Pergamon, New York, 1970) p. 10.

7. B. I. Halperin and D. R. Nelson, *Phys. Rev. Lett.* **41** (1978) 121.

8. D. R. Nelson and B. I. Halperin, *Phys. Rev. B* **19** (1979) 2457.

9. A. P. Young, *Phys. Rev. B* **19** (1979) 1855.

10. B. J. Alder and T. E. Wainwright, *Phys. Rev.* **127** (1962) 359.

11. F. Tsien and J. P. Valleau, *Mol. Phys.* **27** (1974) 177.

12. S. Toxvaerd, *Mol. Phys.* **29** (1975) 373.

13. S. Toxvaerd, *J. Chem. Phys.* **69** (1978) 4750.

14. D. Frenkel and J. P. McTague, *Phys. Rev. Lett.* **42** (1979) 1632.

15. S. Toxvaerd, *Phys. Rev. Lett.* **44** (1980) 1002.

16. F. F. Abraham, *Phys. Rev. Lett.* **44** (1980) 463.

17. F. van Swol, L. V. Woodcock, and J. N. Cape, *J. Chem. Phys.* **73** (1980) 913.

18. J. A. Barker, D. Henderson, and F. F. Abraham, *Physica* **106A** (1981) 226.

19. F. F. Abraham, *Phys. Rev. B* **23** (1981) 6145.

20. S. Toxvaerd, *Phys. Rev. A* 29 (1981) 2735.

21. J. Tobochnik and G. V. Chester, *Phys. Rev. B* **25** (1982) 6678.

22. F. Reif, *Fundamentals of Statistical and Thermal Physics* (McGraw-Hill, New York, 1965) p. 201.

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

24. O. G. Mouritsen, *Computer Studies of Phase Transitions and Critical Phenomena* (Springer-Verlag, New York, 1984) Ch. 2.

25. W. F. Brinkman, D. S. Fisher, and D. E. Moncton, *Science* **217** (1982) 693.

26. S. Otto, A. Kolawa, and A. Hey, *Performance of the Mark II Caltech-JPL Hypercube* (unpublished).

27. A. Frey, *Hypercube Architectures and Their Application to Signal Processing* (unpublished).

28. M. Johnson, *An Interrupt-Driven Communication System* (unpublished).

29. J. Salmon, *A Program to Automatically Decompose a Physical Lattice into a Hypercube* (unpublished).

30. G. M. Amdahl, *AFIPS Conference Proceedings* **30** (Thompson, Washington D.C., 1967) p. 483.

31. G. Fox, *On the Sequential Component of Computation* (unpublished).
    G. Fox, *The Performance of the Caltech Hypercube in Scientific Calculations* (unpublished).

32. E. Felten, S. Karlin, and S. Otto, *Sorting on a Hypercubic, MIMD Computer* (unpublished).

33. G. Fox, R. Gupta, O. Martin, and S. Otto, Caltech preprint, CALT-68-866 (1981).

34. F. Reif, *op. cit.*, p. 301.

35. F. Reif, *op. cit.*, p. 214.

36. F. Reif, *op. cit.*, p. 152.

37. R. P. Feynman, *Statistical Mechanics* (W. A. Benjamin, Inc., Reading, 1972) p. 12.

38. J. M. Phillips and L. W. Bruch, *Surface Sci.* **81** (1979) 109.

39. H. D. Young, *Statistical Treatment of Experimental Data* (McGraw-Hill, New York, 1962) p. 101.

40. G. Dahlquist and A. Bjorck, *Numerical Methods* (Prentice-Hall, Inc., Englewood Cliffs, 1974) p. 101.

41. G. Dahlquist and A. Bjorck, *op. cit.*, p. 131.

42. G. Dahlquist and A. Bjorck, *op. cit.*, p. 453.

43. F. Reif, *op. cit.*, p. 422.

44. I. D. Morrison and S. Ross, *Surface Sci.* **39** (1973) 21.

45. E. D. Glandt, *J. Chem. Phys.* **68** (1978) 2952.

46. D. R. Squire, A. C. Holt, and W. G. Hoover, *Physica* **42** (1969) 388.

47. D. J. Wallace, *Thermodynamics of Crystals* (Wiley, New York, 1972) p. 14.

48. L. Landau and E. M. Lifshitz, *op. cit.*, p. 2.

49. L. Landau and E. M. Lifshitz, *op. cit.*, p. 38.

50. H. Goldstein, *Classical Mechanics* (Addison-Wesley, London, ed. 2, 1980) p. 192.

51. M. Born and K. Huang, *Dynamical Theory of Crystal Lattices* (Oxford University Press, Oxford, 1954) p. 129.

52. R. H. Morf, *Phys. Rev. Lett.* 43 (1979) 931.

53. V. Privman and M. E. Fisher, *J. Stat. Phys.* 33 (1983) 385.

54. G. Fox, M. Johnson, G. Lyzenga, S. Otto, and J. Salmon, *Solving Problems on Concurrent Processors* (Prentice-Hall, Inc., Englewood Cliffs, to be published).

55. H. Goldstein, *op. cit.*, p. 82.

56. N. W. Ashcroft and N. D. Mermin, *Solid State Physics* (Saunders College, Philadelphia, 1976) p. 77.

57. N. W. Ashcroft and N. D. Mermin, *op. cit.*, Ch. 5.

58. R. P. Feynman, *op. cit.*, p. 100.