UNIVERSITY OF ILLINOIS

MAY 13, 1992

THIS IS TO CERTIFY THAT THE THESIS PREPARED UNDER MY SUPERVISION BY

TOM ALAN PASMORE, JR.

ENTITLED A MOLECULAR DYNAMICS STUDY OF LIQUID-SOLID COEXISTENCE

IS APPROVED BY ME AS FULFILLING THIS PART OF THE REQUIREMENTS FOR THE

DEGREE OF BACHELOR OF SCIENCE IN CHEMICAL ENGINEERING

Instructor in Charge

APPROVED: Richard Alkire

HEAD OF DEPARTMENT OF CHEMICAL ENGINEERING

O-1364

A Molecular Dynamics Study of Liquid-Solid Coexistence

by

Tom Alan Pasmore, Jr.

Thesis

for the
Degree of Bachelor of Science
in
Chemical Engineering

College of Liberal Arts and Science
University of Illinois
Urbana, Illinois

1992

Abstract

In this work we report on four different studies of fluid-crystal coexistence involving molecular dynamics simulations. The investigations described herein include wetting on structured walls, determination of the coexistence density of a fluid crystal system, prefreezing in a coexistence system, and spontaneous segregation of a highly ordered fluid into liquid and solid phases. Molecular dynamics were used in order to examine several systems which would be physically. The simulations using the hard sphere are accurate enough to permitted limited applicability. While the preliminary results demonstrate intuitive conclusions, many more simulations are required before complete characterization of these systems can actually been realized.

Acknowledgements

I'd like to thank Spike for the help and guidance in the realization of this thesis and my graduate career, but more importantly for all the cool tunes; Frank for the chance he gave me way back in August; the rest of the group for showing me that computers really aren't out to get me, and Jess for her neverending supply of faith and encouragement. Lastly, special thanks to Frederico Zamboni, may he rest in peace.

Table of Contents

Introduction

Phase transitions are a major part of the world around us, and of ongoing interest within chemical engineering. The solid-liquid transition has been one of the leas studied due to the difficulty of examining the interface of the two dense phases. Studies of two coexisting phases naturally lead into the question the wetting state in the presence of a substrate. Cahn sparked a renewed interest in wetting state studies with his work [1]. Considering, most of these transitions occur on an atomic level and are experimentally difficult to study, molecular dynamics appears the easiest way to collect preliminary data on the system.

Molecular Dynamics (MD) computer simulation has been a useful tool since Alder and Wainwright introduced it in the late 50's [2,3]. Simulations have allowed scientist and engineers to probe unique systems at conditions that would difficult or impossible to physically reproduce in laboratory.

The procedures and methods of MD simulation are discussed thoroughly in Allen and Tildeseley [4]. In the these simulations the computer solves the equations of motion for each particle in the system, as a function of the potential of the system. The hard sphere potential model is one of the simplest models of atomic interactions. It states that the pair interaction is infinite when the particles are separated by a distance less than σ, the atomic diameter, or zero otherwise. By selecting the hard sphere potential, the system can be solved exactly using classical mechanics. Thus assigning a configuration of particles both position and velocity, the thermodynamic state of the system has been defined. The equation of state for hard spheres appears in figure 1, and is given by

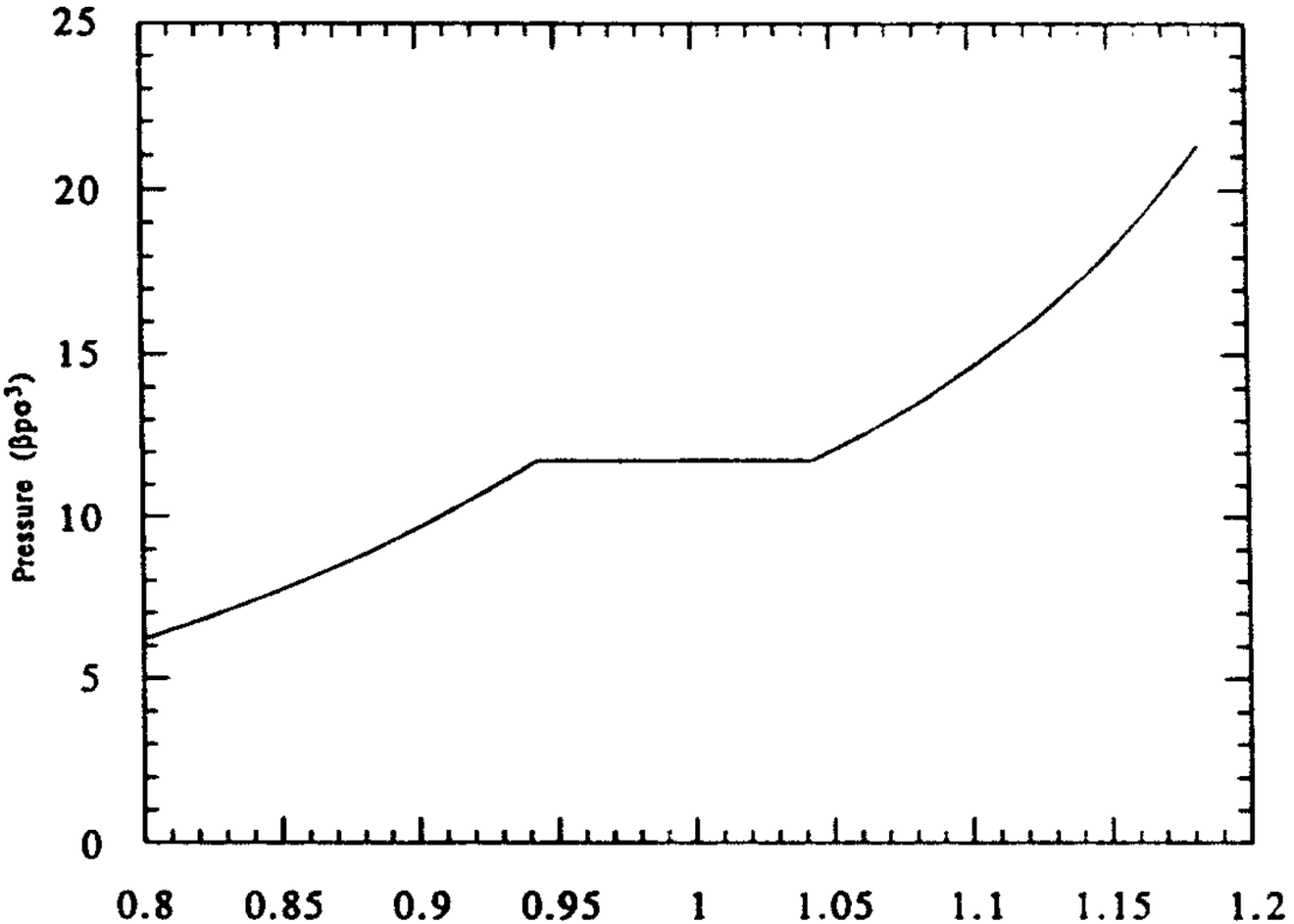


figure 1 - Equation of State for Hard Spheres

$$\frac{\beta p}{\rho^*} = \frac{1+\eta+\eta^2}{(1-\eta)^3} \qquad \text{fluid (1)}$$

$$\frac{\beta p}{\rho^*} = \frac{1+\eta+\eta^2-0.67825\eta^3-\eta^4-0.5\eta^5-6.028e^{(7.9-3.94)\eta^6}}{1-3\eta+3\eta^2-1.04305\eta^3} \qquad \text{solid (2)}$$

where:

$$\eta \equiv \frac{\pi}{6}\frac{n\sigma^3}{V}$$

$$\xi \equiv \pi\sqrt{2}/6-\eta$$

Using MD with hard spheres, each particle is given a unique position and velocity vector. For the hard sphere model no forces act on any particle except upon collision, therefore all particles move in a straight line with constant velocity until such time as they come into contact with another particle. The collision time for each pair of particles, i and j, is then calculated using equation (3):

$$\Delta t_{ij} = -\frac{b_{ij}+\sqrt{b_{ij}^2-\vec{v}_{ij}\cdot\vec{v}_{ij}\times(\vec{r}_{ij}\cdot\vec{r}_{ij}-\sigma^2)}}{\vec{v}_{ij}\cdot\vec{v}_{ij}} \qquad (3)$$

where $\vec{r}_{ij} = \vec{r}_i - \vec{r}_j$, $\vec{v}_{ij} = \vec{v}_i - \vec{v}_j$, and $b_{ij} = \vec{r}_{ij}\cdot\vec{v}_{ij}$.

From all of these times the shortest one is selected and particles are advanced for this amount of time with there current velocities. Now the position for each particle is:

$$\vec{r}_{i,t+\Delta t} = \vec{r}_{i,t} + \vec{v}_i \times \Delta t \qquad (4)$$

This brings the colliding particles within a distance σ from each other. At this point, new velocity vectors are calculated for both colliding particles from equation (5) with both momentum and kinetic energy being conserved.

$$\Delta \vec{v} = \frac{\vec{r}_{ij} \times \vec{b}_{ij}}{\sigma^2} \tag{5}$$

This equation implies that the expressions are all evaluated at the point of contact between the two particles. Now the process is repeated until the desired number of collisions have been completed.

The main disadvantage of the hard sphere potential model is the over simplification of the system. In our studies we have explored several liquid-solid wetting and coexistence phenomena using the hard sphere potential. The choice of hard sphere potential, although eliminating all interactions between particles, is still able to capture the essential physics of the problem, and is applicable primarily to noble element systems.

A Preliminary Study of Wetting on Structured Hard Walls

When a multi-phase system is in the presence of a substrate one phase or the other may preferentially contact the surface. Complete wetting by one phase describes the state where that phase completely covers the substrate. Partial wetting is the state where both phases are simultaneously in contact with the substrate. The wetting state can be described by the contact angle, θ, where θ is defined below as a function of the surface free energies [5]:

$$\gamma_{\alpha\beta}\cos(\theta) = \gamma_{\alpha\chi} - \gamma_{\beta\chi} \quad (6)$$

where $\gamma_{\alpha\beta}$ is the surface free energy of the crystal and liquid, $\gamma_{\alpha\chi}$ is the surface free energy of the wall and the crystal, and $\gamma_{\beta\chi}$ is the surface free energy between the fluid and the wall.

The study conducted on the wetting of smooth walls by Courtemanche and van Swol (C&vS) [6] has laid the foundation for the current study. They demonstrated that a smooth hard wall is completely wet by crystal ($\cos(\theta)=1$). We proceeded to survey various wall configurations in an effort to examine the speed of nucleation and attempt to force wetting by fluid ($\cos(\theta)=0$). The unsystematic approach we have used has yielded some initial results, which will need more work to thoroughly develop.

METHODOLOGY

The system used in the simulations contains 2100 hard sphere particles enclosed by two hard walls in the z direction and periodic boundary conditions (PBC) in both the x and y directions. The dimensions of the box were 5.53764σ by 9.59174σ by 40.234752σ, where σ is the diameter of an individual particle. The particles are in a canonical ensemble at liquid solid coexistence pressure. The configuration was constructed from a hard sphere crystal with the fcc (111) plane aligned parallel to the walls of the box. A section of the crystal was then stretched by spreading the distance in between the crystal planes. This resulted in a lowering of density within the system and thus the decomposition of this region of the bulk crystal into bulk fluid. The densities referred to are defined by:

$$\rho^* = \frac{N\sigma^3}{V} \quad (7)$$

where N is the number of particles and V is the volume of the box . For hard spheres, crystal phase density, ρ^*_c , is equal to 1.0409 and ρ^*_f is 0.943.

The simulations were run on various wall configurations with the system described above. Those conditions surveyed included 50 particles in perfect bulk crystal arrangement on both walls, crystal on one wall and smooth surface on the other, 43 particles whose positions are randomly displaced in directions across the x-y plane, and a 43 particle configuration randomly displaced in all directions. By observing the density profiles from each simulation we can identify the type of wetting occurring.

RESULTS

The first wall configuration to be examined was full crystal on both walls. To accommodate the structure on the wall the distance along the z axis was increased to 40.984752σ, in order to avoid overlaps. The various simulations showed marked acceleration in the crystal formation. In the smooth walled experiments previously done by C&vS nucleation occurred in $4x10^7$ collisions with complete crystallization on both walls in $7x10^7$ collisions. The current system, however, nucleated on both walls in under $3.5x10^6$ collisions and became completely symmetric in $19x10^6$ collisions,as shown in figure 2. This growth occurred approximately ten times faster than the smooth walled system.

Next, the right wall was replaced with a smooth hard wall at the same distance used for the structured wall simulation. Although the amount of crystal decreased, both walls retained the crystal wetting This was done to

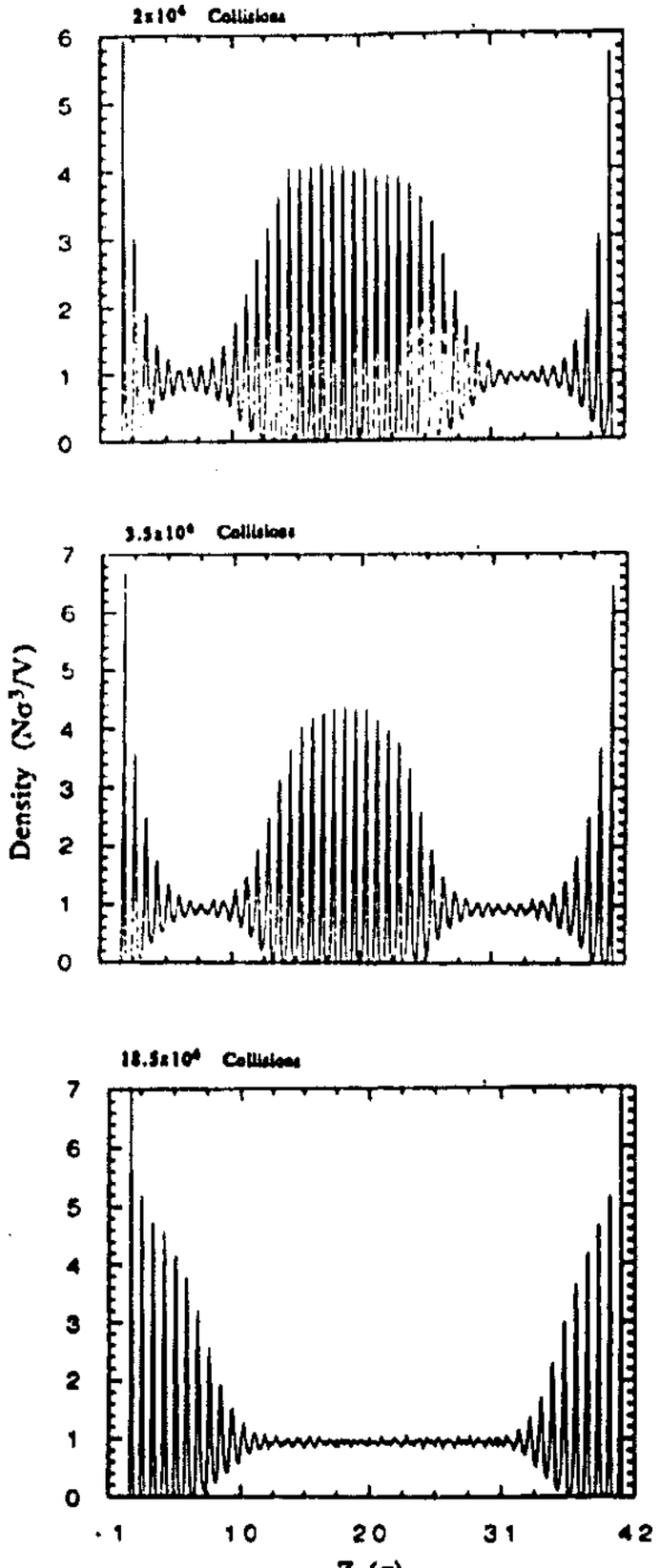


figure 2 - Density Profiles
Wetting on Crystal Structured Walls

check that any melting that might occur in the structured wall simulation was due to the structure on the wall and not the increase in volume of the system. The volume increase by putting a smooth wall at a given position is greater than that of placing a structured wall at the same point. As shown by the graphs in figure 3, we initially lost all but one of the crystal layers, but in the end two more layers had recrystallized on the wall before equilibrium was reestablished. Thus the crystal would not completely melt due to the change in volume. Required for the introduction of structured walls would be necessary.

The second series of experiments tested the ability of a crystal to wet various surfaces. Knowing that a smooth wall provided enough order to promote nucleation, we examined surfaces that would disorder the hard spheres in the wall region to see if wetting by fluid could be induced. The first wall tested had 7 particles removed from the 50 required for a perfect crystal layer and the rest randomly distributed across the surface of the wall, while maintaining their original position in the z direction. The starting configuration, figure 4, of the system was such that the wall was already wet by crystal. This configuration was obtained from an equilibrium system where both walls had a perfect crystal structure.

The number of layers on the disordered wall was monitored by use of the density profile. The density profile in figure 4 showed that the total number of layers on the right wall never dropped below five, and thus retaining crystal wetting on the right wall. When the configuration was rerun with fluid wetting the walls instead of crystal, nucleation was not observed even after 4×10^7 collisions. The density profiles are show in

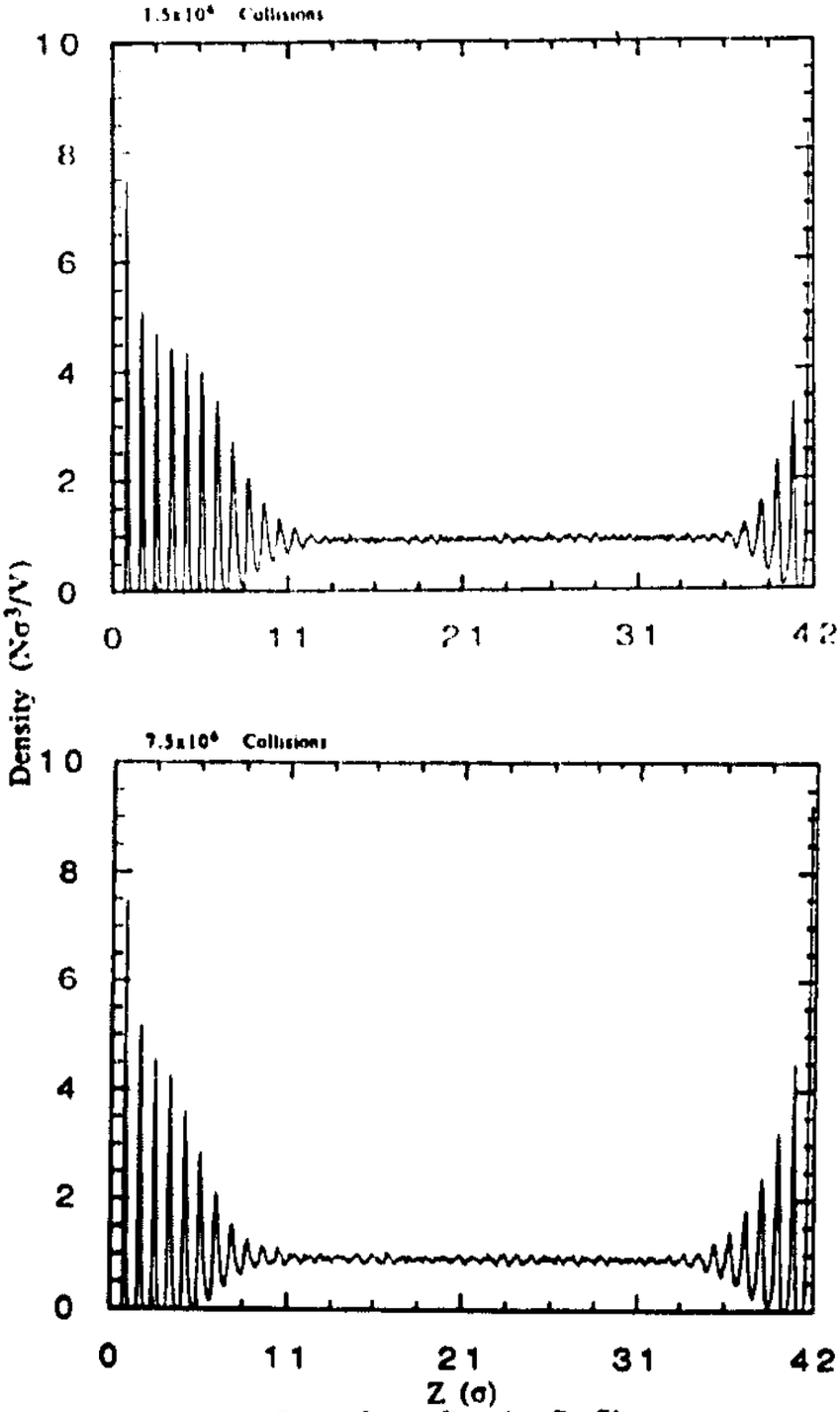


figure 3 - Density Profiles
Wetting on a Crystal Structured Wall & Smooth Wall

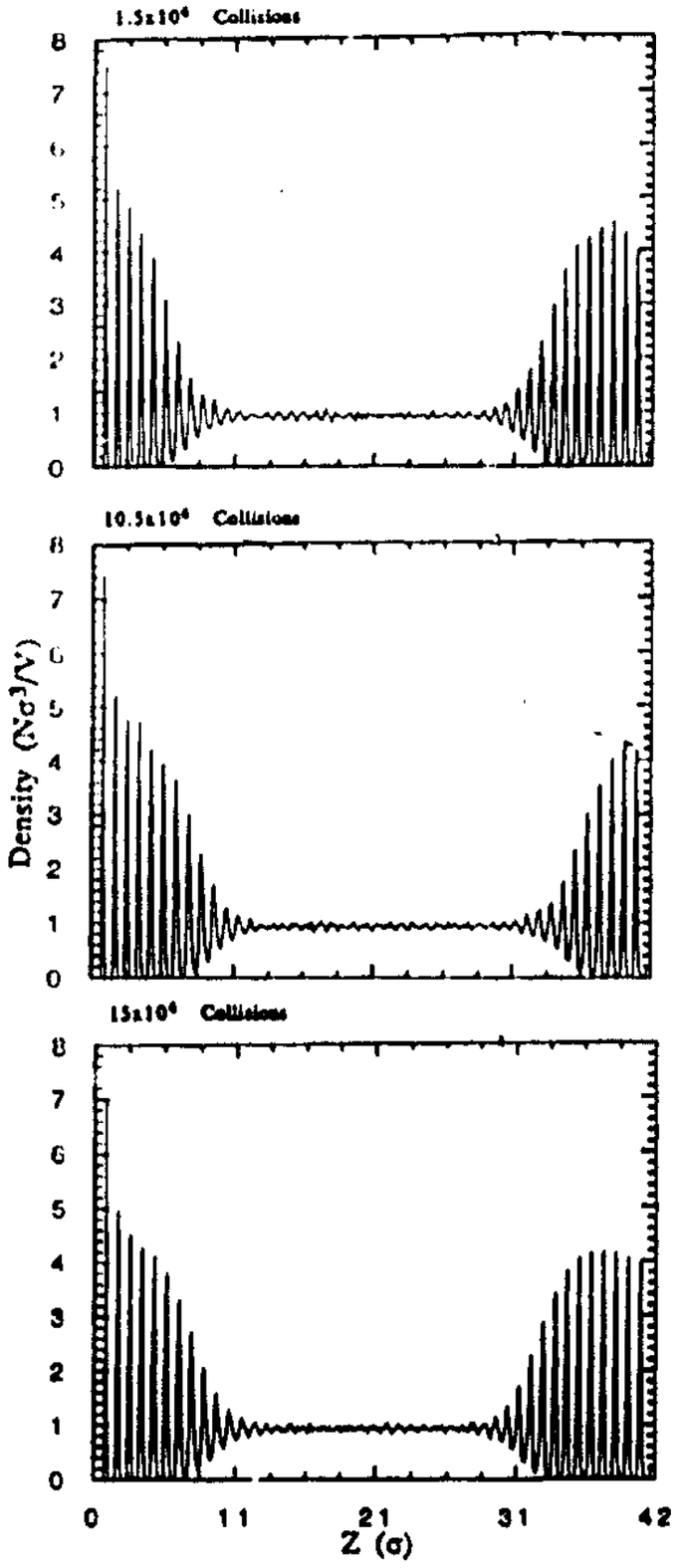


figure 4 - Density Profiles
Wetting on a Crystal Structured Wall &
a Randomized Structured Wall with
Previous Crystal Wetting

figure 5. However, it is not clear whether it is impossible for nucleation to occur against this wall or if more collisions are necessary to observe nucleation. The results are simply inconclusive due to the length of the simulation.

A system was also studied where the 43 particles on the right wall were also randomized with respect to the z direction. Although the system began with crystal wetting on the wall, the randomness of the structure forced wetting by fluid after only 4.5×10^6, as shown in figure 6. The completely random configuration is essentially a snapshot of amorphous static fluid and thus can only allow the wetting by fluid that we observe.

DISCUSSION

The results of the various wall configuration shows a few initial conclusions. First, the speed of crystallization is increases with the number nucleation sites available. The bulk crystal walls nucleated the fastest because they had large numbers of sites available, while the slowest walls were the smooth walls which has no sites on the surface.

Secondly, wetting by crystal is a minimum in the surface free energy of the system. This is demonstrated by crystal wetting the smooth walls and in one case the randomized walls. From our results it appears that order in the direction perpendicular to the interface is necessary condition for crystallization to occur, but more work is needed to confirm that hypothesis.

The results from the static fluid wall make intuitive sense as it was observed that the stabilization in the z direction provided by a smooth wall was sufficient to induce nucleation. The second series of structured wall

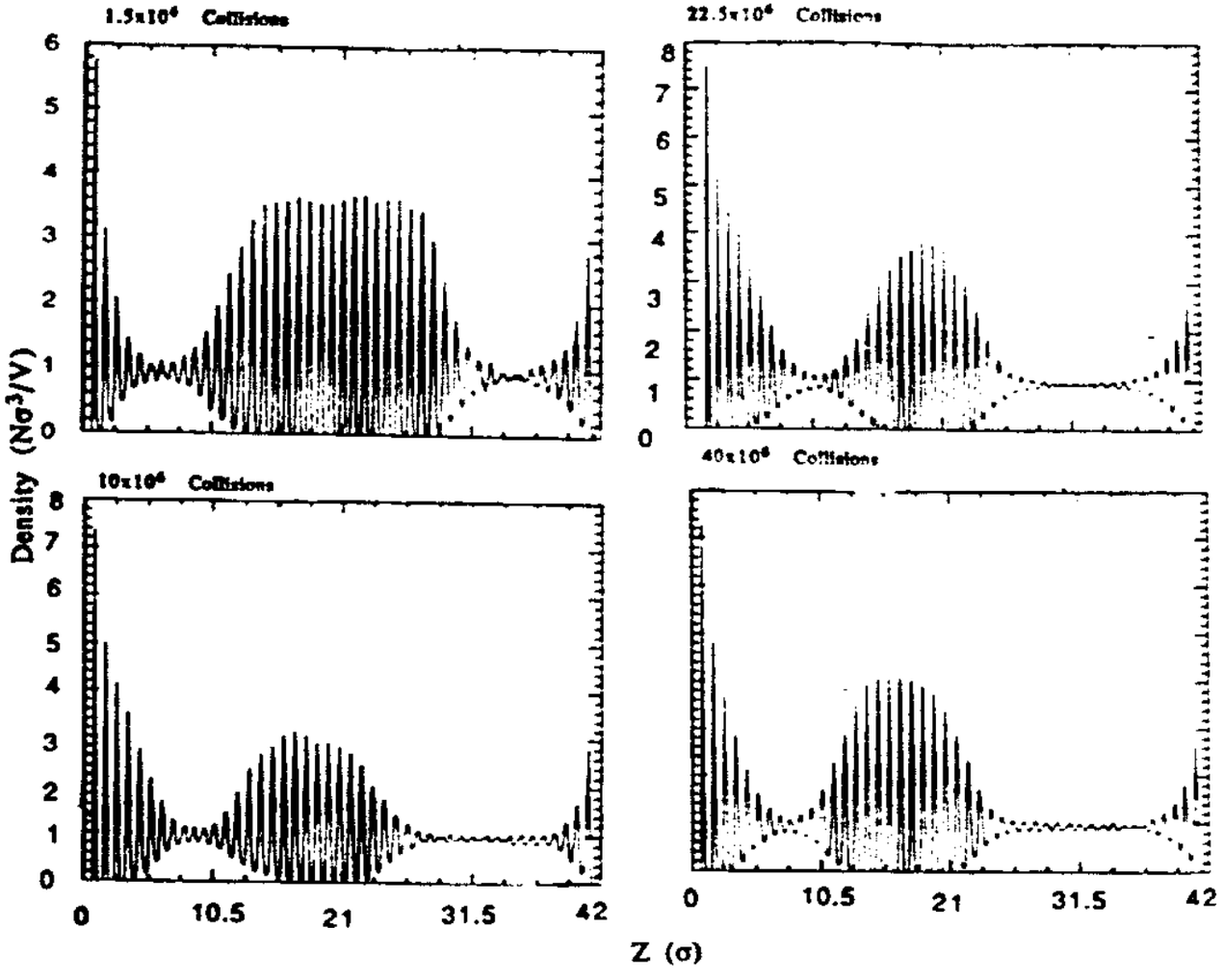


figure 5 - Density Profiles Wetting on a Crystal Structured Wall & a Randomized Structured Wall with Previous Fluid Wetting

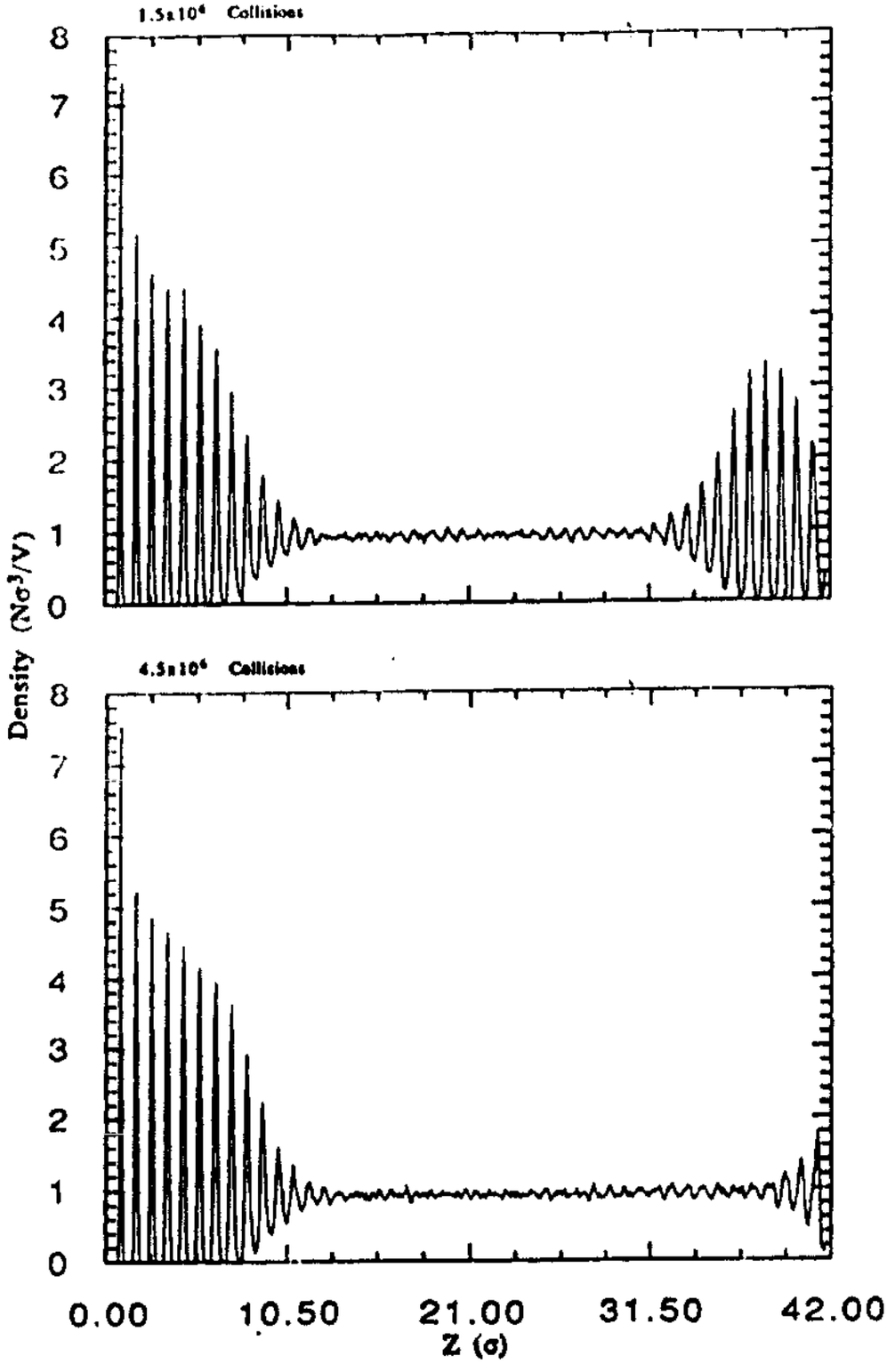


figure 6 - Density Profiles
Wetting on a Crystal Structured Wall &
a Fluid Structured Wall

simulations also indicated that stabilization in the z direction was the dominant factor in crystal wetting.

Determination of Liquid-Solid Coexistence Density for Hard Spheres

By examining a plot of the equation of state for hard spheres as a function of density, there is a region of the curve where the derivative goes to zero. This is the excluded region joining the Hall equation for hard sphere solids [7] and Carnahan-Starling equation for hard sphere fluids [8], and is referred to as a tie bar. The original work on the location of the transition was done by Hoover and Ree [9,10]. They located the melting transition for hard spheres by simulating a solid crystal at various densities. The determination of the transition density was done by using pressure measurements which were used to calculate communal entropy. We wish to confirm the observations seen previously, by incorporating the simulation of a two phase coexistence system in to the study.

METHODOLOGY

The hard sphere particles were used in two different NVT systems with PBC, both of which were involved in the experimentation. One system was a single crystal of 750 particles which were set at the specified densities and run to obtain the resulting pressures. Pressures are calculated by the relative positions and velocities of the particles colliding, as shown below:

$$p_t = k_B T n(z) - \frac{m}{At} \sum_{\nu}^{\nu} \frac{\vec{x}_{ij}^2 + \vec{y}_{ij}^2}{\sigma^2} (\vec{r}_{ij} \cdot \vec{v}_{ij}) \frac{1}{|\vec{z}_{ij}|} \theta\left(\frac{z - \vec{z}_i}{\vec{z}_{ij}}\right) \theta\left(\frac{\vec{z}_j - z}{\vec{z}_{ij}}\right) \qquad (8)$$

$$p_n = k_B T n(z) - \frac{m}{At}\sum_{\gamma}^{\nu}\frac{\vec{z}_{ij}^2}{\sigma^2}(\vec{r}_{ij}\cdot\vec{v}_{ij})\frac{1}{|\vec{z}_{ij}|}\theta(\frac{z-\vec{z}_i}{\vec{z}_{ij}})\theta(\frac{\vec{z}_j-z}{\vec{z}_{ij}}) \qquad (9)$$

where $\vec{r}_{ij}$ is $\vec{r}_i - \vec{r}_j$, $\vec{v}_{ij}$ is $\vec{v}_i - \vec{v}_j$, $\vec{z}_{ij}$ is $\vec{z}_i - \vec{z}_j$ (the difference in z location), θ is the Heaviside unit step function, p_t is the transverse component of the pressure, p_n is the normal component of the pressure, and v(t) is the number of collisions occurring in time t. These equations are described more thoroughly in van Swol and Henderson [11]. The bulk crystal configuration was manufactured by the simulation program for each individual density.

The second system is a 2100 particle, two phase PBC system at coexistence. It was created from earlier work by C&vS, and is shown in figure 7. It is being run simultaneously with the single phase simulation to determine system pressure. When the one phase and two phase pressures are the same, within error limits, true coexistence has been achieved. We are searching for the density or densities for which this the case.

RESULTS

The first crystal density studied was 1.307 $N\sigma^3/V$, which came from the PBC coexistence runs done in the wetting study. The pressure history, figure 8, for both phases was plotted after 3×10^7 collisions to determine the mean pressure for both systems. The mean pressure of the one phase system, 11.574 $\beta p\sigma^3$, was above the equivalent two phase pressure, 11.489 $\beta p\sigma^3$. This indicated that this density was too high for true coexistence, so a new density was chosen.

A crystal density of 1.0361 was then selected because it was calculated to produce a coexistence system at a normal pressure of 11.50 $\beta p\sigma^3$. However, after running this system for 6.5×10^7 collisions, this density

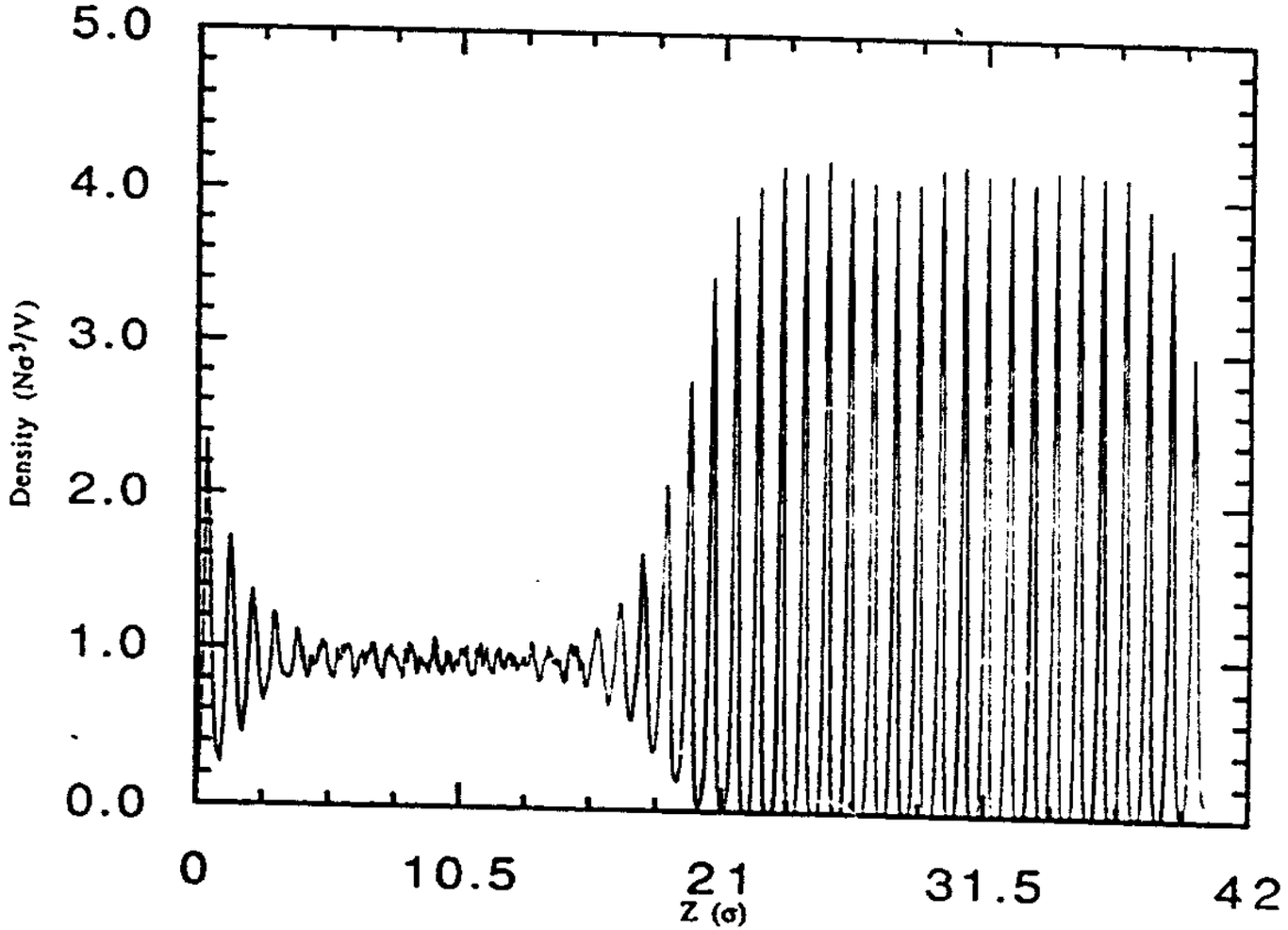
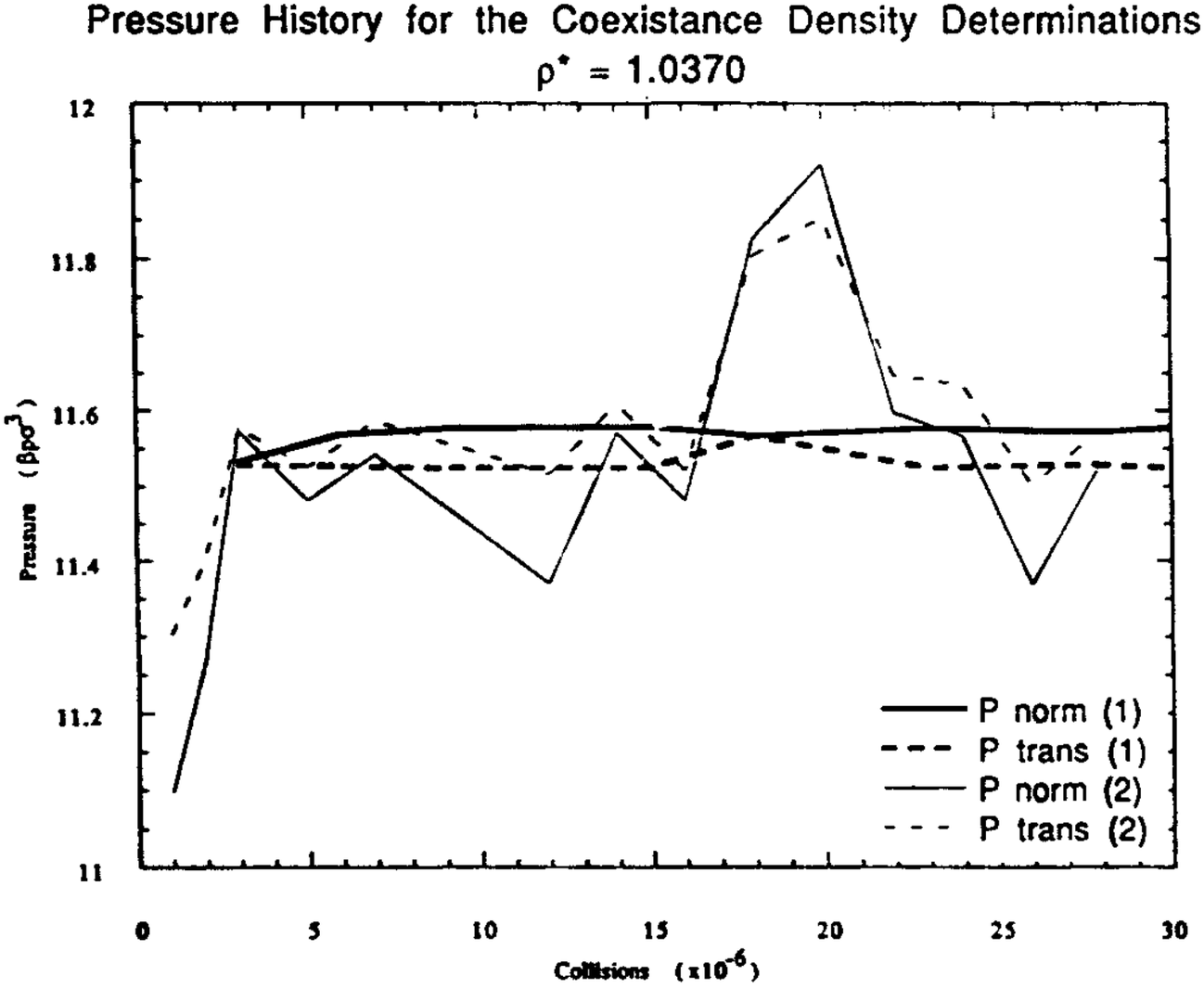


figure 7 - Density Profile
Initial System Configuration of Two Phase PBC Experiments

figure 8


Pressure History for the Coexistance Density Determinations
ρ* = 1.0370
12
11.8
11.6
11.4
11.2
11
Pressure (βpσ³)
0
5
10
15
20
25
30
Collisions (x10⁻⁶)
P norm (1)
P trans (1)
P norm (2)
P trans (2)

produces a mean two phase pressure that was still below the 11.50 $\beta p\sigma^3$. The one phase system equilibrates at a normal pressure of 11.45 $\beta p\sigma^3$ while the two phase mean is 11.39 $\beta p\sigma^3$, as shown in the pressure history, figure 9.

The last density studied was $\rho_c^* = 1.0348$. The resulting mean pressures, from figure 10, of 11.48 $\beta p\sigma^3$ for the one phase and 11.44 $\beta p\sigma^3$ for the two phase demonstrate, within error limits, agreement between the two systems. Therefore we believe this density is correct for true coexistence of this system.

DISCUSSION

From the three experiments performed ($\rho_c^* = 1.037$, 1.0361, & 1.0348) we have observed that the crystal density of coexistence is equal to 1.0348. We observed the best correlation between the one and two phase pressures at this density. The fluctuations present in two phase pressure histories greatly limit the accuracy with which our two systems can be compared. Longer simulations would allow a better curve averaging and thus produce more consistent data. The study should also be expanded to include several more densities to insure thoroughness. It was observed that the normal pressure was generally less than the transverse pressures for the two phase simulations, while the one phase system demonstrated the opposite trend. This is possibly due to the non cubic shape of the box.

Determination of Prefreezing Conditions

There exists a hysteresis loop in the curve of adsorption versus pressure (or, alternatively, chemical potential). This states that there is region of pressures where one may find wetting by either crystal or fluid,

figure 9

Pressure History for the Coexistance Density Determinations

$\rho^* = 1.0361$

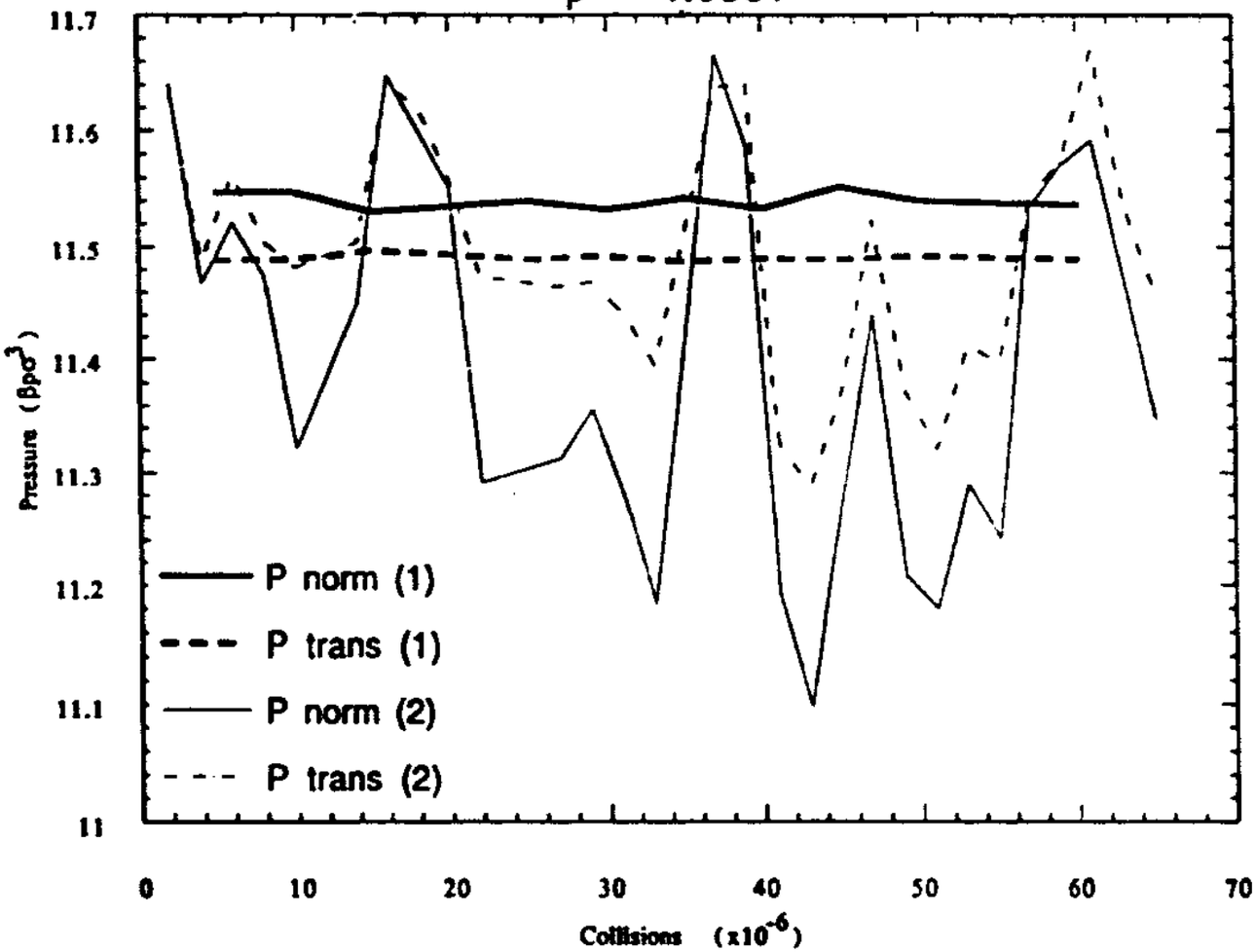
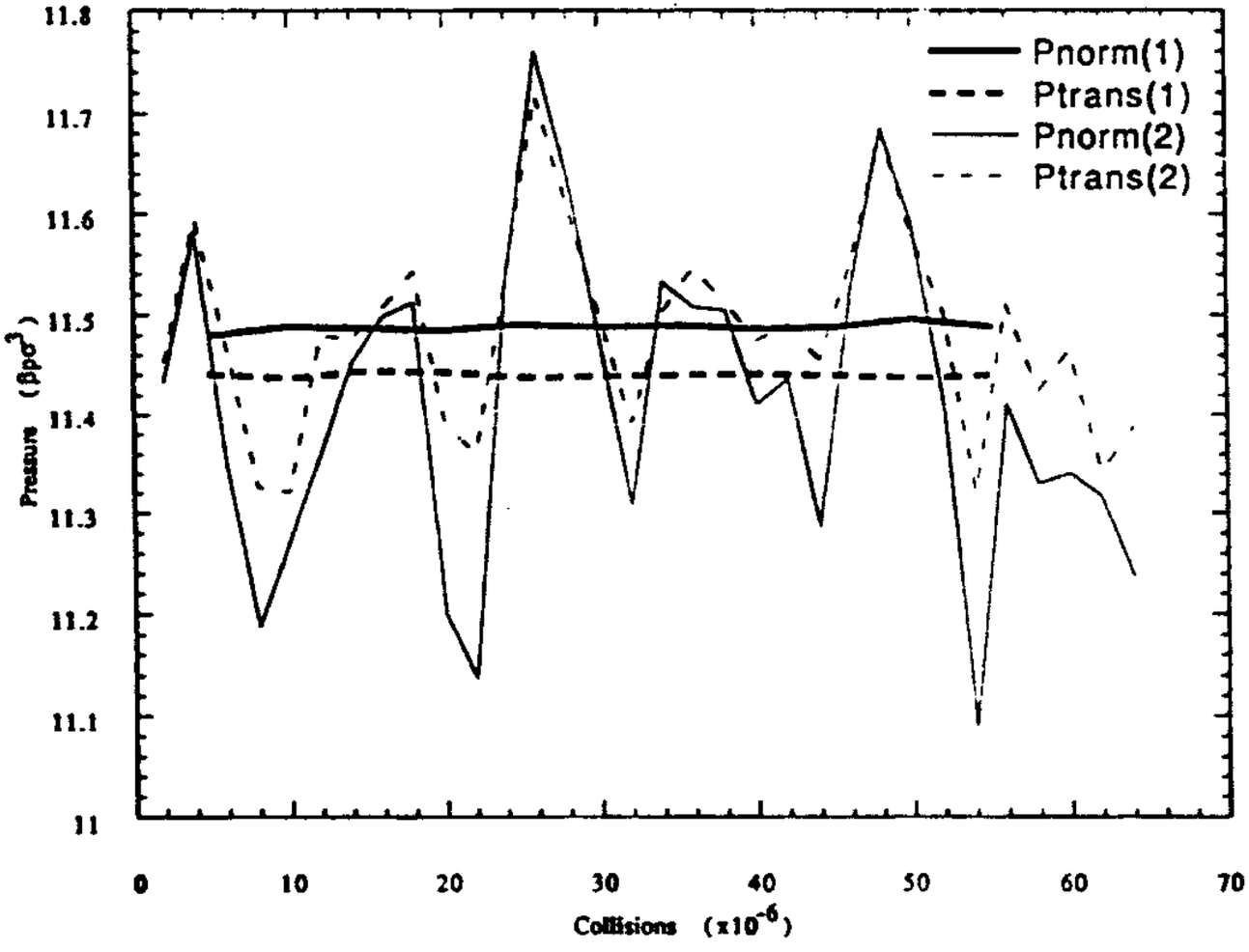


figure 10

Pressure History for the Coexistance Density Determinations

$\rho^* = 1.0348$



depending upon whether that pressure has been approached from above or below.

In our study, we are looking for the point where only one crystal layer is left against the smooth hard wall after the rest have been melted off. This would be the lowest pressure exhibiting prefreezing. Prefreezing is a metastable condition which is difficult to isolate.

METHODOLOGY

An NPT (isobaric/isothermal) ensemble was created by taking a configuration of 2100 hard spheres. between hard walls from the NVT wetting study with a density of $\rho_c^* = 1.0409$ at a pressure of 11.4 $\beta p\sigma^3$ and allowing the walls to move under a constant external pressure. The system is able to maintain constant pressure by accelerating the walls into the system at a constant force. This force applies an equilibrating pressure on the system and adjusts the volume and density accordingly.

Various pressures were used to obtain the conditions described above. The range of pressures were 11.0 to 11.3 $\beta p\sigma^3$, in order to cover the complete range of the hysteresis curve of the crystal phase. The simulations were run until equilibrium was obtained or the crystal disintegrated into fluid, which ever came first.

RESULTS

The first simulations run were at 11.3 $\beta p\sigma^3$. After $4x10^6$ collisions the density profile, figure 11, shows at least three layers of crystal on each wall. This condition persists through $1x10^7$ collisions and up to $2x10^7$ collisions. At this point it was determined that the system had equilibrated, and simulation was discontinued.

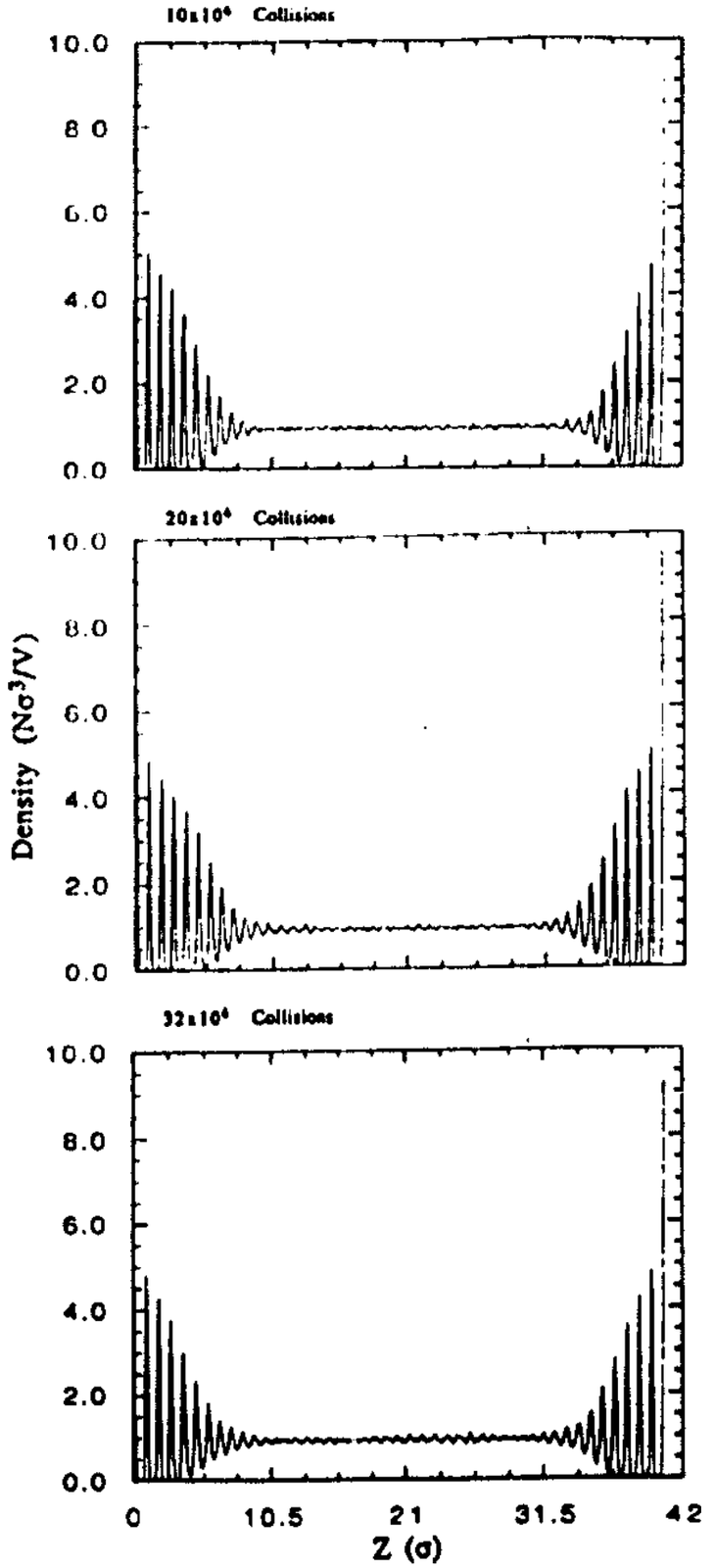


figure 11 - Density Profiles
NPT System at 11.3 $\beta p\sigma^3$

The next pressure selected was chosen to cause complete melting of the crystal in order to determine a range for prefreezing conditions. We selected 11.0 $\beta p\sigma^3$ for this purpose. The starting configuration of the system was the same as that chosen for the previous pressure. After 2.4×10^7 collisions, shown in figure 12, the system was observed to have melted. Equilibrium was then reached in 8×10^6 more collisions and simulation was discontinued.

11.1 $\beta p\sigma^3$ was selected as the next pressure. Again, we begin the simulation with the starting configuration from the NVT simulation. At this pressure the number of crystal layers on the right wall drops to one after 1×10^7 collisions, but a second layer of crystal quickly reformed there indicating a pressure fluctuation caused the event. After 1.8×10^7 collisions the right wall completely melts while still retaining crystal wetting on the other wall, as shown in figure 13. Once the right wall became wet by fluid, the system equilibrated and remained unchanged after 20×10^7 more collisions. The simulation was then discontinued.

The last pressure used was 11.2 $\beta p\sigma^3$. This series of simulation was started from two different configurations. The first was the beginning NVT system used in the other runs. It ran for 1.8×10^7 collisions and maintained crystal wetting upon both walls, figure 14. The second system was taken from the this study at a pressure of 11.3 $\beta p\sigma^3$ after 1.4×10^7 collisions. However, this configuration completely melted after only 4×10^6 collisions, figure 15. This demonstrates the large effect the starting configuration has on the outcome of the simulation, due to the metastable nature of prefreezing

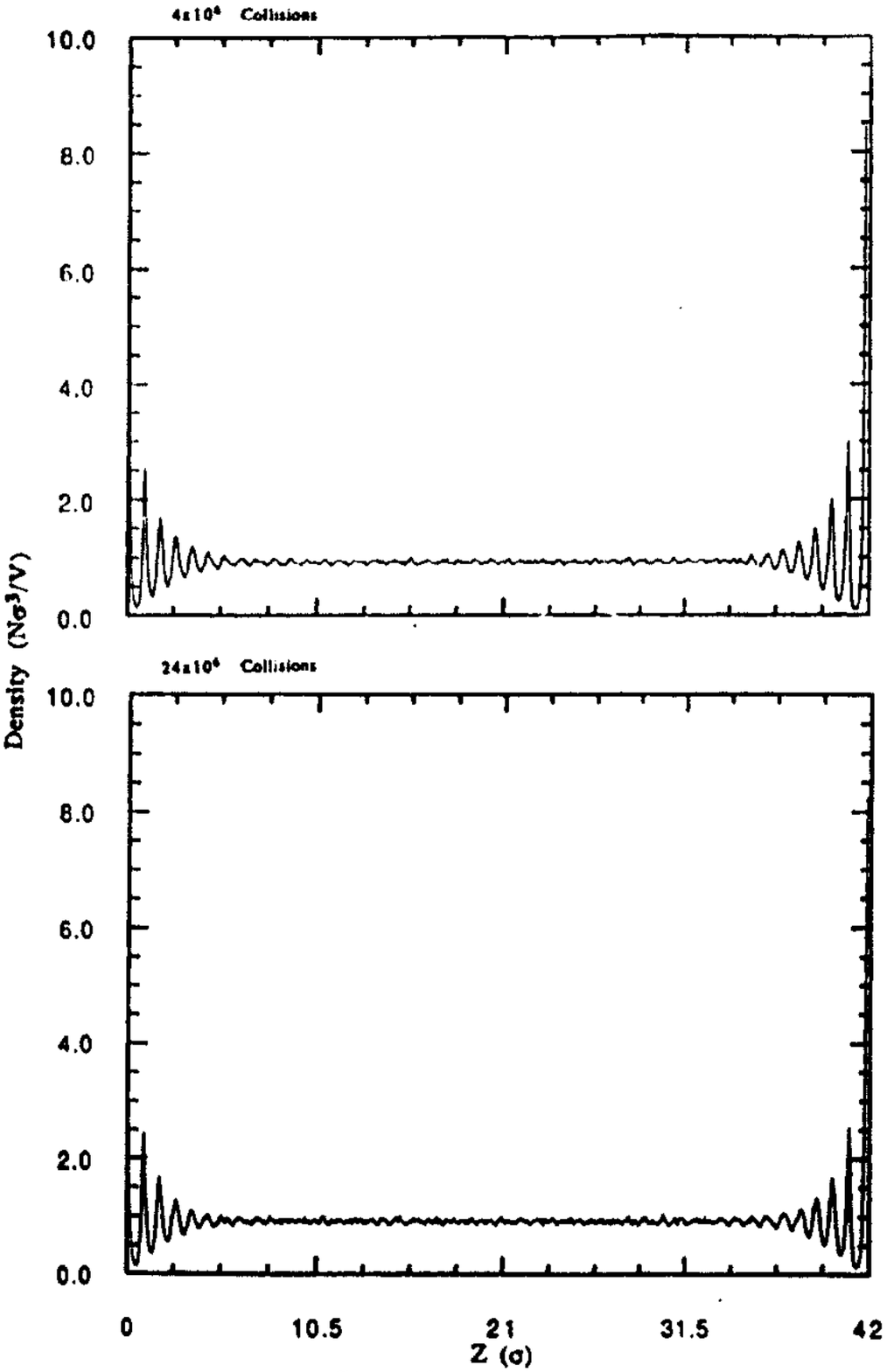


figure 12 - Density Profiles
NPT System at 11.0 $\beta p\sigma^3$

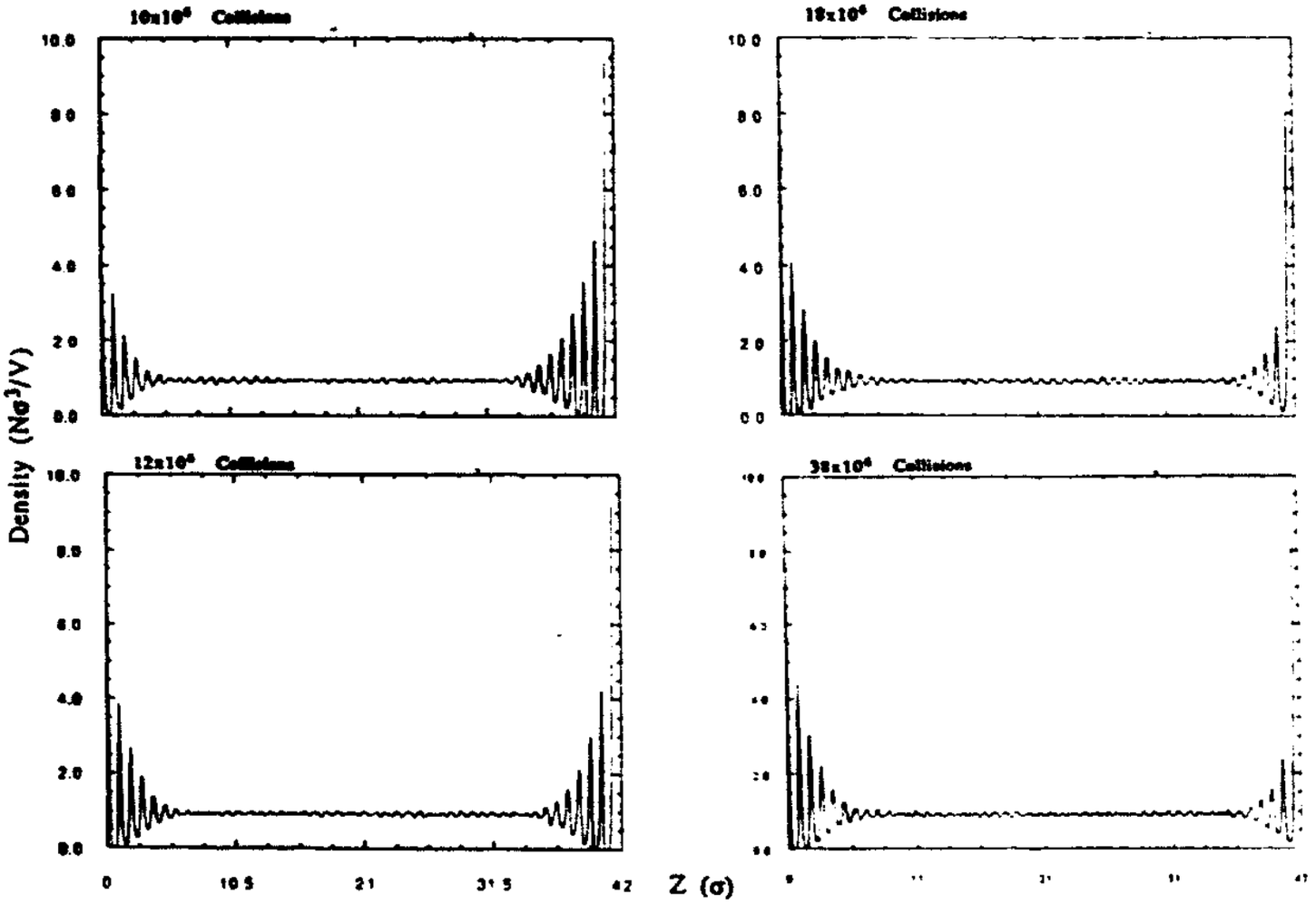


figure 13 - Density Profiles
NPT System at 11.1 $\beta p\sigma^3$

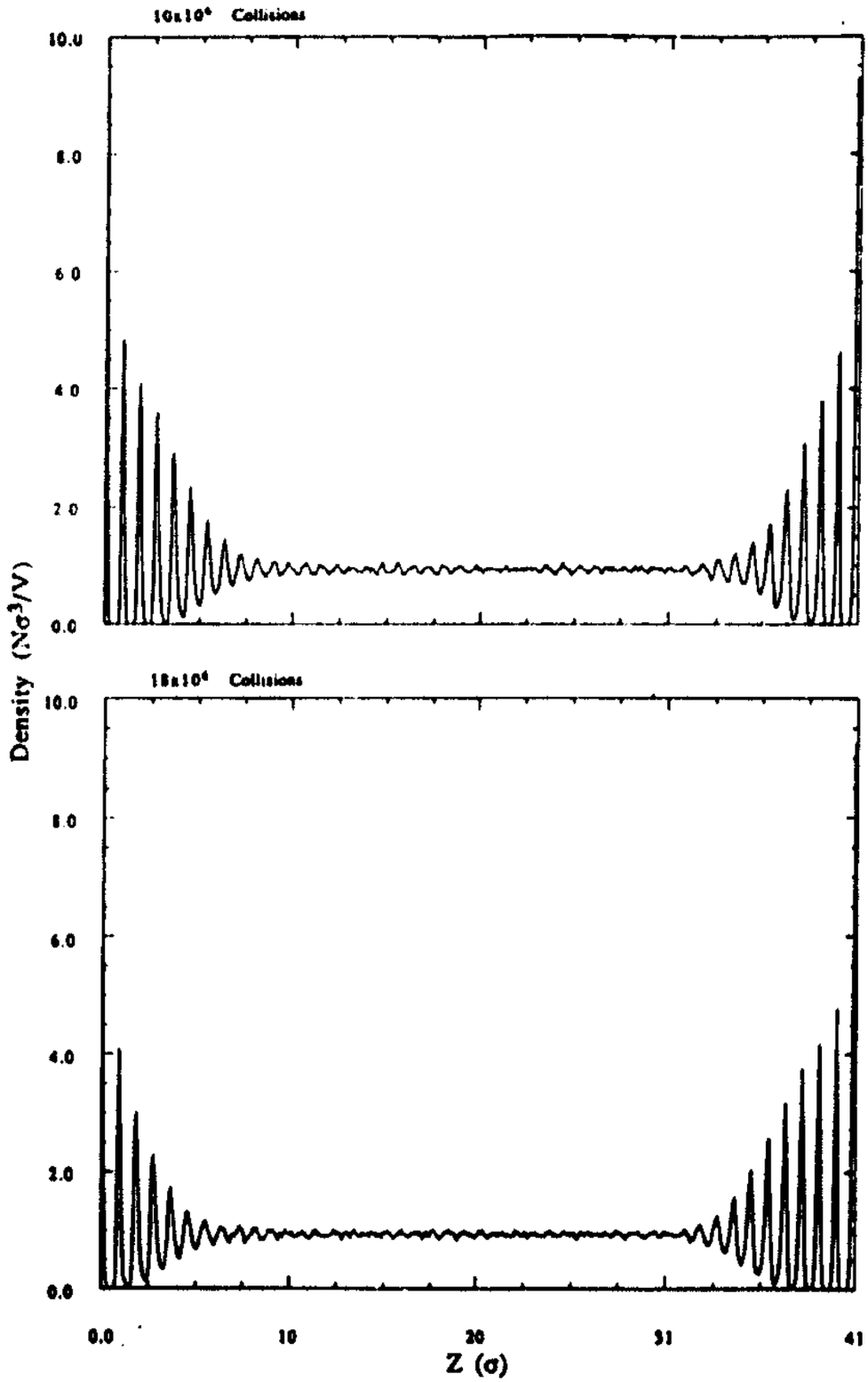


figure 14 - Density Profiles
NPT System at 11.2 $\beta p\sigma^3$, Original Starting Configuration

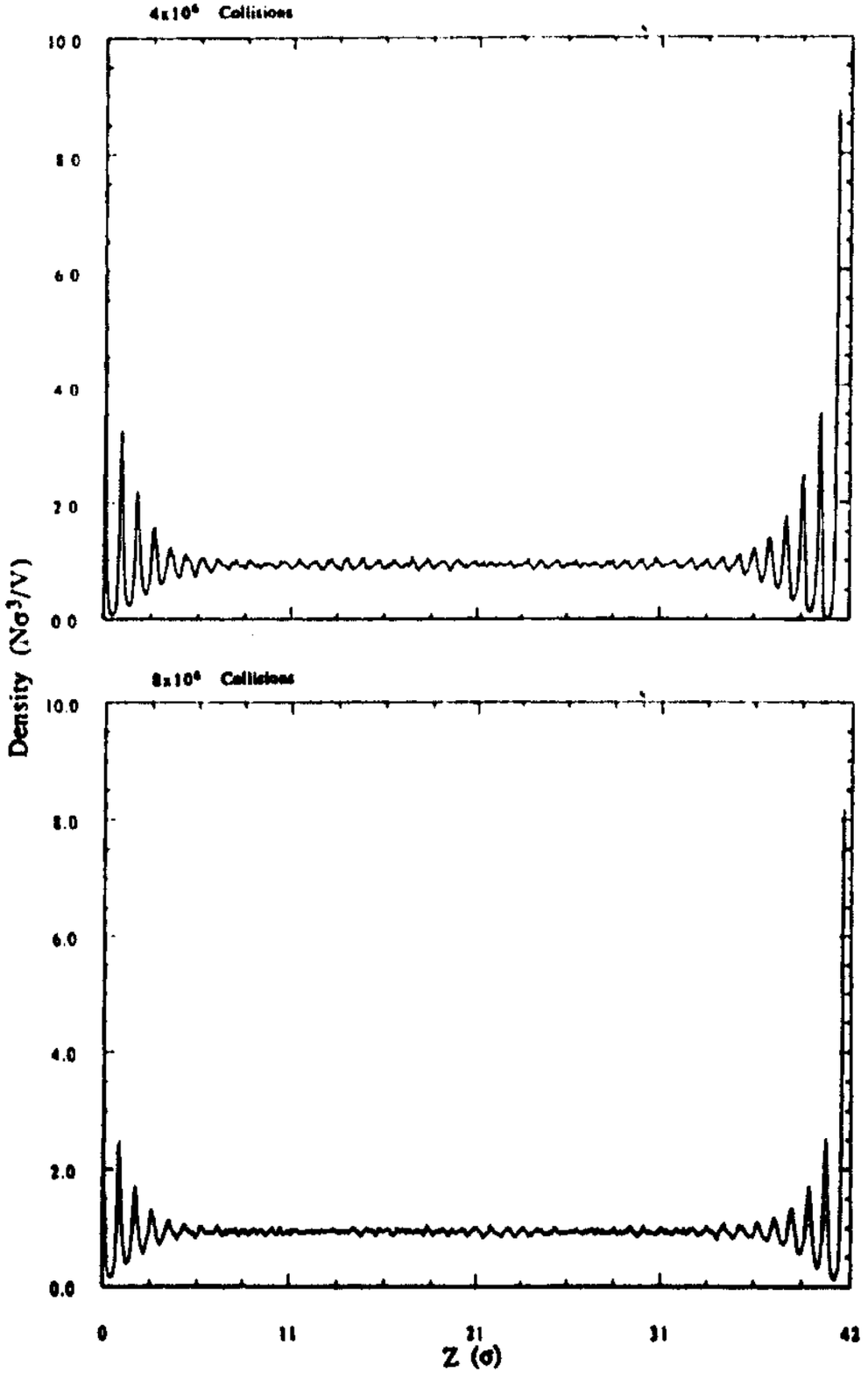


figure 15 - Density Profiles
NPT System at 11.2 $\beta p\sigma^3$, from 11.3 $\beta p\sigma^3$ NPT System

DISCUSSION

The results from this study have so far proved inconclusive. We have been unable to locate the exact range of prefreezing from the pressures selected. In addition, various fluctuations inherent in the system may disrupt the stability of the any prefreezing we witness, as demonstrated by the results generated from the 11.1 $\beta p\sigma^3$ trial. More trials near 11.1 $\beta p\sigma^3$ are needed before the stability of the system can be determined.

These studies seem to be greatly affected by the starting configuration. This is understandable due to the instability of systems within the prefreezing region. Single layer melting and freezing events can cause fluctuations large enough to destroy coexistence.

Study of Spontaneous Segregation

The hard sphere equations of state demonstrate a range of densities for a single melting transition. In practice the system makes sharp transitions from liquid to solid, however hysteresis has been observed in both melting and freezing transitions. In this study We wish to put a system in an unfavorable state by placing it on the tie bar at a density in between the two coexistence densities. We have then tracked the progress of the system to watch the formation of two defined coexistence phases from the ordered fluid.

METHODOLOGY

We again used PBC conditions with a canonical ensemble to simulate bulk crystal. The crystal was stretched as in the wetting experiments, however the entire crystal was stretched as opposed to just a small region.

This effectively lowered the density from $\rho^* = 1.0348$ to 0.99500, which is within coexistence values. The new density is constant throughout the system before the simulation begins.

During the simulation pressure and density profiles were recorded and examined for the segregation of this unstable density into to the two distinct phases of coexisting crystal and fluid.

RESULTS

The ordered fluid produced in the stretching of the bulk crystal, begins the simulation at a constant density $\rho_c^* = 0.99500$. Once the simulations started the system begins to experience density fluctuations which propagated through the system and began segregating the two phases. After 1.3×10^7 collisions, the two distinct phases are noticeably apparent.

As the simulation continued to run the system reverted back to its original configuration. At 35×10^7 collisions, the density profile is almost identical to the original profile. Currently the system is still under investigation but appears to be resegregating into the separate phases . All the density profiles from this study are shown in figure 16.

DISCUSSION

The second law of thermodynamics states a system will spontaneously move towards maximum entropy. The ordered fluid in our system should therefore try to increase entropy by increasing the randomness of the system. By segregating into a coexisting crystal and fluid system that system trades off the increases in entropy caused by the loss in order of the fluid region with the loss of entropy caused by crystalizing the other region.

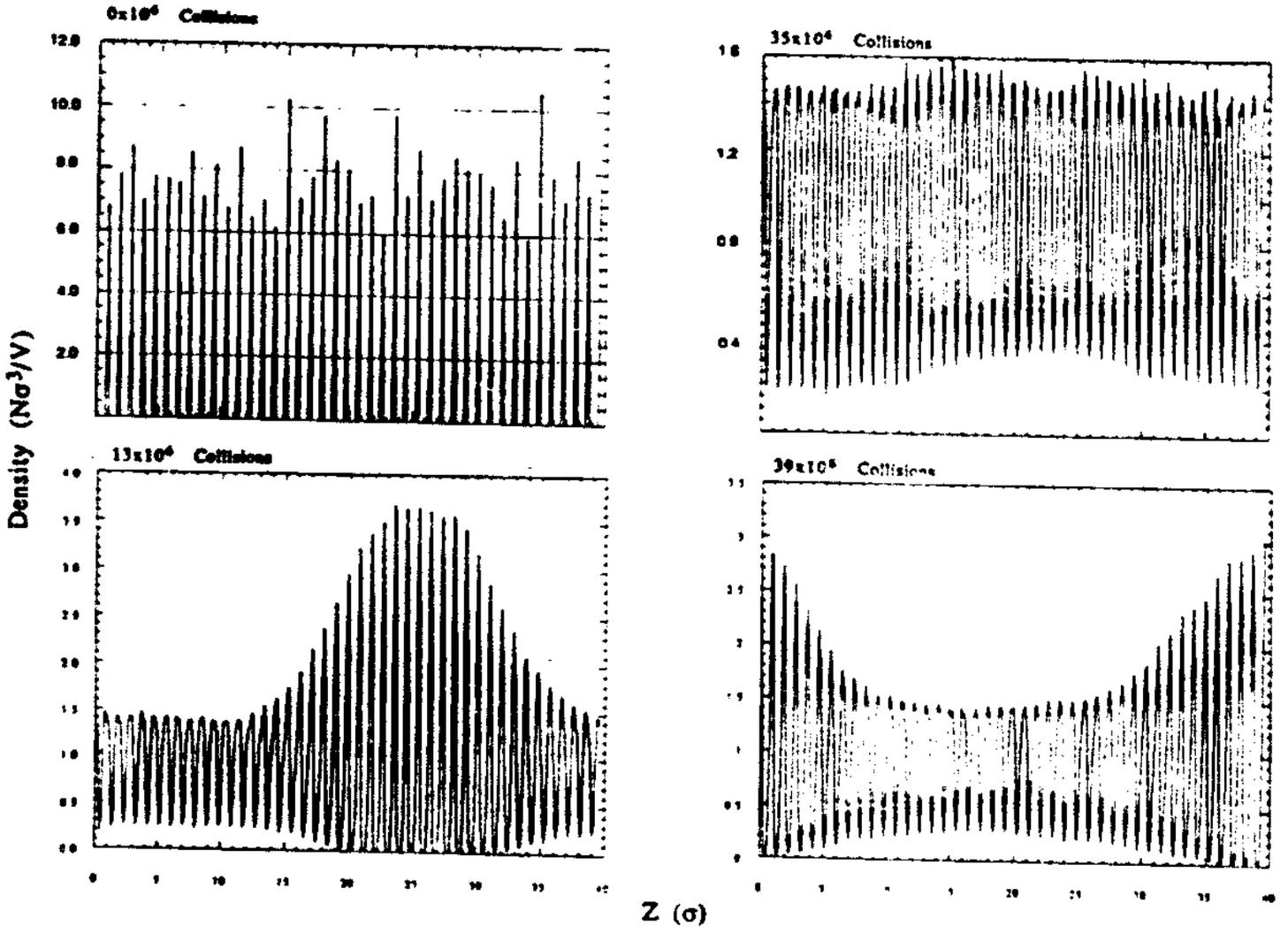


figure 16 - Density Profiles
Spontaneous Segregation Simulations

Because this system begins in the excluded region this tradeoff results in a net increase of entropy.

The simulations reveal the driving force for the spontaneous change is smaller than predicted, allowing the observed metastablity of the stretched to fluid to exist. The fluctuation back and forth between one and two phases will eventually equilibrate at two, but the number of collisions necessary will be much larger than $4x10^7$ collisions already run. An exact number of collisions is very difficult to estimate. Further study will be required to confirm the above conclusions

Recommendations

There are several different directions these studies could go. The wetting and coexistence density investigations each need lengthier runs to confirm the conclusions already drawn. The prefreezing investigation should be expanded to examine the entire hysteresis loop about the melting transition. The stability of the region can be examined by changing the pressure subtly and watch the system adjust to the new conditions. We would also like to define the rest of loop by setting the pressure above 11.4 $\beta p\sigma^3$ attempt to compress the system to crystal. Finally the spontaneous segregation study should be done at several other densities within the excluded region for thoroughness.

New areas that are of interest as a result of these studies. An important follow up study is a comparison of box dimensions with the disparity between the normal and transverse components of the pressure.

In addition we would like to investigate the effect of non isotropic crystal structure on system pressure.

Appendix

Simulation Fortran Code

```
      PROGRAM HSPHB3G
      IMPLICIT REAL*8(A-H,O-Z)
C
C     this program was adapted from hs111 for
C     the difference lies in the option of stretching the box
C     in the z direction  see ZCfact. In addition this program
C     calculates the transverse and normal pressure tensor
C     components
C
C         ***** VAX version ******* of HS111FOR
C               (see the OPEN statements)
C     1)this version contained calls to nag routines to start initial
C      maxwellian velocities
C      these calls have been replaced by calls to "GAUSS
C     2)Three GOTOs have been replaced by if statements that use the
C      variable JUMP in order to get rid of
C         no path to this statement messages
C     3)Switched to double precision
C     4)Disabled the (mixed) output of accumulators etc
C    5)writes out RWL,RWR,ZDIM
C
C                             Urbana   21/12/88
C
C         PROGRAM HSPHB3(tape7,tape8,tape9,TAPE10,input,output
C     *                TAPE1=input,tape2=output)
      DIMENSION RX(1200),RY(1200),RZ(1200)
      DIMENSION VX(1200),VY(1200),VZ(1200)
      DIMENSION DISPX(1200),DISPY(1200),DISPZ(1200)
      DIMENSION NR(1200),TC(1200)
      DIMENSION LINK(1200),K(1200),LMAX(30)
      DIMENSION LO(770),LR(270)
      DIMENSION XA(6),YA(6),ZA(6)
      DIMENSION DEN(10),TEJ(10),PXX(10),PYY(10),PZZ(10)
      DIMENSION ZPOS(10),XREV(10),YREV(10),ZREV(10)
        DATA NHIST/10/,PI/3.141592654/,MRAND/1/
         DATA FL1/7.8116/, FL2/27.9707/, CR1/13.3554/, CR2/21.3719/
      DATA JUMP/1/
C
C     OPEN(UNIT=1,FILE='INPUT.DAT',STATUS='OLD')
C     OPEN(UNIT=2,FILE='OUT.LIS',STATUS='NEW')
C     OPEN(UNIT=7,FILE='OLD.DAT',STATUS='OLD')
C     OPEN(UNIT=8,FILE='NEW.DAT',STATUS='NEW')
C
C
C   DATE 9/9/1981
C THIS IS A BULK VERSION
C  INSERTED GOTO'S 500,501,99,19,37,18
C ICON=0 SO NO CONFIGURATIONS WRITTEN TO TAPE
C
C
C THIS IS A GENERAL HARD SPHERE PROGRAM, USING THE NEW LINK CELL
C METHOD.THE PROGRAM CAN BE USED TO STUDY FLUID WALL INTERFACES
C THE WIDTH OF A LINK CELL SHOULD BE AT LEAST EQUAL TO 1.384 SIGMA
C CALCULATES DIFFERENT COMPONENTS OF PRESSURE TENSOR
C THIS VERSION WRITES CONFIGURATIONS TO TAPE10 EVERY IWRITE
C COLLISIONS (IWRITE=100 CORRESPONDS TO 0.5 MEAN COLLISION TIMES)
C ICON IS THE NUMBER OF CONFIGURATIONS PRESENT ON TAPE
C NCONF IS THE NUMBER OF CONFIGURATIONS WRITTEN IN THIS RUN
C READ INPUT PARAMETERS FROM TAPE 1=INPUT
      READ(1,*)N
      READ(1,*)RHO, zcfact
      READ(1,*)NTOT,NTAU,IWRITE
      READ(1,*)IN,ICON
      READ(1,*)NLX, NLY, NLZ
      WRITE(2,207)NLX, NLY, NLZ
207   FORMAT(1H ,5H NLX=,I3,5H NLY=,I3,5H NLZ=,I3,/)
      WRITE(2,209)N,RHO,IN,NTOT,NTAU,IWRITE,ICON
        write(2,*)' zcfact=',zcfact
209   FORMAT(/,3H N=,I6,4X,4HRHO=,F8.6,4X,4HIN=,I3,4X,5HNTOT=,I6,4X,
     1 5HNTAU=,I6,7HIWRITE=,I6,5HICON=,I6,/)
      READ(1,*)NAC,NCX,NCY,NCZ
131   FORMAT(4I6)
      WRITE(2,131)NAC,NCX,NCY,NCZ
      XNCX=NCX
      XNCY=NCY
      XNCZ=NCZ
      READ(1,*)(XA(I),YA(I),ZA(I),I=1,NAC)
132   FORMAT(3(2X,F8.6))
      WRITE(2,133)(XA(I),YA(I),ZA(I),I=1,NAC)
133   FORMAT(1H ,3(2X,F8.6))

C READ ICON CONFIGURATIONS FROM TAPE 10
      TOTIME=0.0
      IF(ICON.EQ.0)GOTO 78
      ITOT=ICON
      TIME=0.0
76    DO 74 II=1,ITOT
      ICC=II
      READ(10)ICN,TIME,RX,RY,RZ,DISPX,DISPY,DISPZ
C the next statement has been commented out. Probably not standard
C FORTRAN.  12/20/88
C          IF(EOF(10).NE.0.0)GOTO 75
74    CONTINUE
      GOTO 77
75    ITOT=ICC-1
      REWIND 10
      GOTO 76
77    ICON=ITOT
      TOTIME=TIME
      WRITE(2,255)ICON,TOTIME
C
C
C SET ACCUMULATORS TO ZERO
78    NAV = 0
      NCONF=0
      XTIME=0.0
      DO 25 J=1,NHIST
      ZPOS(J)=0.0
      XREV(J)=0.0
      YREV(J)=0.0
      ZREV(J)=0.0
      DEN(J)=0.0
      TEJ(J)=0.0
      PXX(J)=0.0
      PYY(J)=0.0
      PZZ(J)=0.0
25    CONTINUE

C CONVERSION TO PROGRAM UNITS
      VOL=SQRT(2.0)/RHO
      T=0.0D0
      XN=N
C         ...    RHOSOL=1.040D
      KHIST = NHIST
      XC=1.0
      YC=DSQRT(3.0D0)
C              ZC=2.0*DSQRT(6.0D0)*(4.0/3.0)*(RHOSOL/RHO)
      ZC=DSQRT(6.0D0)*(XNCZ/XNCX)*ZCfact
      YC2=0.5D0*YC
      ZC2=0.5D0*ZC
C              XL=(XNAC/(RHO*YC*ZC))**(1.0D0/3.0D0)
         XL=(dsqrt(2.0d0)/rho)**(1.0d0/3.0d0)*NCX
      YL=XL*YC
      ZL=XL*ZC
      FL1=FL1/XL
      FL2=FL2/XL
```

```
      CR1=CR1/XL
      CR2=CR2/XL
      XINT1=FL1
      XINT2=CR1
      XINT3=CR2
      XINT4=FL2
      S=1.0D0/XL
      SS=S*S
       WRITE(2,210)S
      NWR(1)=NWRIT
      NRUN=NTAU
      AREA=XL*YL
  210 FORMAT(//,22H SIGMA IN UNITS OF L =,1PE9.2,/)
      WRITE(2,88) XC,YC,ZC
      WRITE(2,89) XL,YL,ZL
      WRITE(2,*) AREA=,AREA
   88 FORMAT(1H ,35H BOX DIMENSIONS IN PROPER UNITS XC=,1P10.6
     +    4H YC=,1P10.6,4H ZC=,1P10.6,/)
   89 FORMAT(1H ,35H BOX DIMENSIONS IN UNITS OF SIGMA   XL=,1P10.6,
     +    4H YL=,1P10.6,4H ZL=,1P10.6,/)
C INITIALIZE VARIABLES
      I=1
      DO 1 IZ=1,NCZ
      DO 1 IY=1,NCY
      DO 1 IX=1,NCX
      RX(I)=-.5D0+XA(I)/XNCX
      RY(I)=(IY-1)+YA(I)*YC/YNCY
      RZ(I)=(IZ-1)+ZA(I)*ZC/ZNCZ
       RX(I)=RX(I)+0.001D0
       RY(I)=RY(I)+0.001D0
       RZ(I)=RZ(I)+0.001D0
      IF(ICON.EQ.0) DISPX(I)=0.0
      IF(ICON.EQ.0) DISPY(I)=0.0
      IF(ICON.EQ.0) DISPZ(I)=0.0
      I=I+1
    1 CONTINUE
      IF(IN.EQ.0)GO TO 71
C READ IN CONFIGURATION FROM TAPE7
      PX=0.0
      PY=0.0
      PZ=0.0
       XGAUSS=1234.0
      READ(7,*) RWI,RWR,IDIM
      DO 72 I=1,N
       READ(7,*)RX(I),RY(I),RZ(I),VX(I),VY(I),VZ(I),
     +       DISPX(I),DISPY(I),DISPZ(I)
      IF(MRAND.EQ.1) CALL GAUSS(XGAUSS,1.0,0.0,VX(I))
      IF(MRAND.EQ.1) CALL GAUSS(XGAUSS,1.0,0.0,VY(I))
      IF(MRAND.EQ.1) CALL GAUSS(XGAUSS,1.0,0.0,VZ(I))
      PX=PX+VX(I)
      PY=PY+VY(I)
      PZ=PZ+VZ(I)
   72 CONTINUE
C READ IN ACCUMULATORS FROM TAPE 7
      IF(ICON.EQ.0) GOTO 10
      READ(7,*) NAV, XTIME
      DO 70 I=1,NHIST
      READ(7) DEN(I),TEMP(I),PXX(I),PYY(I),PZZ(I)
      READ(7) ZROS(I),XREV(I),YREV(I),ZREV(I)
   70 CONTINUE
      GOTO 10
   71 CONTINUE
C INITIAL MAXWELLIAN VELOCITIES
      PX=0.0
      PY=0.0
      PZ=0.0
       XGAUSS=1234.0
      DO 5 I=1,N
        NRD=1
        CALL GAUSS(XGAUSS,1.0,0.0,VX(I))
        CALL GAUSS(XGAUSS,1.0,0.0,VY(I))
        CALL GAUSS(XGAUSS,1.0,0.0,VZ(I))
      PX=PX+VX(I)
      PY=PY+VY(I)
      PZ=PZ+VZ(I)
    5 CONTINUE
C CHAIN LINK TABLES
   10 XNC=NCX
      XL=1.0
      YNC=NCY
      YL=YC
      ZNC=NCZ
      ZL=ZC
      NC2=NCX*NCY
      NCT=NCX*NCY*NCZ
      DO 320 I=1,NCT
      IHEAD(I)=0
      IHEAD1(I)=0
  320 CONTINUE
      DO 321 I=1,N
      IX=INT(RX(I)*XNC/XL+1.0)
      IY=INT(RY(I)*YNC/YL)
      IZ=INT(RZ(I)*ZNC/ZL)
      IP=IX+NCX*(IY+NC2*IZ)
      IF(IP.GT.NCT.OR.IP.LT.1)GO TO 26
      IC(I)=IP
      ME=IHEAD(IP)
      IF(ME.EQ.0)GO TO 22
      LINK(ME)=I
      IHEAD(IP)=I
      GOTO 321
   22 CONTINUE
      IHEAD(IP)=I
      IHEAD1(IP)=I
  321 CONTINUE
C
C CHECK CELL CHAIN
C
      DO 322 IP=1,NCT
      ME=IHEAD(IP)
      IF(ME.EQ.0)GOTO 322
      LINK(ME)=1.0D0
  322 CONTINUE
      GOTO 329
   26 CONTINUE
      WRITE (2,270) IX, IY, IZ, IP, I
      WRITE (2,211) RX(I), RY(I), RZ(I)
      GOTO 24
  329 CONTINUE
C
C CHECK ON LINK ARRAYS
      WRITE(2,200)
      DO 330 IP=1,NCT
      I=1
      ME=IHEAD(IP)
   31 LMAX(I)=ME
      IF(ME.EQ.0) GOTO 341
      I=I+1
      ME=LINK(ME)
      IF(ME.NE.IHEAD(IP)) GOTO 31
      I=I-1
      IF(IP.GT.50)GOTO 330
  341 WRITE (2,220) IP, (LMAX(KK),KK=1,I)
  330 CONTINUE
C ZERO MOMENTUM
```

```
      TEMP=0.0
      DO 6 I=1,N
       VX(I)=VX(I)-PX/XN
       VY(I)=VY(I)-PY/XN
       VZ(I)=VZ(I)-PZ/XN
      TEMP=TEMP+VX(I)*VX(I)+VY(I)*VY(I)+VZ(I)*VZ(I)
    6 CONTINUE
C SET INITIAL TEMP
      C=SQRT(XN*3.0/TEMP)
      TEMP=0.0
      PX=0.0
      PY=0.0
      PZ=0.0
      DO 16 I=1,N
      VX(I)=VX(I)*C
      VY(I)=VY(I)*C
      VZ(I)=VZ(I)*C
      PX=PX+VX(I)
      PY=PY+VY(I)
      PZ=PZ+VZ(I)
      TEMP=TEMP+VX(I)*VX(I)+VY(I)*VY(I)+VZ(I)*VZ(I)
   16 CONTINUE
      PX=PX/XN
      PY=PY/XN
      PZ=PZ/XN
      TEMP=TEMP/XN*3.0
      WRITE(2,205)TEMP,PX,PY,PZ
  205 FORMAT(/,10H INITIAL TEMPER=,4X,15HINITIAL MOMENTA=,3(2X,E13.6))
C PRINT OUT INITIAL POSITIONS AND DIRECTIONS AND LINK TABLES
      WRITE(2,203)
  203 FORMAT(/,5X,1HI,7X,2HRX,9X,2HRY,9X,2HRZ,12X,2HVX,9X,2HVY,9X,2HV
     1Z,7HLINK(I),4X,2HIC,/)
      DO 14 I=1,20
       WRITE(2,204)I,RX(I),RY(I),RZ(I),VX(I),VY(I),VZ(I),LINK(I),IC(I)
  204    FORMAT(I6,2(3X,F7.3,3(3X,F9.6)),4X,I5,2X,I5)
   14 CONTINUE
      NCOUNT=0
      NCYC=0
      PTOT=0.0
      NCT=0
      P=0.0
      TAU=0.0
      PNORM=0.0
      PTRANS=0.0
C COLLISION PREDICTION TABLES
   29 I=1
      DT=1.000
    9 CONTINUE
C SPHERE COLLISION
      IF(I.EQ.K)GO TO 21
      IF(I.EQ.L)GO TO 21
      IF(NK(I).EQ.K)GO TO 21
      IF(NK(I).EQ.L)GO TO 21
      IF(NCT.EQ.0)GO TO 21
      GO TO 23
   21 CONTINUE
      TC(I)=99.000
      IP=IC(I)
      IPM1=IP-1
C   CALCULATE IX,IY,IZ FROM IP
      IZ=IPM1/NL2
      IY=(IPM1-IZ*NL2)/NLX
      IX=IP-IY*NLX-IZ*NL2
      IY=IY+1
      IZ=IZ+1

C   CALCULATE IP
        DO 40 KZ=1,3
        DO 40 KY=1,3
        DO 40 KX=1,3
          CX=0.0
          CY=0.0
          CZ=0.0
          JX=IX+KX-2
          JY=IY+KY-2
          JZ=IZ+KZ-2
C               PERIODIC BOUNDARY CONDITIONS
        IF(JX-NLX)371,370,371
  370     JX=1
        CX=XL
  371     IF(JY-NLY)373,372,373
  372     JY=1
        CY=YL
  373     IF(JZ-NLZ)375,374,375
  374     JZ=1
        CZ=ZL
  375     IF(JX)377,376,377
  376     JX=NLX
        CX=-XL
  377     IF(JY)379,378,379
  378     JY=NLY
        CY=-YL
  379     IF(JZ)382,380,382
  380     JZ=NLZ
        CZ=-ZL
C
  382   IP=JX+(JY-1)*NLX+(JZ-1)*NL2
C
        I=LCP(IP)
C               SKIP IF EMPTY
      IF(I.EQ.0)GO TO 40
   41 CONTINUE
      IF(I.EQ.I)GO TO 7
C
      X=RX(I)-RX(J)-CX
      Y=RY(I)-RY(J)-CY
      Z=RZ(I)-RZ(J)-CZ
      U=VX(I)-VX(J)
      V=VY(I)-VY(J)
      W=VZ(I)-VZ(J)
      B=X*U+Y*V+Z*W
      IF(B.GE.0.0D0)GO TO 7
      VV=U*U+V*V+W*W
      RR=X*X+Y*Y+Z*Z
      C=VV*(RR-SS)
      BB=B*B
      IF(C.GE.BB)GO TO 7
      Q=SQRT(BB-C)
      D=-(B+Q)/VV
      IF(C.LT.0.0D0)GO TO 17
      IF(D.GT.TC(I))GO TO 7
      TC(I)=D
      NK(I)=J
    7 CONTINUE
      I=LINK(J)
      IF(I.NE.LCP(IP))GO TO 41
   40 CONTINUE
   23 CONTINUE
      IF(TC(I).GT.DT)GO TO 20
      DT=TC(I)
      KS=I
      LS=NK(I)
   20 I=I+1
      IF(I.LE.N)GO TO 9
      K=KS
      L=LS
```

```
      RY=Y*Y*RAS
      RZ=Z*Z*RAS
      PTOT=PTOT+R
      P =P + R
      PNORM =PNORM +R/
      PTRANS=PTRANS+RX +RY
C SKIP THIS PART FOR A BULK FLUID CALCULATION >>>>>>> GOTO 99
      IF(JUMP.EQ.1) GOTO 99
C                          <<<<<<<<
C CALCULATE BINS FOR PRESSURE TENSOR
      KKK=INT(R/(K)*(NHIST/2.))+1
      KKL=INT(R/(L)*(NHIST/2.))+1
      KMAX=MAX0(KKK,KKL)
      KMIN=MIN0(KKK,KKL)
      NBINS=KMAX-KMIN+1
      NHIST2=NHIST/2
      IF(NBINS.LT.NHIST2) GOTO 96
      NBINS=NHIST-KMIN+KMAX+1
      DO 97 KK=1,KMIN
      PXX(KK)=PXX(KK)+RXFLOAT(NBINS)
      PYY(KK)=PYY(KK)+RYFLOAT(NBINS)
      PZZ(KK)=PZZ(KK)+RZFLOAT(NBINS)
97    CONTINUE
      DO 98 KK=KMAX,NHIST
      PXX(KK)=PXX(KK)+RXFLOAT(NBINS)
      PYY(KK)=PYY(KK)+RYFLOAT(NBINS)
      PZZ(KK)=PZZ(KK)+RZFLOAT(NBINS)
98    CONTINUE
      GOTO 99
96    DO 95 KK=KMIN,KMAX
      PXX(KK)=PXX(KK)+RXFLOAT(NBINS)
      PYY(KK)=PYY(KK)+RYFLOAT(NBINS)
      PZZ(KK)=PZZ(KK)+RZFLOAT(NBINS)
95    CONTINUE
99    R=RAS
      X=X*R
      Y=Y*R
      Z=Z*R
C                          <<<<<<<<
      IF(JUMP.EQ.1) GOTO 500
C                          <<<<<<<<
C CALCULATE REVERSAL FREQUENCIES
      VXK=VX(K)
      VYK=VY(K)
      VZK=VZ(K)
      VXL=VX(L)
      VYL=VY(L)
      VZL=VZ(L)
C                          <<<<<<<<<
500   CONTINUE
C                          <<<<<<<<<
      VX(K)=VX(K)+X
      VY(K)=VY(K)+Y
      VZ(K)=VZ(K)+Z
      VX(L)=VX(L)+X
      VY(L)=VY(L)+Y
      VZ(L)=VZ(L)+Z
C                          <<<<<<<<<
      IF(JUMP.EQ.1) GOTO 501
C                          <<<<<<<<<
      REV=VX(K)*VXK
      IF(REV.LT.0.0) XREV(KKK)=XREV(KKK)+1
      REV=VY(K)*VYK
      IF(REV.LT.0.0) YREV(KKK)=YREV(KKK)+1
      REV=VZ(K)*VZK
      IF(REV.LT.0.0) ZREV(KKK)=ZREV(KKK)+1
      REV=VX(L)*VXL
      IF(REV.LT.0.0) XREV(KKL)=XREV(KKL)+1
C UPDATE POSITIONS TO NEXT COLLISION
      IF(NCOUNT.GE.NTOT)DT=0.5*DT
      DO 8 I=1,N
      X=RX(I)
      Y=RY(I)
      Z=RZ(I)
      TC(I)=TC(I)-DT
      X=X+VX(I)*DT
      Y=Y+VY(I)*DT
      Z=Z+VZ(I)*DT
      DISPX(I)=DISPX(I)+VX(I)*DT
      DISPY(I)=DISPY(I)+VY(I)*DT
      DISPZ(I)=DISPZ(I)+VZ(I)*DT
C PERIODIC BOUNDARY CONDITIONS
      IF(X.GE.1.000)X=X-1.000
      IF(X.LT.0.000)X=X+1.000
      IF(Y.GE.Y(C))Y=Y-YC
      IF(Z.GE.ZC)Z=Z-ZC
      IF(X.LT.0.000)X=X+1.000
      IF(Y.LT.0.000)Y=Y+YC
      IF(Z.LT.0.000)Z=Z+ZC
      RX(I)=X
      RY(I)=Y
      RZ(I)=Z
C CALCULATE LINK(I,J)
      IX = INT(RX(I) * XNC/DC + 1.0)
      IY = INT(RY(I) * YNC/YC)
      IZ = INT(RZ(I) * ZNC/ZC)
      IC = IX + NCX*IY + NCX*NCY*IZ
      IF (IP.EQ.IC(I)) GOTO 34
C
C TAKE OUT OF OLD LINK(I,J)
C
      ME = IC(I)
      J = LO(ME)
329   IF (LINK(J).EQ.I) GO TO 327
      J = LINK(J)
      GOTO 328
327   LINK(J) = LINK(I)
      LO(ME) = LINK(I)
      IF (LINK(I).LT.0) LO(ME)=0
C
C PUT I INTO NEW LINK(I,J)
C
      J = LO(IP)
      IF (LO(IP).EQ.0) J=I
      LINK(I) = LINK(J)
      LINK(J) = I
      IC(I) = IP
      LO(IP) = I
34    CONTINUE
8     CONTINUE
      NCT=I
      IF(NCOUNT.GE.NTOT)GO TO 85
C NEW VELOCITIES OF COLLIDING PAIR
      X=RX(K)-RX(L)
      Y=RY(K)-RY(L)
      Z=RZ(K)-RZ(L)
      IF(X.GT.0.5D0)X=X-1.0D0
      IF(Y.GT.YC2) Y=Y-YC
      IF(Z.GT.ZC2) Z=Z-ZC
      IF(X.LT.-0.5D0)X=X+1.0D0
      IF(Y.LT.-YC2) Y=Y+YC
      IF(Z.LT.-ZC2) Z=Z+ZC
      U=VX(K)-VX(L)
      V=VY(K)-VY(L)
      W=VZ(K)-VZ(L)
      R=X*U+Y*V+Z*W
      RX=X*X+RAS
```

```
      REV=VY(L)*VYL
      IF(REV.LT.0.0) YREV(KKL)=YREV(KKL)+1
      REV=VZ(L)*VZL
      IF(REV.LT.0.0) ZREV(KKL)=ZREV(KKL)+1
C CONTINUE FROM HERE FOR A BULK FLUID CALCULATION
C                 <<<<<<<
501   CONTINUE
C                 <<<<<<<
      NCOL=NCOL+1
      T=T+DT
      TAU=TAU+DT
      NCOUNT=NCOL
       NAV=NAV+1
C                 <<<<<<<
      IF(JUMP.EQ.1) GOTO 19
C                 <<<<<<< SKIP FOR BULK CALCULATION
       DO 11 I=1,N
       KK=INT(RZ(I)*XHIST/ZC))+1
      DEN(KK)=DEN(KK)+1.0
       EKI=VX(I)*VX(I) + VY(I)*VY(I) + VZ(I)*VZ(I)
      TEZ(KK)=TEZ(KK)+EKI
      ZPOS(KK)=ZPOS(KK) + RZ(I)
11    CONTINUE
      IF(NCOUNT.LT.NWRITE) GOTO 19
C WRITE CONFIGURATION TO TAPE 10
      TIME=TOTIME + TA
      KCON=KCON+1
      WRITE(10)KCON,TIME,RX,RY,RZ,DISPX,DISPY,DISPZ
      NWRITE=NWRITE+IWRITE
      NCONF=NCONF+1
C                 <<<<<<<<<<< CONTINUE FROM HERE FOR BULK CALC
19    IF(NCOUNT.LT.NRUN)GO TO 29
      TEMP=0.0
       PX=0.0
       PY=0.0
       PZ=0.0
       DO 27 I=1,N
       EKI = VX(I)**2 + VY(I)**2 + VZ(I)**2
      TEMP=TEMP + EKI
      PX=PX+VX(I)
      PY=PY+VY(I)
      PZ=PZ+VZ(I)
27    CONTINUE
      TEMP=TEMP/(XN*3.0)
      PX=PX/XN
      PY=PY/XN
      PZ=PZ/XN
C
C CALCULATE MEAN SQ DISPL FOR THE 3 REGIONS SEPERATELY
      DIFFL=0.0
      DIFCR=0.0
      DIFINT=0.0
      XNFL=0.0
      XNCR=0.0
      XINT=0.0
      DO 37 I=1,N
C*****IF(RZ(I).GE.FL1.AND.RZ(I).LT.FL2) GOTO 38
      XNFL=XNFL+1
      DIFFL=DIFFL+DISPX(I)**2+DISPY(I)**2+DISPZ(I)**2
C                 <<<<<<<
      IF(JUMP.EQ.1) GOTO 37
C                 <<<<<<<
38    IF(RZ(I).LE.CR1.OR.RZ(I).GE.CR2) GOTO 39
      XNCR=XNCR+1
      DIFCR=DIFCR+DISPX(I)**2+DISPY(I)**2+DISPZ(I)**2
39    IF(RZ(I).GT.XINT1.AND.RZ(I).LT.XINT2) GOTO 36
      IF(RZ(I).GT.XINT3.AND.RZ(I).LT.XINT4) GOTO 36
      GOTO 37
36    XNINT=XNINT+1
      DIFINT=DIFINT+DISPX(I)**2+DISPY(I)**2+DISPZ(I)**2
37    CONTINUE
      DIFFL=DIFFL/(XNFL*SS)
C*****DIFCR=DIFCR/(XNCR*SS)
C*****DIFINT=DIFINT/(XNINT*SS)
      WRITE(2,202)NCOL,K,L,T,TEMP,PX,PY,PZ
202   FORMAT(/,10H COLLISION,I6,4X,11H BETWEEN,2I5,4X,5HAT T=,F18.4,4X,5H
     1TEMP=,F8.4,4X,7HMOMENTA,3(2X,E13.6))
      AP=1.0+PTOT/(3.0*XN*T)
      AP=AP/VOL
       PSUBAV=1.0 + PA/(3.0*XN*TAU)
       PSUBAV=PSUBAV/VOL
       CP=1.0+FLOAT(NTAU)*SQRT(PI)*S/(TAU*(XN-1.0)*3.0)
      CP=CP/VOL
       APN=1.0+PNORM/(XN*T)
       APT=1.0+PTRANS/(2.0*XN*T)
       WRITE(7,2005) APN*XN*SAREA*Z), APT*XN*SAREA*Z)
       WRITE(6,2005) APN*XN*SAREA*Z), APT*XN*SAREA*Z)
      WRITE(2,251)PSUBAV, AP, CP, DIFFL, DIFCR, DIFINT
251   FORMAT(10H PRESSURE=,F13.6,3X,4H AP=,E13.6,6H CP=,E13.6
     +,3(F12.6))
      NRUN=NRUN+NTAU
       P=0.0
       TAU=0.0
      GO TO 29
C END OF THIS RUN
15    CONTINUE
       WRITE(2,234)T,NCOL
234   FORMAT(/,10H END OF RUN,3X,2HT=,F9.6,3X,5HNCOL=,I8/)
      PTOT=1.0+PTOT/(3.0*XN*T)
      PTOT=PTOT/VOL
       WRITE(2,250)PTOT
250   FORMAT(/,17H AVERAGE PRESSURE,2X,F10.6,/)
       CP=1.0+FLOAT(NCOL)*SQRT(PI)*S/(T*(XN-1.0)*3.0)
      CP=CP/VOL
       WRITE(2,253) CP
       APN=APN*XN*SAREA*Z)
       APT=APT*XN*SAREA*Z)
        WRITE(2,*) 'units of beta*sigma**3  PNORM = ',apn
        WRITE(2,*) 'units of beta*sigma**3  PTRANS= ',apt
C   overall density
      RHOTOT=XN/(XL*YL*Z)*XL*XL*XL)
       WRITE(2,*) 'overall density = ',RHOTOT
      REALT=T/S
       WRITE(2,820)REALT
820   FORMAT(/,28H TIME IN HARD SPHERE UNITS :,2X,F12.9,/)
2005  FORMAT(7H PNORM=,E15.8,2X,8H PTRANS=,E15.8)
253   FORMAT(/,29H PRESSURE FROM COLLISION RATE,2X,F10.6,/)
      XTIME=XTIME+T
       WRITE(2,254)NCONF
      WRITE(2,256)KCON,TIME,XTIME
       WRITE(2,201)T
201   FORMAT(/,32H POSITIONS AND VELOCITIES AT T= ,F8.4,/)
       DO 12 I=10,100,10
       WRITE(2,200)I,RX(I),RY(I),RZ(I),VX(I),VY(I),VZ(I),NRI),TC(I)
200     FORMAT(I6,2(3X,3(2X,F9.6)),4X,I5,3X,F8.5)
12    CONTINUE
C PUNCH OUT FINAL CONFIGURATION
       WRITE(2,252)
252   FORMAT(/,22H PUNCHED CONFIGURATION/)
C..........
       RWL=0.0
      RWR=ZC
      ZDIM=RWR-RWL
       WRITE(2,1966) RWL,RWR,ZDIM
1966   FORMAT(1X,' RWL=',F20.16,' RWR= ',F20.16,' ZDIM=',F20.16/)
C
```

```
      XPXX=PXX(L)/XTIME
      YPYY=PYY(L)/XTIME
      ZPZZ=PZZ(L)/XTIME
 41   DENTOT = DENTOT + DEN(L)/XNAV
      TE2TOT = TE2TOT + TE2(L)/XNAV * 3.0)
      ZPOST=ZPOST+ ZPOS(L)*XL/XNAV
      XRVTOT=XRVTOT+XRLV(L)/T
      YRVTOT=YRVTOT+YRLV(L)/T
      ZRVTOT=ZRVTOT+ZRLV(L)/T
      PXXTOT=PXXTOT+PXX(L)/XTIME
      PYYTOT=PYYTOT+PYY(L)/XTIME
      PZZTOT=PZZTOT+PZZ(L)/XTIME
      TENTOT=PZZTOT-0.5*(PXXTOT+PYYTOT)
      IF(IPRINT.EQ.1) GOTO 50
      WRITE(2,290) Z,D,ZP,DENTOT,TE2TOT,XRVTOT,YRVTOT,ZRVTOT
      WRITE(9,290) Z,D,ZP,DENTOT,TE2TOT,XRVTOT,YRVTOT,ZRVTOT
      GOTO 35
 50   WRITE(2,290)Z,DENTOT,TENTOT,PZZTOT,PXXTOT,PYYTOT,ZPOST,ZPZZ
      WRITE(9,290)Z,DENTOT,TENTOT,PZZTOT,PXXTOT,PYYTOT,ZPOST,ZPZZ
 35   CONTINUE
      IPRINT=IPRINT+1
      IF(IPRINT.EQ.1) GOTO 52
      GO TO 10
 17     WRITE(2,204)I,J
 204  FORMAT(/,23H CATASTROPHIC COLLISION 20 /)
      WRITE(2,233)I,J,D,X,Y,Z,U,V,W
       WRITE(2,236)RX(I),RY(I),RZ(I),IC(I),LINK(I)
      WRITE(2,236)RX(J),RY(J),RZ(J),IC(J),LINK(J)
 233     FORMAT(2I6,2X,G14.6,3(2X,F9.5),3X,3(2X,F9.5))
 236    FORMAT(3F10.7,2I6)
 10   CONTINUE
 206  FORMAT(1H1)
 211  FORMAT(1H ,20HRX(I), RY(I), Z(I) ARE RESP.,3(2X,F9.7))
 220   FORMAT(I6,2X,25I5)
 254  FORMAT(1H ,I10,24HCONFIGURATIONS TO TAPE 10,/)
 255  FORMAT(1H ,I10,19HCONFIGURATIONS READ,2X,7HTOTIME =,G12.6)
 256  FORMAT(1H ,I10,24HCONFIGURATIONS ON TAPE 10,2X,7HTOTIME = G12.6
     +6HXTIME=,G12.6,/)
 270  FORMAT(1H ,27HIX, IY, IZ, IP, L ARE RESP.,5(2X,I6))
 290    FORMAT(F8.4,8(1X,G14.8))
 291    FORMAT(3I8,F9.6)
 401    FORMAT(I20,G14.8)
 24   CONTINUE
      close(8)
      close(2)
      STOP
      END
C*************************************************************
      SUBROUTINE GAUSS(XX,S,AM,V)
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*4 Y
      SUM=0.0
      DO 1 I=1,12
       Y = RANF(XX)
       SUM=SUM+Y
 1    CONTINUE
      V=(SUM-6.0)*S+AM
      RETURN
      END

      real function ranf(seed)

C...............
C         random no. generator
C         Peter Manson
C...............

      double precision seed,d2p31m,d2p31

C WRITE OUT FINAL POSITIONS OF THE WALLS TO TAPE 8
      WRITE(8,*)RWL,RWR,ZDIM
      DO 30 I=1,N
        WRITE(8,*)RX(I),RY(I),RZ(I),VX(I),VY(I),VZ(I),
     +         DISPX(I),DISPY(I),DISPZ(I)
 249       FORMAT(2X,3F10.8,2X,3F10.4,2X,3F12.5,I5)
        IF(I.GT.20)GO TO 30
        WRITE(2,249)RX(I),RY(I),RZ(I),VX(I),VY(I),VZ(I)
 30   CONTINUE

C           ********
C           ********

C  the next bit should be changed so that it writes formatted output
C  in agreement w/ the way we write out the positions, velocities and
C  displacements
C
      IF(JUMP.ne.1) then
C WRITE ACCUMULATORS TO TAPE 8
        WRITE(8,401) NAV,XTIME
        DO 73 I=1,NHIST
         WRITE(8) DEN(I), TE2(I),PXX(I),PYY(I),PZZ(I)
C             WRITE(8) ZPOS(I),P2XX(I),P2YY(I),P2ZZ(I)
 73     CONTINUE
        WRITE(8) RL,RLL,R12,N12,NPOS,NPAIRS,DENSQ,XLENXL
          WRITE(8)RT3,RL13,R123,N123,NPOS1,G,NPAIRS,DENSQ,XL13,XL23
      endif
C

C           ******** SKIP FOR BCC CALCULATIONS
      IF(JUMP.EQ.1) GOTO 10
C           ********

C
C PRINT OUT PROFILES
      IPRINT=0
 52   DENTOT=0.0
      TE2TOT=0.0
      ZPOST=0.0
      XRVTOT=0.0
      YRVTOT=0.0
      ZRVTOT=0.0
      PXXTOT=0.0
      PYYTOT=0.0
      PZZTOT=0.0
      XNAV=NAV
      VOLHIS=(ZL/XHIST) * YL * XL * XL**3
      WRITE(2,291) IN,NTOT,IPRINT,RHO
      WRITE(9,291) IN,NTOT,IPRINT,RHO
      DO 35 L=1,NHIST
      Z = (ZL/XHIST) * (L-1)* XL
      D = DEN(L)/XNAV * VOLHIS)
      IF(D.NE.0.0)GOTO 42
      ZP=0.0
      TZ = 0.0
      T=XTIME/5
      XRV=0.0
      YRV=0.0
      ZRV=0.0
      XPXX=0.0
      YPYY=0.0
      ZPZZ=0.0
      GOTO 41
 42   TZ = TE2(L)/(DEN(L)*3.0)
      ZP=ZPOS(L)*XL/DEN(L)
      XRV=XRLV(L)/DEN(L)
      YRV=YRLV(L)/DEN(L)
      ZRV=ZRLV(L)/DEN(L)

c*        implicit real*8 (a-h,o-z)

          data    d2p31m/2147483647.0d0/
          data    d2p31/2147483648.0d0/

          seed  =  dmod(16807.0d0*seed,d2p31m)
          ranf  =  seed/d2p31
          return
          end
```

```
      PROGRAM Spar
C
C                   12/19/1989
C     converted to VAX fortran
C
      IMPLICIT REAL*8(A-H, O-Z)
       COMMON/POSITION/ RX(2100),RY(2100),RZ(2100)
       COMMON/LINKCELL/ LINK(2100),KY(2100),LMAX(30), L0(812),L1(812)

       DIMENSION VX(2100),VY(2100),VZ(2100)
      DIMENSION DISPX(2100),DISPY(2100),DISPZ(2100)
      DIMENSION NK(2100),TC(2100)
     DIMENSION XA(6),YA(6),ZA(6)
       DIMENSION DEN(1010),TEL(1010),PAX(1010),PYY(1010),PZZ(1010)
        DIMENSION ZPOS(1010),ZPOS2(1010),XREV(1010),YREV(1010),ZREV(1010)
        DATA NHIST/1000/ ,PI/3.14159265/
         DATA EL1/7.8116/, EL2/27.9707/, CR1/13.3554/, CR2/ .3719/

C THIS IS A GENERAL HARD SPHERE PROGRAM USING THE NEW LINK CELL
C METHOD THE PROGRAM CAN BE USED TO STUDY SOLID FLUID INTERACTS
C THE WIDTH OF A LINK CELL SHOULD BE AT LEAST EQUAL TO 1 SIGMA
C CALCULATES DIFFERENT COMPONENTS OF PRESSURE TENSOR
C THIS VERSION WRITES CONFIGURATIONS TO TAPE 10 EVERY IWRITE
C COLLISIONS (IWRITE=100 CORRESPONDS TO 0.5 MEAN COLLISION TIMES)
C ICON IS THE NUMBER OF CONFIGURATIONS PRESENT ON TAPE
C NCONF IS THE NUMBER OF CONFIGURATIONS WRITTEN IN THIS RUN
C RHO IS THE NUMBER DENSITY IN THE SOLID REGION
C ZC IS THE BOX LENGTH AFTER STRETCHING IS ACCOUNTED FOR
C ZCS IS THE BOX LENGTH OF A ONE PHASE SOLID (I.E. UNSTRETCHED)

C READ INPUT PARAMETERS FROM INPUT FILE
       READ(1,*)N
       READ(1,*)RHO, IAV
      READ(1,*)NTOT,NTAU,IWRITE
      READ(1,*)IN,ICON
      READ(1,*)NLX, NLY, NLZ
       WRITE(2,207)NLX, NLY, NLZ
207   FORMAT(1H ,5H NLX=,I3,5H NLY=,I3,5H NLZ=,I3/)
      WRITE(2,209)N,RHO,IN,NTOT,NTAU,IWRITE,ICON
209   FORMAT(/,3H N=,I6,4X,4HRHO=,F8.6,4X,4HRUN=,I3,4X,5HNTOT=,I6,4X,
       5HNTAU=,I6,7HIWRITE=,I6,5HICON=,I6/)
      READ(1,*)NAC,NCX,NCY,NCZ
       WRITE(2,131)NAC,NCX,NCY,NCZ
131   FORMAT(4I6)
      XNCX=NCX
      XNCY=NCY
      XNCZ=NCZ
       READ(1,*)(XA(J),YA(J),ZA(J),J=1,NAC)
        WRITE(2,133)(XA(J),YA(J),ZA(J),J=1,NAC)
133   FORMAT(1H ,3(2X,F8.6))
      READ(1,*) YC, ZCS, ZC

C READ ICON CONFIGURATIONS FROM TAPE 10
C  if ICON=1 NO configurations read but averaging
C  IS continued
C
      TOTIME=0.0
      IF(ICON.EQ.1)GOTO 78
      ITOT=ICON
      TIME=0.0
76    DO 74 II=1,ITOT
      ICC=II
C*********************************************************read tape
C     READ(10)ICN,TIME,RX,RY,RZ,DISPX,DISPY,DISPZ,
C    +VX,VY,VZ
C*     IF(EOF(10).NE.0.0) GOTO 75
74    CONTINUE
      GOTO 77
75    ITOT=ICC-1
      REWIND 10
      GOTO 76
77    ICON=ITOT
      TOTIME=TIME
      WRITE(2,255) ICON,TOTIME
C
C
C SET ACCUMULATORS TO ZERO
78    NAV = 0
      NCONF=0
      XTIME=0.0

      DO 25 I=1, NHIST
      ZPOS(I)=0.0
      ZPOS2(I)=0.0
      XREV(I)=0.0
      YREV(I)=0.0
      ZREV(I)=0.0
      DEN(I)=0.0
      TEL(I)=0.0
      PXX(I)=0.0
      PYY(I)=0.0
      PZZ(I)=0.0
25    CONTINUE

C CONVERSION TO PROGRAM UNITS
      VOL=SQRT(2.0)/RHO
      T=0.0D0
      XN=N
      XHIST=NHIST
      N=1.0

      YC2=0.5D0*YC
      ZC2=0.5D0*ZC
        XL=(dsqrt(2.0d0)/RHO)**(1.0d0/3.0d0)*...
      YL=XL*YC
      ZL=XL*ZC
      AREA=XL*YL
      EL1=EL1/XL
      EL2=EL2/XL
      CR1=CR1/XL
      CR2=CR2/XL
      XINT1=EL1
      XINT2=CR1
      XINT3=CR2
      XINT4=EL2
      S=1.0D0/XL
      SS=S*S
       WRITE(2,210)S
      DELZ=ZC/XHIST
      NWRITE=IWRITE
      NRUN=NTAU
210   FORMAT(/,22H SIGMA IN UNITS OF L =,F9.6/)
      WRITE(2,88) XC,YC,ZC
      WRITE(2,89) XL,YL,ZL
       WRITE(2,2007) AREA
        write(2,2008) xl/nlx, yl/nly, zl/nlz
2007  FORMAT(1H ,5HAREA=,F13.8)
2008  format(' link cell dimensions',3(2x,f13.8))
88    FORMAT(1H ,35H BOX DIMENSIONS IN PROGR UNITS XC=,F10.6,
     +  4H YC=,F10.6,4H ZC=,F10.6/)
89    FORMAT(1H ,35H BOX DIM. IN UNITS OF SIGMA   XL=,F10.6,
     +  4H YL=,F10.6,4H ZL=,F10.6/)
C INITIAL FCC LATTICE
      I=1
      DO 1 IZ=1,NCZ
      DO 1 J=1,NAC
```

```
      DO 1 IX=1,NCX
      DO 1 IY=1,NCY
       RX(I)=((IX-1)+XA(I))/XNCX
       RY(I)=((IY-1)+YA(I))*YC/XNCY
       RZ(I)=((IZ-1)+ZA(I))*ZC/XNCZ
       RX(I)=RX(I)+0.001D0
       RY(I)=RY(I)+0.001D0
       RZ(I)=RZ(I)+0.001D0
      IF(ICON.EQ.0) DISPX(I)=0.0
      IF(ICON.EQ.0) DISPY(I)=0.0
      IF(ICON.EQ.0) DISPZ(I)=0.0
      I=I+1
    1 CONTINUE
      IF(IN.EQ.0)GO TO 71
C READ IN CONFIGURATION FROM TAPE 7
      PX=0.0
      PY=0.0
      PZ=0.0
      DO 72 I=1,N
       READ(7,*)RX(I),RY(I),RZ(I),VX(I),VY(I),VZ(I)

      PX=PX+VX(I)
      PY=PY+VY(I)
      PZ=PZ+VZ(I)
   72 CONTINUE
C READ IN ACCUMULATORS FROM TAPE 7 ********************************
      IF(IAV.EQ.1) THEN
       READ(7,*) NAV, XTIME
       DO 79 J=1,NBIST
       READ(7,*) IR(J),TU(J),PXX(J),PYY(J),PZZ(J)
       READ(7,*) ZION(J),XRI V(J),YRI V(J),ZRI V(J)
   79  CONTINUE

      ENDIF
      GOTO 10

   71 CONTINUE
C INITIAL MAXWELLIAN VELOCITIES
      PX=0.0
      PY=0.0
      PZ=0.0
      sgauss = 2 * INT(N * KBO / (ZC * YC)) + 1
      DO 5 I=1,N
      NX(I)=1
       CALL GAUSS(sgauss, 1.0, 0.0,VX(I))
       CALL GAUSS(sgauss, 1.0, 0.0,VY(I))
       CALL GAUSS(sgauss, 1.0, 0.0,VZ(I))
       WRITE(15,*) VX(I), VY(I), VZ(I)
      PX=PX+VX(I)
      PY=PY+VY(I)
      PZ=PZ+VZ(I)
    5 CONTINUE
C CHAIN LINK TABLES
   10 XNL = NLX
      XF = 1.0
      YNL = NLY
      YF = YC
      ZNL = NLZ
      ZF = ZC
      NL2 = NLX * NLY
      NL3 = NLX * NLY * NLZ
      DO 320 LL=1,NL3
      LB(LL) = 0
      LI(LL) = 0
  320 CONTINUE
      DO 321 I= 1, N
      IX = INT( RX(I) * XNL/XF + 1.0)
      IY = INT( RY(I) * YNL/YF)
      IZ = INT( RZ(I) * ZNL/ZF )
      IP = IX + NLX*IY + NL2*IZ
      IF(IP.GT.NL3) GOTO 26
      KI(I) = IP
      MI = LB(IP)
      IF (MI.EQ.0) GO TO 22
      LINK(MI) = I
      LI(IP) = I
      GOTO 321
   22 CONTINUE
      LB(IP) = I
      LI(IP) = I
  321 CONTINUE
C
C COMPUTE CHAIN
C
      DO 325 IP = 1, NL3
      MI = LB(IP)
      IF (MI.EQ.0) GOTO 325
      LINK(MI) = LB(IP)
  325 CONTINUE
      GOTO 329
   26 CONTINUE
C     WRITE (2,270) IX, IY, IZ, IP, I
C     WRITE (2,211) RX(I), RY(I), RZ(I)
      GOTO 24
  329 CONTINUE
C
C CHECK ON LINK ARRAYS
      WRITE(2,206)
      DO 330 IP=1, NL3
      J=1
      MI = LB(IP)
   31 LMAX(J) = MI
      IF( MI.EQ.0) GOTO 341
      J=J+1
      MI = LINK(MI)
      IF (MI.NE.LB(IP)) GOTO 31
      J=J-1
      IF(IP.GT.50)GOTO 330
  341 continue
C     WRITE (2,220) IP, (LMAX(KK),KK=1,J)
  330 CONTINUE
C ZERO MOMENTUM
      TEMP=0.0
      DO 6 I=1,N
       VX(I)=VX(I) PX/XN
       VY(I)=VY(I) PY/XN
       VZ(I)=VZ(I) PZ/XN
      TEMP=TEMP+VX(I)*VX(I)+VY(I)*VY(I)+VZ(I)*VZ(I)
    6 CONTINUE
C SET INITIAL TEMP
      C=SQRT(XN*3.0/TEMP)
      TEMP=0.0
      PX=0.0
      PY=0.0
      PZ=0.0
      DO 16 I=1,N
      VX(I)=VX(I)*C
      VY(I)=VY(I)*C
      VZ(I)=VZ(I)*C
      PX=PX+VX(I)
      PY=PY+VY(I)
      PZ=PZ+VZ(I)
      TEMP=TEMP+VX(I)*VX(I)+VY(I)*VY(I)+VZ(I)*VZ(I)
   16 CONTINUE
      PX=PX/XN
      PY=PY/XN
```

```
      PZ=PZ/XN
      TEMP=TEMP/(XN*3.0)
      WRITE(2,205)TEMP,PX,PY,PZ
205   FORMAT(/,13H INITIAL TEMP,F8.4,4X,15HINITIAL MOMENTA,3(2X,E13.6)
C PRINT OUT INITIAL POSITIONS VELOCITIES AND LINK TABLES
      WRITE(2,203)
203   FORMAT(/,5X,1HI,7X,2HRX,9X,2HRY,9X,2HRZ,12X,2HVX,9X,2HVY,9X,2H
     1X,7HLINK(I),4X,2HIC,/)
      DO 14 I=1,20
C     WRITE(2,208)I,RX(I),RY(I),RZ(I),VX(I),VY(I),VZ(I),LINK(I),IC(I)
208   FORMAT(I6,2(3X,3(2X,F9.6)),4X,I5,2X,I5)
14    CONTINUE
      NCOUNT=0
      NCYL=0
      PTOT=0.0
      NCT=0
      P=0.0
      TAU=0.0
      PNORM=0.0
      PTRANS=0.0

      CALL OVERLAP(NLX,NLY,NLZ,NL,XL,YL,ZL,SS)
C COLLISION PREDICTION TABLES
29    I=1
      DT=1.0D0
9     CONTINUE
C SPHERE COLLISION
      IF(I.EQ.K)GO TO 21
      IF(I.EQ.L)GO TO 21
      IF(NA(I).EQ.K)GO TO 21
      IF(NA(I).EQ.L)GO TO 21
      IF(NCT.EQ.0)GO TO 21
      GO TO 23
21    CONTINUE
      TC(I)=99.0D0
      IP=IC(I)
      IPM1=IP - 1
C   CALCULATE IX,IY,IZ FROM IP
      IZ = IPM1/NL2
      IY = (IPM1 - IZ*NL2)/NLX
      IX = IP - IY*NLX - IZ*NL2
      IY = IY + 1
      IZ = IZ + 1

C   CALCULATE JP
        DO 40  KZ = 1, 3
        DO 40  KY = 1, 3
        DO 40  KX = 1, 3
           CX = 0.0
           CY = 0.0
           CZ = 0.0
           JX = IX + KX - 2
           JY = IY + KY - 2
           JZ = IZ + KZ - 2
C          ____PERIODIC BOUNDARY CONDITIONS*****????
        IF (JX - NLX-1)  371, 370, 371
370        JX = 1
         CX = XF
371        IF (JY - NLY-1)  373, 372,373
372        JY = 1
         CY = YF
373        IF (JZ - NLZ-1)  375, 374, 375
374        JZ = 1
         CZ = ZF
375        IF (JX)          377, 376, 377
376        JX = NLX
         CX = -XF
377        IF (JY)          379, 378, 379
378        JY = NLY
         CY = -YF
379        IF (JZ)          382, 380, 382
380        JZ = NLZ
         CZ = -ZF
C
382        JP = JX + (JY-1) * NLX + (JZ-1) * NL2
C
         J = LO(JP)
C          ____ CELL JP EMPTY ?
        IF (J.EQ.0)  GOTO 40
41    CONTINUE
        IF (J.EQ.I)  GOTO 7
C
      X=RX(I)-RX(J)-CX
      Y=RY(I)-RY(J)-CY
      Z=RZ(I)-RZ(J)-CZ
      U=VX(I)-VX(J)
      V=VY(I)-VY(J)
      W=VZ(I)-VZ(J)
      B=X*U+Y*V+Z*W
      IF(B.GE.0.0D0)GO TO 7
      VV=U*U+V*V+W*W
      RR=X*X+Y*Y+Z*Z
      C=VV*(RR-SS)
      BB=B*B
      IF(C.GE.BB)GO TO 7
      Q=SQRT(BB-C)
      D=-(B+Q)/VV
      IF(C.LT.0.0D0)GO TO 37
      IF(D.GT.TC(I))GO TO 7
      TC(I)=D
      NA(I)=J
7     CONTINUE
      J=LINK(J)
      IF(J.NE.LO(JP))GO TO 41
40    CONTINUE
23    CONTINUE
      IF(TC(I).GT.DT)GO TO 20
      DT=TC(I)
      KS=I
      LS=NA(I)
20    I=I+1
      IF(I.LE.N)GO TO 9
      K=KS
      L=LS
C UPDATE POSITIONS TO NEXT COLLISION
      IF(NCOUNT.GE.NTOT)DT=0.5*DT
      DO 8 I=1,N
      X=RX(I)
      Y=RY(I)
      Z=RZ(I)
      TC(I)=TC(I)-DT
      X=X+VX(I)*DT
      Y=Y+VY(I)*DT
      Z=Z+VZ(I)*DT
      DISPX(I)=DISPX(I)+VX(I)*DT
      DISPY(I)=DISPY(I)+VY(I)*DT
      DISPZ(I)=DISPZ(I)+VZ(I)*DT
C BOUNDARY CONDITIONS
      IF(X.GE.1.0D0)X=X-1.0D0
      IF(Y.GE.YC)Y=Y-YC
      IF(Z.GE.ZC)Z=Z-ZC
      IF(X.LT.0.0D0)X=X+1.0D0
      IF(Y.LT.0.0D0)Y=Y+YC
      IF(Z.LT.0.0D0)Z=Z+ZC
      RX(I)=X
      RY(I)=Y
```

```
      R(I)=Z
C CALCULATE LINK CELL
      IX = INT( RX(I) * XNLX/XL + 1.0)
      IY = INT( RY(I) * YNLY/YL)
      IZ = INT( RZ(I) * ZNLZ/ZL)
      IP = IX + NLX*IY + NLX*NLY*IZ
      IF (IP.EQ.ICELL(I)) GOTO 34
C
C TAKE OUT OF OLD LINK CELL
C
      ML = ICELL(I)
      J = LCELL(ML)
328   IF (LINK(J).EQ.I) GO TO 327
      J = LINK(J)
      GOTO 328
327   LINK(J) = LINK(I)
      LCELL(ML) = LINK(I)
      IF (LINK(I).LT.0) LCELL(ML) = 0
C
C PUT INTO NEW LINK CELL
C
      J = LCELL(IP)
      IF (LCELL(IP).EQ.0) GO TO 33
      LINK(I) = LINK(J)
      LINK(J) = I
      ICELL(I) = IP
      LCELL(IP) = I
34    CONTINUE
8     CONTINUE
      NCT=I
      IF(NCOUNT.GE.NTOT) GO TO 35
C NEW VELOCITIES OF COLLIDING PAIR
      X=RX(K)-RX(L)
      Y=RY(K)-RY(L)
      Z=RZ(K)-RZ(L)
      IF(X.GT.0.5D0)X=X-1.0D0
      IF(Y.GT.YC2) Y=Y-YC
      IF(Z.GT.ZC2) Z=Z-ZL
      IF(X.LT.-0.5D0)X=X+1.0D0
      IF(Y.LT.-YC2) Y=Y+YC
      IF(Z.LT.-ZC2) Z=Z+ZL
      U=VX(K)-VX(L)
      V=VY(K)-VY(L)
      W=VZ(K)-VZ(L)
      B=X*U+Y*V+Z*W
      BX=X*B/SS
      BY=Y*B/SS
      BZ=Z*B/SS
      PTOT=PTOT-B
      P  =P  - B
      PNORM =PNORM -B/
      PTRANS=PTRANS-BX-BY
C CALCULATE BINS FOR PRESSURE TENSOR
      KKK=INT(RZ(K)*(XHIST/ZL))+1
      KKL=INT(RZ(L)*(XHIST/ZL))+1
      IF(KKK.EQ.KKL) GOTO 701
      KMAX=MAX0(KKK,KKL)
      KMIN=MIN0(KKK,KKL)
      NBINS=KMAX-KMIN+1
      NHIST2=NHIST/2
      IF(NBINS.LT.NHIST2) GOTO 705
C LINE JOINING THE CENTRES CROSSES THE Z BOUNDARY
      NBINS=NHIST+KMIN-KMAX+1
      WL=FLOAT(NBINS-2)*DHZ
      ABSZ=ABS(Z)
      IF(Z.GT.0.0) GOTO 706
C RZ(K).LT.RZ(L) ------ -
      ZMIN=(1.0AT(KKK)*DHZ-RZ(K)
      ZMAX=Z-ZMIN-WL
      GOTO 709
C RZ(K).GT.RZ(L) ------ --
706   ZMIN=(FLOAT(KKL)*DHZ-RZ(L)
      ZMAX=Z-ZMIN-WL
709   DO 707 KK=KMAX,NHIST
      ZI ACT=DHZ/ABSZ
      IF(KK.EQ.KMAX) ZI ACT=ZMIN/ABSZ
      PXX(KK)=PXX(KK)-BX*ZI ACT
      PYY(KK)=PYY(KK)-BY*ZI ACT
      PZZ(KK)=PZZ(KK)-BZ*ZI ACT
707   CONTINUE
      DO 708 KK=1,KMIN
      ZI ACT=DHZ/ABSZ
      IF(KK.EQ.KMIN) ZI ACT=ZMAX/ABSZ
      PXX(KK)=PXX(KK)-BX*ZI ACT
      PYY(KK)=PYY(KK)-BY*ZI ACT
      PZZ(KK)=PZZ(KK)-BZ*ZI ACT
708   CONTINUE
      GOTO 704
C LINE JOINING THE CENTRES DOES NOT CROSS THE Z BOUNDARY
705   WL=FLOAT(NBINS-2)*DHZ
      ABSZ=ABS(Z)
      IF(Z.GT.0.0) GOTO 701
C RZ(K).LT.RZ(L) ------ -
      ZMIN=(FLOAT(KKK)*DHZ-RZ(K)
      ZMAX=Z-ZMIN-WL
      GOTO 96
C RZ(K).GT.RZ(L)
701   ZMIN=(FLOAT(KKL)*DHZ-RZ(L)
      ZMAX=Z-ZMIN-WL
96    DO 95 KK=KMIN,KMAX
      ZI ACT=DHZ/ABSZ
      IF(KK.EQ.KMIN) ZI ACT=ZMIN/ABSZ
      IF(KK.EQ.KMAX) ZI ACT=ZMAX/ABSZ
      PXX(KK)=PXX(KK)-BX*ZI ACT
      PYY(KK)=PYY(KK)-BY*ZI ACT
      PZZ(KK)=PZZ(KK)-BZ*ZI ACT
95    CONTINUE
      GOTO 704
C BOTH PARTICLES IN THE SAME BIN KKK
703   PXX(KKK)=PXX(KKK)-BX
      PYY(KKK)=PYY(KKK)-BY
      PZZ(KKK)=PZZ(KKK)-BZ
704   CONTINUE
99    B=B/SS
      X=X*B
      Y=Y*B
      Z=Z*B
C CALCULATE REVERSE VELOCITIES
      VXK=VX(K)
      VYK=VY(K)
      VZK=VZ(K)
      VXL=VX(L)
      VYL=VY(L)
      VZL=VZ(L)
      VX(K)=VX(K)-X
      VY(K)=VY(K)-Y
      VZ(K)=VZ(K)-Z
      VX(L)=VX(L)+X
      VY(L)=VY(L)+Y
      VZ(L)=VZ(L)+Z
      RELV=VX(K)*VXK
      IF(REVIEW.LT.0.0) IOREV(KKK)=IOREV(KKK)+1
      RELV=VY(K)*VYK
      IF(REVIEW.LT.0.0) IOREV(KKK)=IOREV(KKK)+1
      RELV=VZ(K)*VZK
      IF(RELV.LT.0.0) IOREV(KKK)=IOREV(KKK)+1
```

```
      REV=VX(L)*VXL
      IF(REV.LT.0.0) XREV(KKL)=XREV(KKL)+1
      REV=VY(L)*VYL
      IF(REV.LT.0.0) YREV(KKL)=YREV(KKL)+1
      REV=VZ(L)*VZL
      IF(REV.LT.0.0) ZREV(KKL)=ZREV(KKL)+1
      NCOL=NCOL+1
      T=T+DT
      TAU=TAU+DT
      NCRUNT=NCOL
       NAV=NAV+1

       DO 11 I=1,N
       KK=INT(RZ(I)*(XHIST/ZL))+1
      DEN(KK)=DEN(KK)+1.0
       EKI=VX(I)*VX(I) + VY(I)*VY(I) + VZ(I)*VZ(I)
      TEZ(KK)=TEZ(KK)+EKI
      ZPOS(KK)=ZPOS(KK) + RZ(I)
      ZPOS2(KK)=ZPOS2(KK) + RZ(I)*RZ(I)
11    CONTINUE

      IF(NCOUNT.LT.NWRIT) GOTO 19
C WRITE CONFIGURATION TO TAPE 10
      TIME=TOTIME + T/S
      KCON=KCON+1
C       WRITE(10)KCON,TIME,RX,RY,RZ,DISPX,DISPY,DISPZ,
C             +VX,VY,VZ
      NWRIT=NWRIT +IWRIT
      NCONT=NCONT+1
19    IF(NCOUNT.LT.NRUN)GO TO 29
      TEMP=0.0
      PX=0.0
      PY=0.0
      PZ=0.0
      DO 27 I=1,N
      EKI = VX(I)**2 + VY(I)**2 + VZ(I)**2
      TEMP=TEMP + EKI
      PX=PX+VX(I)
      PY=PY+VY(I)
      PZ=PZ+VZ(I)
27    CONTINUE
      TEMP=TEMP/(XN*3.0)
      PX=PX/XN
      PY=PY/XN
      PZ=PZ/XN
C
C CALCULATE MEAN SQ DISPL FOR THE 3 REGIONS SEPARATELY
      DIFFL=0.0
      DIFCR=0.0
      DIFINT=0.0
      XNFL=0.0
      XNCR=0.0
      XNINT=0.0
      DO 37 I=1,N
      IF(RZ(I).GE.FL1.AND.RZ(I).LT.FL2) GOTO 38
      XNFL=XNFL+1
      DIFFL=DIFFL+DISPX(I)**2+DISPY(I)**2+DISPZ(I)**2
38    IF(RZ(I).LE.CR1.OR.RZ(I).GE.CR2) GOTO 39
      XNCR=XNCR+1
      DIFCR=DIFCR+DISPX(I)**2+DISPY(I)**2+DISPZ(I)**2
39    IF(RZ(I).GT.XINT1.AND.RZ(I).LT.XINT2) GOTO 36
      IF(RZ(I).GT.XINT3.AND.RZ(I).LT.XINT4) GOTO 36
      GOTO 37
36    XNINT=XNINT+1
      DIFINT=DIFINT+DISPX(I)**2+DISPY(I)**2+DISPZ(I)**2
37    CONTINUE
      DIFFL=DIFFL/(XNFL*SS)
      DIFCR=DIFCR/(XNCR*SS)
      DIFINT=DIFINT/(XNINT*SS)
      WRITE(2,202)NCOL,KL,T,TEMP,PX,PY,PZ
      WRITE(6,202)NCOL,KL,T,TEMP,PX,PY,PZ
202   FORMAT(/,10H COLLISION,I8,4X,11H BETWEEN,2I5,4X,5HAT T=,F8.4,4X,[illegible]
     1TEMP=,F8.4,4X,7HMOMENTA,3(2X,E13.6))
      AP=1.0+PTOT/(3.0*XN*T)
      AP=AP/VOL
      APN=1.0+PNORM/(XN*T)
      APT=1.0+PTRANS/(2.0*XN*T)
      PSURAV=1.0 + P/(3.0*XN*TAU)
      PSURAV=PSURAV/VOL
      CP=1.0+FLOAT(NTAU)*SQRT(PI)*S/(TAU*(XN-1.0)*3.0)
      CP=CP/VOL
      WRITE(2,251)PSURAV, AP, CP, DIFFL, DIFCR, DIFINT
251   FORMAT(10H PRESSURE =,E13.6,5X,4H AP=,E13.6,6H COLP=,[illegible]
     +E12.6)
      WRITE (2,2005) APN, APT
      NRUN=NRUN+NTAU
      P=0.0
      TAU=0.0
      GO TO 29
C END OF THIS RUN
85    CONTINUE
      WRITE(2,234)T,NCOL
234   FORMAT(/,11H END OF RUN,3X,2HT=,F9.6,3X,5HNCOL=,I10/)
      PTOT=1.0+PTOT/(3.0*XN*T)
      PTOT=PTOT/VOL
      WRITE(2,250)PTOT
250   FORMAT(/,17H AVERAGE PRESSURE,2X,E10.6,/)
      CP=1.0+FLOAT(NCOL)*SQRT(PI)*S/(T*(XN-1.0)*3.0)
      CP=CP/VOL
      BETAG=(APN-APT)*XN/(2.0*AREA)
      WRITE(2,253) CP
      WRITE(2,2003) BETAG
      APN=APN*XN*S/(AREA*ZL)
      APT=APT*XN*S/(AREA*ZL)
      WRITE(2,*)' units of beta*sigma**3  PNORM = ',apn
      WRITE(2,*)' units of beta*sigma**3  PTRANS= ',apt
C   overall density
      RHOTOT=XN/(XL*YL*ZL*XL*XL*XL)
      WRITE(2,*)' overall density = ',RHOTOT
      REALT=T/S
      WRITE(2,820)REALT
820   FORMAT(/,20H TIME IN HARD SPHERE UNITS ,2X,F12.9,/)
253   FORMAT(/,29H PRESSURE FROM COLLISION RATE,2X,E10.6,/)
      XTIME=XTIME+T
      WRITE(2,254)NCONT
      WRITE(2,256)KCON,TIME,XTIME
      WRITE(2,201)T
201   FORMAT(/,31H  POSITIONS AND VELOCITIES AT T= ,F8.4,/)
      DO 12 I=10,100,10
C       WRITE(2,200)I,RX(I),RY(I),RZ(I),VX(I),VY(I),VZ(I),NN(I),TC(I)
200     FORMAT(I6,2(3X,3(E7.5,F9.6)),4X,I5,3X,F8.5)
12    CONTINUE
C PUNCH OUT FINAL CONFIGURATION
      WRITE(2,252)
252   FORMAT(/,22H PUNCHED CONFIGURATION,/)
      DO 30 I=1,N
      WRITE(8,249)RX(I),RY(I),RZ(I),VX(I),VY(I),VZ(I)
C,
C     +       DISPX(I),DISPY(I),DISPZ(I)
249     FORMAT(2X,3(F10.8,2X),3(E10.4,2X),3(E12.5,1X))
      IF(I.GT.20)GO TO 30
C       WRITE(2,249)RX(I),RY(I),RZ(I),VX(I),VY(I),VZ(I)
30    CONTINUE

C WRITE ACCUMULATORS TO TAPE8
      WRITE(8,401) NAV,XTIME
```

```
      DO 73 J=1,NHIST
      WRITE(8,402) DEN(J),T12(J),PXX(J),PYY(J),PZZ(J)
       WRITE(8,403) ZPOS(J),XREV(J),YREV(J),ZREV(J),ZPOS2(J)
73    CONTINUE
C
C PRINT OUT PROFILES
      IPRINT=0
52    DENTOT=0.0
      T12TOT=0.0
      ZPOST=0.0
      ZPOST2=0.0
      XRVTOT=0.0
      YRVTOT=0.0
      ZRVTOT=0.0
      PXXTOT=0.0
      PYYTOT=0.0
      PZZTOT=0.0
      TENTOT=0.0
      XNAV=NAV
      VOLBIS=(ZL/XHIST) * YL * XL * XL**3




      WRITE(2,291) IN,NTOT,IPRINT,RHO
c        WRITE(9,291) IN,NTOT,IPRINT,RHO
      DO 35 L=1, NHIST
      Z = (ZL/XHIST) * (L-1)* XL
      D = DEN(L)/(XNAV * VOLBIS)
      IF(D.NE.0.0) GOTO 42
      ZP=0.0
      TZ = 0.0
      T=XTIME/3
      XRV=0.0
      YRV=0.0
      ZRV=0.0
      GOTO 43
42    TZ = T12(L)/(DEN(L)*3.0)
      ZP=ZPOS(L)*XL/DEN(L)
      ZP2=ZPOS2(L)*XL*XL/DEN(L)
      XRV=XREV(L)/DEN(L)
      YRV=YREV(L)/DEN(L)
      ZRV=ZREV(L)/DEN(L)
43    XPXX=PXX(L)/XTIME
      YPYY=PYY(L)/XTIME
      ZPZZ=PZZ(L)/XTIME
      DENTOT = DENTOT + DEN(L)/XNAV
      T12TOT = T12TOT + T12(L)/(XNAV * 3.0)
      ZPOST=ZPOST+ ZPOS(L)*XL/XNAV
      ZPOST2=ZPOST2+ ZPOS2(L)*XL*XL/XNAV
      XPXX=D+XPXX/(AREA*DELZ*XL)
      YPYY=D+YPYY/(AREA*DELZ*XL)
      ZPZZ=D+ZPZZ/(AREA*DELZ*XL)
      XRVTOT=XRVTOT+XREV(L)/T
      YRVTOT=YRVTOT+YREV(L)/T
      ZRVTOT=ZRVTOT+ZREV(L)/T
      PXXTOT=PXXTOT+PXX(L)/XTIME
      PYYTOT=PYYTOT+PYY(L)/XTIME
      PZZTOT=PZZTOT+PZZ(L)/XTIME
      TENTOT=PZZTOT-0.5*(PXXTOT+PYYTOT)
      IF(IPRINT.EQ.1) GOTO 50
      WRITE(2,290) Z,D,ZP,DENTOT,ZPOST,XPXX,YPYY,ZPZZ
c*    WRITE(9,290) Z,D,ZP,DENTOT,T12TOT,XRVTOT,YRVTOT,ZRVTOT
       WRITE(3,2911) Z,D, zpzz-(xpxx+ypyy)/2.0, zp, zp2
2911     format(3x,f10.6,4(2x,f14.6))
      GOTO 35
50    CONTINUE
c     WRITE(2,290)Z,DENTOT,T12TOT,PZZTOT,PXXTOT,PYYTOT,ZPOST,ZP2
c     WRITE(9,290)Z,DENTOT,T12TOT,PZZTOT,PXXTOT,PYYTOT,ZPOST,ZP2
35    CONTINUE
      IPRINT=IPRINT+1
      IF(IPRINT.EQ.1)GOTO 52
      GOTO 18
17      WRITE(2,204)I,J
204   FORMAT(/,23H CATASTROPHIC COLLISION,2I5,/)
      WRITE(2,233)I,J,D,X,Y,Z,U,V,W
       WRITE(2,236)RX(I),RY(I),RZ(I),IC(I),LINK(I)
      WRITE(2,236)RX(J),RY(J),RZ(J),IC(J),LINK(J)
233     FORMAT(2I6,2X,G14.6,3(2X,F9.5),3X,3(2X,F9.5))
236     FORMAT(3F10.7,2I6)
18    CONTINUE
206   FORMAT(1H1)
211   FORMAT(1H ,28HRX(I) RY(I) Z(I) ARE RESP ,3(2X,F9.7))
220    FORMAT(I6,2X,2I15)
254   FORMAT(1H ,I10,24HCONFIGURATIONS TO TAPE 10,/)
255   FORMAT(1H ,I10,19HCONFIGURATIONS READ,2X,7HTOTIME =,G12.6)
256   FORMAT(1H ,I10,24HCONFIGURATIONS ON TAPE 10,2X,7HTOTIME =,G12.6
     +6HXTIME=,G12.6,/)
270   FORMAT(1H ,27HIX, IY, IZ, IP, I ARE RESP ,5(2X,I8))
290    FORMAT(2X,F8.4,8(2X,G14.8))
291    FORMAT(3I8,F9.6)
401    FORMAT(I20,G14.8)
402    FORMAT(5(2X, G14.8))
403    FORMAT(4(2X, G14.8))
2003  FORMAT(1H ,10X,32HDELTAGAMMA/SIG.SQ SURFACE TENSION =,F10.6,/)
2005  FORMAT(7H PNORM=,F15.8,2X,8H PTRANS=,F15.8)
24    CONTINUE
      close(2)
      close(6)
      close(3)
      close(8)
      close(15)
    STOP
    END

C*********************************************************************
      SUBROUTINE OVERLAP(NLX,NLY,NLZ,NL2,XL,YL,ZL,SS)

      IMPLICIT REAL*8(A-H,O-Z)
       COMMON/POSITION/ RX(2100),RY(2100),RZ(2100)
       COMMON/LINKCELL/ LINK(2100),KC(2100),LMAX(30), LO(512),LH(512)

29    I=1
9     CONTINUE

      IP=KC(I)
      IPM1=IP-1
C     CALCULATE IX,IY,IZ FROM IP
      IZ = IPM1/NL2
      IY = (IPM1 - IZ*NL2)/NLX
      IX = IP - IY*NLX - IZ*NL2
      IY = IY + 1
      IZ = IZ + 1

C     CALCULATE JP
         DO 40 KZ = 1, 3
         DO 40 KY = 1, 3
         DO 40 KX = 1, 3
             CX = 0.0
             CY = 0.0
             CZ = 0.0
             JX = IX + KX - 2
             JY = IY + KY - 2
             JZ = IZ + KZ - 2
C                   PERIODIC BOUNDARY CONDITIONS********
```

```
      IF (JX - NLX-1)  371, 370, 371
370     JX = 1
      CX = XL
371     IF (JY - NLY-1)  373, 372,373
372     JY = 1
      CY = YL
373     IF (JZ - NLZ-1)  375, 374, 375
374     JZ = 1
      CZ = ZL
375     IF (JX)          377, 376, 377
376     JX = NLX
      CX = -XL
377     IF (JY)          379, 378, 379
378     JY = NLY
      CY = -YL
379     IF (JZ)          382, 380, 382
380     JZ = NLZ
      CZ = -ZL
C
382     JP = JX + (JY-1) * NLX + (JZ-1) * NLZ
C
      J = LO(JP)
C             ....... CELL JP EMPTY ?
      IF (J.EQ.0)  GOTO 40
41  CONTINUE
      IF (I.EQ.J)  GOTO 7
C
   X=RX(I)-RX(J)-CX
   Y=RY(I)-RY(J)-CY
   Z=RZ(I)-RZ(J)-CZ
  RR=X*X+Y*Y+Z*Z
  IF (RR .LT. SS) THEN
   WRITE(2,*) 'CONFIGURATION HAS AN OVERLAP',I,J
  ENDIF
7   CONTINUE
  J=LINK(J)
  IF(J.NE.LO(JP))GO TO 41
40  CONTINUE
23  CONTINUE
30    I=I+1
  IF(I.LE.N)GO TO 9

  RETURN
  END
C*********************************************************
   SUBROUTINE gauss(XX,S,AM,V)
   IMPLICIT REAL*8 (A-H,O-Z)
   REAL*4 Y
   SUM=0.0
   DO 1 I=1,12
    Y=RANF(XX)
    SUM=SUM+Y
1  CONTINUE
   V=(SUM-6.0)*S+AM
  RETURN
  END
C****************************************************************
        real function ranf(seed)

C ...........................................................
C        random no generator
C        Peter Monson
C ...........................................................

        double precision seed,d2p31m,d2p31

        data  d2p31m/2147483647.0d0/
        data  d2p31/2147483648.0d0/

        seed = dmod(16807.0d0*seed,d2p31m)
        ranf = seed/d2p31
        return
        end
        integer function lnonblnk(string)

        character string*(*)

        do 100 i = len(string),1,-1
              if (string(i:i).ne.' ') goto 110
100     continue
110     lnonblnk = i
        return
        end

c 1950
c 0.94100000  .0                          rho, IAV
c    1     1  2000                  # of col, salt as # and something else
c 19  1          iconst  ....averaging continued
c 4  6 26              nlx, nly ,nlz (link cells)
c    6    5    5   13  nac ncx ncy ncz
c  0.000000  0.000000  0.000000        frac coords
c   0.500000   0.500000   0.000000
c   0.000000   0.333334   0.333334
c   0.500000   0.833334   0.333334
c   0.000000   0.666667   0.666667
c   0.500000   0.166667   0.666667
C YC,ZCS,ZF
```

```
      PROGRAM HSNPT
C
C     1)this version contained calls to nag routines to start initial
C      maxwellian velocities
C      these calls have been replaced by calls to "GAUSS"
C     2)These GOTO's have been replaced by if statements that use the
C      variable JUMP in order to get rid of
C         no path to this statements messages
C     3)Switched to double precision
C     4)Disabled the (mixed) output of accumulators etc
C    5)this version has accelerated walls to do NPT MD
C    6)this 'this version' allows the option of only having the right
C       wall move   (spike 16/3/92)
C
C                               Urbana   16/12/88
C
C            (TAPE7,TAPE8,TAPE9,TAPE10,INPUT,OUTPUT,
C   +         TAPE1=INPUT,TAPE2=OUTPUT)
      IMPLICIT REAL*8 (A-H,O-Z)
       COMMON/BLOC/ RX(2100),RY(2100),RZ(2100)
        COMMON/VELOC/VX(2100),VY(2100),VZ(2100),TC(2100)
       DIMENSION DISPX(2100),DISPY(2100),DISPZ(2100)
      DIMENSION NK(2100)
       DIMENSION LINK(2100),KC(2100),LMAX(130)
       DIMENSION L0(900),L1(900)
      DIMENSION XA(6),YA(6),ZA(6)
       DIMENSION DEN(910),TEE(910),PXX(910),PYY(910),PZZ(910)
       DIMENSION P2XX(910),P2YY(910),P2ZZ(910)
       DIMENSION ZPOS(910),XREV(10),YREV(10),ZREV(10)
      DIMENSION NPAIRS(2),DENSQ(2),NI(2),NPART(2,50)
      DIMENSION RT(2),NPRS(2),NT(2), NIK(2), XT1(2), XT2(2)
       DIMENSION RT1(2), RT2(2), NT2(2), G(2,160)
       DIMENSION NPAIRSX(2),DENSQX(2),NIX(2),NPARTX(2,80)
        DIMENSION RTX(2),NPRSX(2),NTX(2),XT1X(2),XT2X(2)
        DIMENSION RT1X(2),RT2X(2),NT2X(2),GX(2,160)
         DATA  NHIST/900/ ,NHISTG/150/,  PI/3.141592654/
           DATA  FL1/7.8116/,  FL2/27.9707/,  CR1/13.3554/,  CR2/23.3719/
            DATA  vw1/0.0/,vwr/0.0/,N301/0/,N00/0/,NEWR/0/,NIMP/1/
     LOGICAL LSCALE
C
C     OPEN(UNIT=1,FILE='INPUT.DAT',STATUS='OLD')
C     OPEN(UNIT=2,FILE='OUT.LIS',STATUS='NEW')
C     OPEN(UNIT=7,FILE='OLD.DAT',STATUS='OLD')
C     OPEN(UNIT=8,FILE='NEW.DAT',STATUS='NEW')
C
C   DATE  27/9/1983
C THIS IS A HARD WALL VERSION
C   INSERTED GOTO'S 500,501,99,19,37,18
C ICON=0 SO NO CONFIGURATIONS WRITTEN TO TAPE
C
C
C THIS IS A GENERAL HARD SPHERE PROGRAM, USING THE NEW LINK CELL
C METHOD.THE PROGRAM CAN BE USED TO STUDY FLUID-WALL INTERFACES
C THE WIDTH OF A LINK CELL SHOULD BE AT LEAST EQUAL TO 1.304 SIGMA
C CALCULATES DIFFERENT COMPONENTS OF PRESSURE TENSOR
C THIS VERSION WRITES CONFIGURATIONS TO TAPE10 EVERY IWRITE
C COLLISIONS (IWRITE=100 CORRESPONDS TO 6.3 MEAN COLLISION TIMES)
C ICON IS THE NUMBER OF CONFIGURATIONS PRESENT ON TAPE
C NCONF IS THE NUMBER OF CONFIGURATIONS WRITTEN IN THIS RUN
C CALCULATES THE STRESS AT POINT OF CONTACT
C AND IN THE CORRECT IRVING KIRKWOOD WAY
C  if move equals 0 no walls move; 1 only right wall, 2 both walls

C READ INPUT PARAMETERS FROM TAPE1=INPUT
      READ(1,*)N, LSCALE, NSCALE
        READ(1,*)RHO,rhocol,pext,pmass,move
      READ(1,*)NTOT,NTAU,IWRITE
      READ(1,*)IN,ICON,IAV,KCONT
      READ(1,*)NLX, NLY, NLZ
         write(2,*) 'pmass=',pmass
          write(2,*) 'ext pressure=',pext
         if(move.eq.2) write(2,*) 'moving walls'
         if(move.eq.1) write(2,*) 'moving right wall'
        if(move.eq.0) write(2,*) 'fixed  walls'
       WRITE(2,*)LSCALE
      IF(LSCALE) THEN
        WRITE(2,*) 'VELOCITIES SCALED  EVERY ',NSCALE
      ENDIF
      WRITE(2,*) KCONT
      IF(KCONT.EQ.1) THEN
         WRITE(2,*) 'VELOCITIES are kept'
           WRITE(2,*) 'no temp scaling or zeroing of momentum'
      ENDIF
      WRITE(2,207)NLX, NLY, NLZ
 207  FORMAT(1H ,5H NLX=,I3,5H NLY=,I3,5H NLZ=,I3,/)
      WRITE(2,209)N,RHO,IN,NTOT,NTAU,IWRITE,ICON,IAV
 209  FORMAT(/,3H N=,I6,4X,4HRHO=,F8.6,4X,4HRUN=,I3,4X,5HNTOT=,I6,4X
     1,5HNTAU=,I6,7HIWRITE=,I6,5HICON=,I6,5H IAV=,I6,/)
      READ(1,*)NAC,NCX,NCY,NCZ
 111  FORMAT(4I6)
      WRITE(2,111)NAC,NCX,NCY,NCZ
      XNCX=NCX
      XNCY=NCY
      XNCZ=NCZ
      NLZM1=NLZ-1
       READ(1,*)(XA(I),YA(I),ZA(I),I=1,NAC)
        WRITE(2,133)(XA(I),YA(I),ZA(I),I=1,NAC)
 133  FORMAT(1H ,3(2X,F8.6))
      READ(1,*) YC, ZC

C READ ICON CONFIGURATIONS FROM TAPE 10
      TOTIME=0.0
      IF(ICON.EQ.0) GOTO 78
      ITOT=ICON
      TIME=0.0
 76   DO 74 II=1,ITOT
        KC=II
        READ(10)KCN,TIME,RX,RY,RZ,DISPX,DISPY,DISPZ
C the next statement has been commented out. Probably not standard
C FORTRAN. 12/20/88
C          IF(EOF(10).NE.0.0) GOTO 75
 74   CONTINUE
      GOTO 77
 75   ITOT=KC-1
      REWIND 10
      GOTO 76
 77   ICON=ITOT
      TOTIME=TIME
      WRITE(2,255) ICON,TOTIME
C
C
C SET ACCUMULATORS TO ZERO
 78   NAV = 0
       bmi=0.0
      NCONF=0
      XTIME=0.0
      DO 25 J=1, NHIST
C***  XREV(J)=YREV(J)=ZREV(J)=0.0
        PXX(J)=0.0
        PYY(J)=0.0
        PZZ(J)=0.0
         P2XX(J)=0.0
         P2YY(J)=0.0
         P2ZZ(J)=0.0
        DEN(J)=0.0
        TEE(J)=0.0
```

```
      ZPOS(I)=0.0
25    CONTINUE
      NEND=NHISTG+10
      DO 4011 I=1,2
       DO 4011 J=1,NEND
         G(I,J)=0.0
         GX(I,J)=0.0
4011  CONTINUE
C ZERO ACCUMULATORS USED FOR FIRST AND LAST BIN
      DO 4012 I=1,2
       NPAIRS(I)=0
       N(I)=0
       NPRS(I)=0
       NT(I)=0
       NT2(I)=0
       DENS(X(I)=0.0
       RT(I)=0.0
       RT1(I)=0.0
       RT2(I)=0.0
       XT1(I)=0.0
       XT2(I)=0.0
       NPAIRSX(I)=0
       N1X(I)=0
       NPRSX(I)=0
       NT3(I)=0
       NT2X(I)=0
       DENSQX(I)=0.0
       RT3(I)=0.0
       RT1X(I)=0.0
       RT2X(I)=0.0
       XT1X(I)=0.0
       XT2X(I)=0.0
       DO 4012 J=1,50
        NPART(I,J)=0
        NPARTX(I,J)=0
4012  CONTINUE

C CONVERSION TO PROGRAM UNITS
      VOL=SQRT(2.0)/RHO
      T=0.000
      XN=N
      XHIST=NHIST
      NHIST1=NHIST+1
      NHIST3=NHIST1-3
      XHISTG=NHISTG
      XC=1.0
c     YC=DSQRT(3.000)
c*****      ZC=2.0*DSQRT(6.000)*(4.0/5.0)*RHOSOL/RHO
c      ZC=DSQRT(6.000)*(area/Vacs)*RHOSOL/RHO
      YC=0.500*YC
      ZC=0.500*ZC
c*****      XL=(XN/(RHO*YC*ZC))**(1.000/3.000)
          sl = (dsqrt(2.000/Mmol))**(1.000/3.000)*sca
      YL=XL*YC
      ZL=XL*ZC
      ZDIM=ZC
       ZHALF=ZL+1.0-1.91)/XL
      AREA=XL*YL
      FL1=FL1/XL
      FL2=FL2/XL
      CR1=CR1/XL
      CR2=CR2/XL
      XINT1=FL1
      XINT2=CR1
      XINT3=CR2
      XINT4=FL2
       S=1.000/XL
       SS=S*S
      WRITE(2,210)S
C                  set up the piston mass and external pressure
C                 etc
       amt1   = 1.0  /(1.0+pmass)
       amt2   = pmass/(1.0+pmass)
       amv1   = 1.0  /(1.0+pmass)
       amv2   = pmass/(1.0+pmass)
      fest  = pext*area
       aleft = fest/(5*pmass)
        aright= fest/(5*pmass)
      if (move .eq. 1) then
       amt1   = 0.0
       amt2   = 1.0
       aleft  = 0.0
      endif
       if (move .eq. 0) then
       amt1   = 0.0
       amt2   = 1.0
       amv1   = 0.0
       amv2   = 1.0
       aleft  = 0.0
        aright= 0.0
      endif
C
     NWRITE=NWRITE
     NRUN=NTAU
210   FORMAT(//,22H SIGMA IN UNITS OF 1.0 ,F10.6,/)
     WRITE(2,88) XC,YC,ZC
     WRITE(2,89) XL,YL,ZL
      WRITE(2,2007) AREA
2007  FORMAT(1H ,5HAREA=,F13.8)
88    FORMAT(0H ,35H BOX DIMENSIONS IN PROGRAM UNITS XC=,F10.6
     +4H YC=,F10.6,4H ZC=,F10.6,/)
89    FORMAT(1H ,35H BOX DIM  IN UNITS OF  SIGMA   XL=,F10.6,
     +4H YL=,F10.6,4H ZL=,F10.6,/)
C INITIAL FCC LATTICE
      rx1 = 0.0
      rx2 = dx
      vx1 = 0.0
      vx2 = 0.0
     IX1,IZ=ZDIM/XHIST
     RWLTST=RWL*0.04*5
     RWRTST=RWR*0.04*5

     I=1
     DO 1 IZ=1,NCZ
       DO 1 J=1,NAC
         DO 1 IX=1,NCX
           DO 1 IY=1,NCY
              RX(I)=((IX-1)+XA(J))/XNCX
              RY(I)=((IY-1)+YA(J))*YC/XNCY
              RZ(I)=((IZ-1)+ZA(J))*ZC/XNCZ
               RX(I)=RX(I)+0.001D0
               RY(I)=RY(I)+0.001D0
               RZ(I)=RZ(I)+0.001D0
              IF(ICON.EQ.0) DISPX(I)=0.0
              IF(ICON.EQ.0) DISPY(I)=0.0
              IF(ICON.EQ.0) DISPZ(I)=0.0
              I=I+1
1     CONTINUE
     IF(IN.EQ.0)GO TO 71
C READ IN CONFIGURATION FROM TAPE7
     PX=0.0
     PY=0.0
     PZ=0.0
     ZMIN=ZC
     ZMAX=0.0
C READ IN WALL POSITIONS FROM THE PREVIOUS RUN
```

```
      kze=3
      if(iz eq 0)  kze=2
      if(iz eq nlz) kze=2
c
C*****CK WALL COLLISIONS
c
        CALL  WCOL(move,rad,pmass,vwl,vwr,RZ(I),VZ(I),dt(i),angle,
     &          ic(i),ng(i))
c       write(3,*) 'c',  ic ,ic(i),ng(i),ng(i)
c
C   CALCULATE  IP
      DO 40  KZ = KZS, KZI
        DO 40  KY = 1, 3
          DO 40  KX = 1, 1
            CX = 0.0
            CY = 0.0
            IX = IX + KX - 2
            IY = IY + KY - 2
            IZ = IZ + KZ - 2
C   IMPOSE BOUNDARY CONDITIONS IN X AND Y DIRECTIONS
            IF (IX .NE. X1) 371, 370, 371
370         IX = 1
            CX = XI
371           IF (IY .NE. Y1) 373, 372,373
372           IY = 1
            CY = YI
373           IP = IX(IX)          377, 376, 377
376           IX = NEX
            CX = -XI
377           IP = IY(IY)          379, 378, 379
378           IY = NEY
            CY = -YI
C
379         IP = IX + (IY-1) * NEX + (IZ-1) * NL2
C
            I = LO(IP)
C                   ... CELL IP IMPTY ?
            IF (I.EQ.0)   GOTO 40
41          CONTINUE
            IF (I.EQ.I)   GOTO 7
C
             X=RX(I)-RX(J)-CX
             Y=RY(I)-RY(J)-CY
             Z=RZ(I)-RZ(J)
              U=VX(I)-VX(J)
              V=VY(I)-VY(J)
              W=VZ(I)-VZ(J)
            R=X*U+Y*V+Z*W
            IF(R.GE.0.000)GO TO 7
            VV=U*U+V*V+W*W
            RR=X*X+Y*Y+Z*Z
             C=VV*(RR-SS)
             BB=R*R
            IF(C.GE.BB)GO TO 7
            Q=SQRT(BB-C)
              D=-(R+Q)/VV
            IF(C.LT.0.000)GO TO 17
            IF(D.GT.TC(I))GO TO 7
            TC(I)=D
            NK(I)=J
7           CONTINUE
            J=LINK(I)
            IF(J.NE.LO(IP))GO TO 41
40    CONTINUE
23    CONTINUE
      IF(TC(1).GT.DT)GO TO 20
      DT=TC(1)
      KS=1
            if (ic.ne.0) then
             VX(I)=VX(I)*C
             VY(I)=VY(I)*C
             VZ(I)=VZ(I)*C
            endif
           PX=PX+VX(I)
           PY=PY+VY(I)
           PZ=PZ+VZ(I)
           TEMP=TEMP+VX(I)*VX(I)+VY(I)*VY(I)+VZ(I)*VZ(I)
16    CONTINUE
      PX=PX/XN
      PY=PY/XN
      PZ=PZ/(PMASS*(VWL+VWR))
      PZ=PZ/XN
      TEMP=TEMP/(XN*3.0)
c
        if (move .eq. 0) go to 1112
      WALL=TEMP*(VWL**2+VWR**2)*(PMASS/6.0)
      WRITE(42,*)  WALL, TEMP, WALL/TEMP
1112    continue
c
      WRITE(42,205)TEMP,PX,PY,PZ
205   FORMAT(/,'THE INITIAL  TEMP IS',E14.4,5X,'INITIAL  MOMENTA',3(2X,E13.6))
C*****PRINT OUT INITIAL POSITIONS VELOCITIES AND LINK TABLES
c     WRITE(42,201)
201   FORMAT(/,5X,'III',2X,'IOLD',9X,'IRX',9X,'IRY',9X,'IRZ',12X,'IVX',9X,'IVY',9X,'IVZ',
     1X,'III',5X,'I',4X,'IOLD',/)
c     DO 14 I=1,20
c       WRITE(42,206)I,RX(I),RY(I),RZ(I),VX(I),VY(I),VZ(I),LINK(I),IC(I)
206     FORMAT(I6,2(3X,3(2X,F9.6)),4X,I5,2X,I5)
c14   CONTINUE
      NCOLINT=0
      NCOL=0
       swest=0
       pwest=0
      PTEST=0.0
      NCT=0
       P=0.0
       pe=0.0
       pwest=0.0
      PNORM=0.0
      PTRANS=0.0
      TAU=0.0
       wtav=0.0
       dtav=0.0
C*****RESTORE COLLISION TABLES
29    I=1
      DT=1.000
9     CONTINUE
C*****NO COLLISION
      IF(I.EQ.KS)GO TO 21
      IF(I.EQ.1)GO TO 21
      IF(NRD.EQ.KS)GO TO 21
c         IF(NRD.EQ.1.AND.NE().NE.1)GO TO 21
      IF(NRD.EQ.1)GO TO 21
      IF(NCT.EQ.0)GO TO 21
      GO TO 23
21    CONTINUE
      TC(I)=99.000
      IP=K(I)
      IPM1=IP-1
C   CALCULATE IX,IY,IZ FROM IP
      IZ = IPM1/NL2
      IY = (IPM1 - IZ*NL2)/NLX
      IX = IP - IY*NLX - IZ*NL2
      IY = IY + 1
      IZ = IZ + 1
      kze=1
```

```
      READ(7,*) RWL,RWR
      READ(7,*) vwl,vwr
      READ(7,*) ZDIM
      DO 72 I=1,N
        READ(7,*)RX(I),RY(I),RZ(I)
        READ(7,*)VX(I),VY(I),VZ(I)
c       READ(7,*)DISPX(I),DISPY(I),DISPZ(I)
C     SHIFT ALL THE PARTICLES TO THE LEFT
C     IF(IN.NE.0)RZ(I)=RZ(I)-RWL
      PX=PX+VX(I)
      PY=PY+VY(I)
      PZ=PZ+VZ(I)
      IF(RZ(I).LT.ZMIN) ZMIN=RZ(I)
      IF(RZ(I).GT.ZMAX) ZMAX=RZ(I)
72    CONTINUE
C SHIFT THE TWO WALLS TO THE LEFT
C     RWR=RWR-RWL
C     RWL=RWL-RWL
      ZDIM=RWR-RWL
      WRITE(2,2002) RWL,RWR,ZDIM,ZRLQ,vwl,vwr

C CHANGE TO NEW DIMENSIONS OF THE WALL POSITIONS
C     RWL=RWL*S/2.0
C     RWR=RWR*S/2.0
C     ZDIM=RWR-RWL
C     WRITE(2,2002) RWL,RWR,ZDIM,ZRLQ,vwl,vwr

      DO 2 Z=ZDIM/XHIST
      RWLTST=RWL-0.04*S
      RWRTST=RWR+0.04*S
C READ ACCUMULATORS FROM TAPE7
      IF(IAV.EQ.0) GOTO 10
      READ(7,*) NAV, XTIME,hist
      DO 70 I=1,NHIST
        read(7,*) DEN(I),TEST(I),PXX(I),PYY(I),PZZ(I)
        read(7,*)ZPOS(I),P2XX(I),P2YY(I),P2ZZ(I)
70    CONTINUE
      read(7,*)RT,RT1,RT2,NT2,NPRS,G,NPAIRS,DENSQ,XT1,XT2
        read(7,*)RT3,RT13,RT*3,NT23,NPRS3,G3,NPAIRS3,DENSQ3,
     +XT13,XT23
      GOTO 10
11    CONTINUE
C INITIAL MAXWELLIAN VELOCITIES
      PX=0.0
      PY=0.0
      PZ=0.0
      XGAUSS=123
      DO 5 I=1,N
        NR(I)=1
        CALL GAUSS(XGAUSS,1.0,0.0,VX(I))
        CALL GAUSS(XGAUSS,1.0,0.0,VY(I))
        CALL GAUSS(XGAUSS,1.0,0.0,VZ(I))
        PX=PX+VX(I)
        PY=PY+VY(I)
        PZ=PZ+VZ(I)
5     CONTINUE
C CHAIN LINK TABLES
10    XNL=NLX
      XF=1.0
      YNL=NLY
      YF=YC
      ZNL=NLZ
      ZF=ZDIM
      NL3=NLX*NLY
      NL3=NLX*NLY*NLZ
      DO 320 LL=1,NL3
        L0(LL)=0
        L1(LL)=0
320   CONTINUE
      DO 321 I=1,N
        IX=INT(RX(I)*XNL/XF+1.0)
        IY=INT(RY(I)*YNL/YF)
        IZ=INT(RZ(I)*ZNL/ZF)
        IP=IX+NLX*IY+NL2*IZ
        IF(IP.GT.NL3) GOTO 26
        IC(I)=IP
        MI=L1(IP)
        IF(MI.EQ.0) GO TO 22
        LINK(MI)=I
        L1(IP)=I
        GOTO 321
22      CONTINUE
        L0(IP)=I
        L1(IP)=I
321   CONTINUE
C
C COMPUTE CHAIN
C
      DO 323 IP=1,NL3
        MI=L1(IP)
        IF(MI.EQ.0)GOTO 323
        LINK(MI)=L0(IP)
323   CONTINUE
      GOTO 329
26    CONTINUE
      WRITE(2,270) IX,IY,IZ,IP,I
      WRITE(2,210) RX(I),RY(I),RZ(I)
      GOTO 24
329   CONTINUE
C
C CHECK ON LINK ARRAYS
C     WRITE(2,206)
      DO 330 IP=1,NL3
        J=1
        MI=L0(IP)
31      I,MAX(J)=MI
        IF(MI.EQ.0) GOTO 341
        J=J+1
        MI=LINK(MI)
        IF(MI.NE.L0(IP)) GOTO 31
        J=J-1
        IF(IP.GT.50)GOTO 330
341     WRITE(2,220) IP,(I,MAX(KK),KK=1,J)
330   CONTINUE
C ZERO MOMENTUM
      TEMP=0.0
      PZ=PZ+PMASS*(VWL+VWR)
      DO 6 I=1,N
        if(icont.eq.0) then
          VX(I)=VX(I)-PX/XN
          VY(I)=VY(I)-PY/XN
          VZ(I)=VZ(I)-PZ/XN
        endif
        TEMP=TEMP+VX(I)*VX(I)+VY(I)*VY(I)+VZ(I)*VZ(I)
6     CONTINUE
C SET INITIAL TEMP
      C=SQRT(XN*3.0/TEMP)
      TEMP=0.0
      PX=0.0
      PY=0.0
      PZ=0.0
       if (icont.eq.0) then
       VWL=VWL*C
       VWR=VWR*C
      endif
      DO 16 I=1,N
```

```
      LS=NK(I)
20    I=I+1
      IF(I.LE.NK(K)) GOTO 9
      K=KS
      L=LS
C
C UPDATE POSITIONS TO NEXT COLLISION
C
      IF(NCOUNT.GE.NTOT)DT=0.5*DT
C
C                 UPDATE WALL POSITIONS AND VELOCITIES
       if (move .eq. 0) go to 1113
      RWR=RWR + VWR*DT + 0.5*aright*DT**2
        vwr=vwr + aright*dt
       if (move .eq. 1) go to 1117
      RWL=RWL + VWL*DT + 0.5*aleft*DT**2
        vwl=vwl + aleft*dt
1117    continue
      WALLTEMP=(VWL**2 + VWR**2)*PMASS/NDIV
        wtav=wtav+walltemp
1113    continue
C
C  shift all the particles to the left(or to the right if
C  the box was expanded) such that
C  the left wall ends up at  z = 0
C  doing this at every collision avoids problems w/ the profiles
C
       if (move .eq. 2) then
        DO 1965 I=1,N
         RZ(I)=RZ(I) - RWL
1965    CONTINUE
C                  now adjust the wall positions
C                   starting w/ the right wall
      RWR=RWR-RWL
      RWL=0.0
      endif
      ZDIM=RWR-RWL
      DELZ=ZDIM/XHIST
      RWLTST=RWL-0.04*S
      RWRTST=RWR+0.04*S
       do 99 i = 1,N
        RZ(I)=RZ(I) + VZ(I)*DT
99       continue
C
C

     DO 8 I=1,N
      X=RX(I)
      Y=RY(I)
      Z=RZ(I)
      TC(I)=TC(I)-DT
      X=X+VX(I)*DT
      Y=Y+VY(I)*DT
      DISPX(I)=DISPX(I)+VX(I)*DT
      DISPY(I)=DISPY(I)+VY(I)*DT
      DISPZ(I)=DISPZ(I)+VZ(I)*DT
C BOUNDARY CONDITIONS
       IF(X.GE.1.000)X=X-1.000
      IF(Y.GE.YC)Y=Y-YC
      IF(Z.GT.RWRTST) GOTO 2000
      IF(X.LT.0.000)X=X+1.000
      IF(Y.LT.0.000)Y=Y+YC
      IF(Z.LT.RWLTST) GOTO 2000
      RX(I)=X
      RY(I)=Y
C CALCULATE LINK CELL
      IX = INT( RX(I) * XNL/XL + 1.0)
      IY = INT( RY(I) * YNL/YL )
      IZ = INT( RZ(I) * ZNL/ZDIM)
      IF(IZ.LE.NLZM1) GOTO 7000
      IZ=NLZM1
      NRWR=NRWR+1
7000    IP = IX + NLX*IY + NL2*IZ
      IF (IP.EQ.KC(I)) GOTO 34
C
C TAKE OUT OF OLD LINK CELL
C
      MF = KC(I)
      J = LO(MF)
328     IF ( LINK(J).EQ.I) GO TO 327
      J = LINK(J)
      GOTO 328
327     LINK(J) = LINK(I)
      LO(MF) = LINK(J)
      IF (I.EQ.J) LO(MF)=0
C
C PUT I INTO NEW LINK CELL
C
      J = LO(IP)
      IF ( LO(IP).EQ.0) J=I
      LINK(I) = LINK(J)
      LINK(J) = I
      KC(I) = IP
      LO(IP) = I
34    CONTINUE
8   CONTINUE
     NCT=1

     IF(NCOUNT.GE.NTOT)GO TO 85
C
C NEW VELOCITIES OF COLLIDING PAIR
C
      if(L.LT.0) goto 502
      X=RX(K)-RX(L)
      Y=RY(K)-RY(L)
      Z=RZ(K)-RZ(L)
       IF(X.GT.0.5000)X=X-1.000
      IF(Y.GT.YC2) Y=Y-YC
       IF(X.LT.-0.5000)X=X+1.000
      IF(Y.LT.-YC2) Y=Y+YC
      U=VX(K)-VX(L)
      V=VY(K)-VY(L)
      W=VZ(K)-VZ(L)
      B=X*U+Y*V+Z*W
      BX=X*X*B/SS
      BY=Y*Y*B/SS
      BZ=Z*Z*B/SS
      PTOT=PTOT-B
      P  =P  -B
      PNORM=PNORM-BZ
      PTRANS=PTRANS-BX-BY
       goto 503
C
C                             perform the  wall  collision
C
502     if(nj(k).eq.-1)  then
        delv=vz(k)-vwl
        vz(k)=vz(k) - 2.0 * aml2*delv
       vwl  =vwl   + 2.0 * aml1*delv
         absvz=abs(-2.0*aml2*delv)
     else
        delv=vz(k)-vwr
        vz(k)=vz(k) - 2.0 * amr2*delv
       vwr  =vwr   + 2.0 * amr1*delv
         absvz=abs(-2.0*amr2*delv)
     endif
```

```
C
      pwup=pwup+absvz
       pwtot=pwtot+absvz
      nwcol=nwcol+1
       nwtot=nwtot+1
     GOTO 501
C
 501 CONTINUE
C
C CALCULATE BINS FOR PRESSURE TENSOR
      KKK=INT(RZ(K)*XHIST/ZDIM)+1
      KKL=INT(RZ(L)*XHIST/ZDIM)+1
      IF(KKK.GT.NHIST) KKK=NHIST
      IF(KKL.GT.NHIST) KKL=NHIST
      IF(KKK.EQ.KKL)GOTO 703
      KMAX=MAX0(KKK,KKL)
      KMIN=MIN0(KKK,KKL)
      NBINS=KMAX-KMIN+1
      WL=FLOAT(NBINS-2)*DELZ
       ABSZ=ABS(Z)
      IF(Z.GT.0.0) GOTO 70..
C RZ(K)LT RZ(L)
     ZMIN=FLOAT(KKK)*DELZ-RZ(K)
     ZMAX=Z-ZMIN-WL
     GOTO 96
C RZ(K)GT RZ(L)
701  ZMIN=FLOAT(KKL)*DELZ-RZ(L)
     ZMAX=Z-ZMIN-WL
96   DO 95 KK=KMIN,KMAX
       ZLACT=DELZ/ABSZ
       IF(KK.EQ.KMIN) ZLACT=ZMIN/ABSZ
       IF(KK.EQ.KMAX) ZLACT=ZMAX/ABSZ
        P2XX(KK)=P2XX(KK)-BX*ZLACT
        P2YY(KK)=P2YY(KK)-BY*ZLACT
       P2ZZ(KK)=P2ZZ(KK)-BZ*ZLACT
95   CONTINUE
     GOTO 704
703    P2XX(KKK)=P2XX(KKK)-BX
     P2YY(KKK)=P2YY(KKK)-BY
     P2ZZ(KKK)=P2ZZ(KKK)-BZ
704  CONTINUE
C
C THIS PART IS NOT USED AT THE MOMENT
C CALCULATE THE PRESSURE TENSOR AT THE POINT OF CONTACT
C***    ZCONT=0.5*(RZ(K)+RZ(L))
C***    KK=INT(ZCONT*XHIST/ZDIM)+1
C***    PXX(KK)=PXX(KK)-BX
C***    PYY(KK)=PYY(KK)-BY
C***    PZZ(KK)=PZZ(KK)-BZ
C
      B=B/SS
     X=X*B
     Y=Y*B
     Z=Z*B
C                        <<<<<<<
     IF(JUMP.EQ.1)GOTO 500
C                        <<<<<<<
C THIS PART IS NOT USED
C CALCULATE REVERSAL FREQUENCIES
     VXK=VX(K)
     VYK=VY(K)
     VZK=VZ(K)
     VXL=VX(L)
     VYL=VY(L)
     VZL=VZ(L)
C                        <<<<<<<<
500  CONTINUE
C                        <<<<<<<<
      VX(K)=VX(K)-X
      VY(K)=VY(K)-Y
      VZ(K)=VZ(K)-Z
      VX(L)=VX(L)+X
      VY(L)=VY(L)+Y
      VZ(L)=VZ(L)+Z
C                   <<<<<<<<
      IF(JUMP.EQ.1)GOTO 501
C                   <<<<<<<<
C THIS PART IS NOT USED
      REV=VX(K)*VXK
      IF(REV.LT.0.0) XREV(KKK)=XREV(KKK)+1
      REV=VY(K)*VYK
      IF(REV.LT.0.0) YREV(KKK)=YREV(KKK)+1
      REV=VZ(K)*VZK
      IF(REV.LT.0.0) ZREV(KKK)=ZREV(KKK)+1
      REV=VX(L)*VXL
      IF(REV.LT.0.0) XREV(KKL)=XREV(KKL)+1
      REV=VY(L)*VYL
      IF(REV.LT.0.0) YREV(KKL)=YREV(KKL)+1
      REV=VZ(L)*VZL
      IF(REV.LT.0.0) ZREV(KKL)=ZREV(KKL)+1
C CONTINUE FROM HERE FOR A ... CALCULATION
C                 <<<<<<
501  CONTINUE
C                 <<<<<<
     NCOLL=NCOLL+1
     T=T+DT
     TAU=TAU+DT
    NCOLINT=...
C*********
C
      EKI=0.0
      DO 756 III=1,N
         EKI=EKI + VX(III)**2 + VY(III)**2 + VZ(III)**2
756  CONTINUE
      ...
C CALL SCALE EVERY NSCALE COLLISIONS IF LSCALE=TRUE
     IF(LSCALE) THEN
      IMOD=MOD(NCOL,NSCALE)
      IF(IMOD.EQ.0) CALL SCALE(N,XN,EKI,CSCALE)
      IF(IMOD.EQ.0) VWL=VWL*CSCALE
      IF(IMOD.EQ.0) VWR=VWR*CSCALE
     ENDIF
C***********
     IMOD=MOD(NCOL,10)
        IF(IMOD)8050,8051,8050
8051    NAV=NAV+1
      NI(1)=0
      NI(2)=0
      NI3(1)=0
      NI3(2)=0
      bott=bott+zdim/...
     DO 1 I=1,N
      KK=INT(RZ(I)*XHIST/ZDIM)+1
      IF(KK.LE.0) THEN
        N00=N00+1
       KK=1
      ENDIF
      IF(KK.LT.NHIST+1)GOTO 3000
        N301=N301+1
      KK=NHIST
3000    DEN(KK)=DEN(KK)+1.0
       EKI=VX(I)*VX(I) + VY(I)*VY(I) + VZ(I)*VZ(I)
      TLE(KK)=TLE(KK)+EKI
      ZPOS(KK)=ZPOS(KK) + RZ(I)
C CALCULATE NUMBER OF PARTICLES IN FIRST AND LAST BIN
C AND STORE PARTICLE INDEX I IN ARRAY NPART OR NPART3
```

```
      IF(KK GT 3) GOTO 4000
      IF(KK NE 1) GOTO 4030
       NK(1)=NK(1)+1
      II=NK(1)
      NPART(1,II)=I
4030       NT3(1)=NT3(1)+1
      III=NT3(1)
      NPART3(1,III)=I
      GOTO 4001
4000    IF(KK LT NHIST-1) GOTO 4001
     IF (KK NE NHIST) GOTO 4031
      NK(2)=NK(2)+1
      II=NK(2)
      NPART(2,II)=I
4031       NT3(2)=NT3(2)+1
      III=NT3(2)
      NPART3(2,III)=I
4001    CONTINUE
11   CONTINUE

C  AVERAGE NUMBER OF PARTICLE DISTANCES IN FIRST AND LAST BIN
       NPAIRS(1)=NPAIRS(1)+NK(1)*(NK(1)-1)/2
       NPAIRS(2)=NPAIRS(2)+NK(2)*(NK(2)-1)/2
    DENSQ(1)=DENSQ(1)+(FLOAT(NK(1)))**2
    DENSQ(2)=DENSQ(2)+(FLOAT(NK(2)))**2
C  AVERAGE NUMBER OF PARTICLE DISTANCES IN FIRST AND LAST BINS
       NPAIRS3(1)=NPAIRS3(1)+NT3(1)*(NT3(1)-1)/2
       NPAIRS3(2)=NPAIRS3(2)+NT3(2)*(NT3(2)-1)/2
    DENSQ3(1)=DENSQ3(1)+(FLOAT(NT3(1)))**2
    DENSQ3(2)=DENSQ3(2)+(FLOAT(NT3(2)))**2

   CALL BINAV(RT,NPRS,NK,NPART,G,XC,XHIST,G,S,RT1,RT2,NT2,
  +XT1,XT2,YC2,YC)
    CALL BINAV(RT3,NPRS3,NT3,NPART3,G3,XC,XHIST,G,S,RT1,RT2,
  +NT23,XT13,XT23,YC2,YC)
C               <<<<<<<<<<<<<<<<
8050 CONTINUE
   IF(JUMP EQ 1)GOTO 19
C               <<<<<<<<<<<<<<<<
   IF(NCOUNT LT NWRIT1)GOTO 19
C WRITE CONFIGURATION TO TAPE 10
   TIME=TOTIME+TS
   KCON=KCON+1
   WRITE(10)KCON,TIME,RX,RY,RZ,DISPX,DISPY,DISPZ
   NWRIT1=NWRIT1+IWRIT1
   NCONT=NCONT+1
C               <<<<<<<<<< CONTINUE FROM HERE FOR EQUILIBR
19  IF(NCOUNT LT NRUN)GO TO 29
   TEMP=0.0
   PX=0.0
   PY=0.0
   PZ=0.0
   DO 27 I=1,N
    EKI = VX(I)**2 + VY(I)**2 + VZ(I)**2
   TEMP=TEMP + EKI
    PX=PX+VX(I)
    PY=PY+VY(I)
    PZ=PZ+VZ(I)
27  CONTINUE
   PX=PX/XN
   PY=PY/XN
c  ...........  add wall momentum
    PZ=PZ + pmass*(vwl+vwr)
C
   PZ=PZ/XN
   TEMP=TEMP/(XN*3.0)
c
      walltemp=wtav/float(ncount)
       wallpot = cwl*fexl/s + cwr*fexr/s
       enwall  =0.5*pmass*(vwl**2 +vwr**2) + wallpot
     enutot  = (N*1.5*TEMP + enwall)/(XN+MOVE)
c
C
C  CALCULATE MEAN SQ DISP FOR THE REGIONS SEPERATELY
    DIFTL=0.0
    DIFCR=0.0
    DIFINT=0.0
    XNTL=0.0
    XNCR=0.0
    XNINT=0.0
    DO 37 I=1,N
C*****IF(RZ(I) GE ... AND RZ(I) LE ...) GOTO 36
     XNTL=XNTL+1
     DIFTL=DIFTL+DISPX(I)**2+DISPY(I)**2+DISPZ(I)**2
c               .......
     GOTO 37
c               .......
36    IF(RZ(I) LE CR1 OR RZ(I) GE CR2) GOTO 39
     XNCR=XNCR+1
     DIFCR=DIFCR+DISPX(I)**2+DISPY(I)**2+DISPZ(I)**2
39    IF(RZ(I) GT XINT1 AND RZ(I) LT XINT2) GOTO 38
     IF(RZ(I) GT XINT3 AND RZ(I) LT XINT4) GOTO 38
     GOTO 37
38     XNINT=XNINT+1
     DIFINT=DIFINT+DISPX(I)**2+DISPY(I)**2+DISPZ(I)**2
37  CONTINUE
   DIFTL=DIFTL/(XNTL*SS)
C*****DIFCR=DIFCR/(XNCR*SS)
C*****DIFINT=DIFINT/(XNINT*SS)
      tempavek=av/(ncol*3.0*xn)
   WRITE(2,202)NCOL,K,L,T,TMPAV,PX,PY,PZ
   WRITE(6,202)NCOL,K,L,T,TMPAV,PX,PY,PZ
202  FORMAT(/,10H COLLISION,I7,4X,8H BETWEEN,2I5,4X,5H(AT T = ,F8.4,4X,5H
   1TEMP=,F8.4,4X,7HMOMENTA,3(2X,E13.6))
c
        write(2,883)vwl,cwl,vwl,cwr,(vwl-vwl)/s
      if (nav .ne. 0) write(2,885)boxl/nav
       write(2,884)walltemp,enutot
883   format(1x,' vwl  cwl  vwl  cwr  ...  ',7g20.6)
884    format(1x,' wall temp,  total energy  ',7g20.6)
885  format(1x,'             ...  ',7g20.6)
c
   AP=1.0+PTOT/(3.0*XN*T)
   AP=AP/VOL
   APN=1.0+PNORM/(XN*T)
   APT=1.0+PTRANS/(2.0*XN*T)
    if (nav .ne. 0)then
     APN=APN*XN*S/(AREA*(boxl*s/nav))
    APT=APT*XN*S/(AREA*(boxl*s/nav))
   endif
   PSUBAV=1.0 + P/(3.0*XN*TAU)
   PSUBAV=PSUBAV/VOL
   CP=1.0+FLOAT(NTAU ncol)*SQRT(PI*S/(TAU*(XN-1.0)*3.0)
  CP=CP/VOL
   PWALL=(PWTOT*S/T)/(2.0*AREA)
   PWSUB=(PW*S/TAU)/(2.0*AREA)
   WRITE(2,251)PSUBAV, AP, CP, DIFTL, DIFCR, DIFINT, pwsub, pwall
   if (nav .ne. 0) PWALL=PWALL*XL*YL*(boxl*s/nav)*XL*XL*XL/XN
   WRITE(2,2005) APN, APT, PWALL
251  FORMAT(10H PRESSURE=,G13.6,3X,4H AP=,E13.6,6H COLP=,E13.6
  +5G12.6)
   NRUN=NRUN+NTAU
   P=0.0
   TAU=0.0
   PW=0.0
  NWCOL=0
```

```
      GO TO 29
C END OF THIS RUN
85    CONTINUE
       WRITE(2,234)T,NCOL
234   FORMAT(/,11H END OF RUN,3X,2HT=,F9.6,1X,5HNCOL=,I8,/)
      REALT=T/5
       WRITE(2,820)REALT
820   FORMAT(/,28H TIME IN HARD SPHERE UNITS ,2X,F12.9,/)
      PTOT=1.0+PTOT/(3.0*XN*T)
      PTOT=PTOT/VOL
       WRITE(2,250)PTOT
250   FORMAT(/,17H AVERAGE PRESSURE ,2X,F10.6,/)
      CP=1.0+FLOAT(NCOL-NWTOT)*SQRT(PI)*S/(3*(XN-1.0)*3.0)
      CP=CP/VOL
       WRITE(2,253) CP
       BETAG=(APN-APT)*XNA/2.0*AREA)
      WRITE(2,2003) NWTOT, BETAG
       write(2,*) 'average boxlength=                ',boxl/nav
       WRITE(2,*) 'units of beta*sigma**3  PNORM = ',apn
       WRITE(2,*) 'units of beta*sigma**3  PTRANS= ',apt
C     overall density
      RHOTOT=XN/(XL*YL*(boxl*l/nav)*XL*XL*XL)
       WRITE(2,2006) RHOTOT
253   FORMAT(/,29H PRESSURE FROM COLLISION RATE,2X,F10.6,/)
      XTIME=XTIME+1
       WRITE(2,254)NCONT
      WRITE(2,256)ICON,TIME,XTIME
       WRITE(2,201)T
       WRITE(2,2004) RWL,RWR,ZDIM,vwl,vwr
201   FORMAT(/,33H POSITIONS AND VELOCITIES AT T= ,F8.4,/)
c       DO 12 I=10,100,10
c           WRITE(2,200)I,RX(I),RY(I),RZ(I),VX(I),VY(I),VZ(I),NRD,TEC(I)
200         FORMAT(I6,2(3X,3(2X,F9.6)),4X,I5,3X,F8.5)
c12    CONTINUE
C PUNCH OUT FINAL CONFIGURATION
c       WRITE(2,252)
252   FORMAT(/,22H PUNCHED CONFIGURATION,/)
c
C WRITE OUT FINAL POSITIONS OF THE WALLS TO TAPE8
c

      WRITE(8,*) RWL,RWR
       WRITE(8,*) vwl,vwr
      WRITE(8,*) ZDIM
       DO 30 I=1,N
         WRITE(8,*)RX(I),RY(I),RZ(I)
         WRITE(8,*)VX(I),VY(I),VZ(I)
c         WRITE(8,*)DISPX(I),DISPY(I),DISPZ(I)
249           FORMAT(2X,3(F10.8,2X),3(F10.4,2X),3(E12.5,1X))
c        IF(I.GT.20)GO TO 30
c          WRITE(2,249)RX(I),RY(I),RZ(I),VX(I),VY(I),VZ(I)
30    CONTINUE

C              cccccccc
C              cccccccc

C   the next bit should be changed so that it writes formatted output
C   in agreement w/ the way we write out the positions, velocities and
C   displacements
C
      if(JUMP.eq.1) then
C WRITE ACCUMULATORS TO TAPE8
       WRITE(8,*) NAV,XTIME,boxl
c       DO 73 J=1,NHIST
c          write(8,*) DEN(J),TEZ(J),PXX(J),PYY(J),PZZ(J)
c            write(d,*) ZPOS(J),P2XX(J),P2YY(J),P2ZZ(J)
c73     CONTINUE
c       write(8,*) RT,RT1,RT2,NT2,NPRS,G,NPAIRS,DENSQ,XT1,XT2
c          write(8,*)RT13,RT13,RT23,NT23,NPRS3,G3,NPAIRS3,DENSQ3,XT13,XT23
      endif
      XNAV=NAV
      XNAV3=XNAV
c
c        if(nav.ne.0.0) then
c       CALL PRINTAV(XNAV,NPAIRS,DENSQ,NT2,XT1,XT2,RT1,RT2,SS)
c        CALL PRINTAV(XNAV3,NPAIRS3,DENSQ3,NT23,XT13,XT23,
c     +  RT13,RT23,SS)
C CALCULATE RADIAL DISTRIBUTION FUNCTION
c       CALL RADIAL(NHISTG,XC,XL,S,XNAV,NPAIRS,DENSQ,PI,AREA,G)
c        CALL RADIAL(NHISTG,XC,XL,S,XNAV3,NPAIRS3,DENSQ3,PI,AREA,G3)
c       endif
c
C                          PRINT OUT PROFILES
        if(nav.ne.0) then
        IPRINT=0
52      DENTOT=0.0
       TEZTOT=0.0
       ZPOST=0.0
       XRVTOT=0.0
       YRVTOT=0.0
       ZRVTOT=0.0
       PXXTOT=0.0
       PYYTOT=0.0
       PZZTOT=0.0
       TENTOT=0.0
       P2XTOT=0.0
       P2YTOT=0.0
       P2ZTOT=0.0
       XNAV=NAV
c
          zdim=boxl*l/nav
          delz=zdim/nhist
C
      VOLHIS=(ZDIM/XHIST) * YL * XL * XL**3
C AVERAGE DENSITY PROFILE FOR TOP AND BOTTOM WALL
        DO 1111 L=1,50
          Z = (ZDIM/XHIST) *((2*L-1)/2.0) * XL
        LL=NHIST-L+1
        D1=DEN(L)/(XNAV*VOLHIS)
        D2=DEN(LL)/(XNAV*VOLHIS)
         D=0.5*(D1+D2)
        DLD=0.0
        IF(D.GT.0.0)DLD=LOG(D)
         WRITE(2,290) Z,D1,D2,D,DLD
1111    CONTINUE
      WRITE(2,291) IN,NTOT,IPRINT,RHO
C**** WRITE(9,291) IN,NTOT,IPRINT,RHO
      WRITE(2,*) ' PROFILES '
      DO 35 L=1, NHIST
         Z = (ZDIM/XHIST) *((2*L-1)/2.0) * XL
        D = DEN(L)/(XNAV * VOLHIS)
        IF(D.NE.0.0) GOTO 42
         ZP=0.0
        TZ = 0.0
        T=XTIME/5
         XRV=0.0
         YRV=0.0
         ZRV=0.0
        GOTO 43
42       TZ = TEZ(L)/(DEN(L)*3.0)
       ZP=ZPOS(L)*XL/DEN(L)
C***  XRV=XREV(L)/DEN(L)
C***  YRV=YREV(L)/DEN(L)
C***  ZRV=ZREV(L)/DEN(L)
C***     XPXX=PXX(L)/XTIME
C***     YPYY=PYY(L)/XTIME
```

```
C***    ZP2ZZ=P2ZZ(I)/XTIME
C***  change statement label 43 if spas etc is used
43        XP2XX=P2XX(I)/XTIME
         YP2YY=P2YY(I)/XTIME
         ZP2ZZ=P2ZZ(I)/XTIME
         XP2XX=D+XP2XX/(AREA*DELZ*XL)
         YP2YY=D+YP2YY/(AREA*DELZ*XL)
         ZP2ZZ=D+ZP2ZZ/(AREA*DELZ*XL)
        DENTOT = DENTOT + DEN(I)/XNAV
         T13TOT = T13TOT + T13(I)/(XNAV * 10)
         ZPOST=ZPOST+ZPOS(I)*XL/XNAV
C*** XRVTOT=XRVTOT+XREV(I)/T
C*** YRVTOT=YRVTOT+YREV(I)/T
C*** ZRVTOT=ZRVTOT+ZREV(I)/T
C*** PXXTOT=PXXTOT+PXX(I)/XTIME
C*** PYYTOT=PYYTOT+PYY(I)/XTIME
C*** PZZTOT=PZZTOT+PZZ(I)/XTIME
         P2ZTOT=P2ZTOT+P2ZZ(I)/XTIME
         P2XTOT=P2XTOT+P2XX(I)/XTIME
         P2YTOT=P2YTOT+P2YY(I)/XTIME
         TENTOT=P2ZTOT-0.5*(P2XTOT+P2YTOT)
         IF(IPRINT.EQ.1) GOTO 50
         WRITE(2,290) Z,D,ZP,DENTOT,T13TOT,XP2XX,YP2YY,ZP2ZZ
C*** WRITE(9,290) Z,D,ZP,DENTOT,T13TOT,XRVTOT,YRVTOT,ZRVTOT
         GOTO 35
50       WRITE(2,290)Z,DENTOT,TENTOT,P2XTOT,P2YTOT,P2ZTOT,ZPOST,ZP2ZZ
C*** WRITE(9,290)Z,DENTOT,TENTOT,PZZTOT,PXXTOT,PYYTOT,ZPOST,ZP2ZZ
35      CONTINUE
       IPRINT=IPRINT+1
       IF(IPRINT.EQ.1) GOTO 52
      enddo
C
    GO TO 18
37      WRITE(2,204)I,J
204   FORMAT(/,23H CATASTROPHIC COLLISION,2I5,/)
     WRITE(2,233)I,J,D,X,Y,Z,U,V,W
      WRITE(2,236)RX(I),RY(I),RZ(I),IC(I),LINK(I)
     WRITE(2,236)RX(J),RY(J),RZ(J),IC(J),LINK(J)
233     FORMAT(2I6,2X,G14.6,3(2X,F9.5),3X,3(2X,F9.5))
236     FORMAT(3F10.7,2I6)
    GOTO 18
2000   WRITE(2,2001) I,RZ(I),Z,RWL,RWR
      write(2,*) va(i), kt(i), np(i)
      write(2,*) dt
18   CONTINUE
206   FORMAT(1H1)
211   FORMAT(1H ,'RX(I), RY(I), Z(I) ARE RESP.',3(2X,F9.7))
220    FORMAT(I6,2X,25I3)
254   FORMAT(1H ,I10,'CONFIGURATIONS TO TAPE10',/)
255   FORMAT(1H ,I10,'CONFIGURATIONS READ',2X,'TOTIME=',G12.6)
256   FORMAT(1H ,I10,'CONFIGURATIONS ON TAPE10',2X,'TOTIME=',G12.6,
     +'XTIME=',G12.6,/)
270   FORMAT(1H ,'IX, IY, IZ, IP, I ARE RESP.',5(2X,I8))
290    FORMAT(F8.4,8(1X,G14.8))
291    FORMAT(I10,F9.6)
401    FORMAT(I20,G14.8)
2001   FORMAT(1H ,'WALL OVERLAP',I8,4(2X,G14.8),/)
2002   FORMAT(1H ,'RWL=',G14.8,'RWR=',G14.8,'ZDIM=',G14.8,
     +'ZREQ=',G14.8,'VWL=',G14.8,'VWR=',g14.8,/)
2003 FORMAT(1H ,'TOTAL NUMBER OF WALL COLLISIONS=',I10,
   +5X,'BETAGAMMASIGSQ SURFACE TENSION =',F10.6,/)
2004   FORMAT(1H ,'RWL=',G14.8,'RWR=',G14.8,'ZDIM=',G14.8,
     +'VWL=',G14.8,' VWR=',G14.8,/)
2005   FORMAT(1X,'PNORM=',E13.6,2x,' PTRANS=',E13.6,
    +7X,'PWALL=',E13.6)
2006  FORMAT(1H ,'OVERALL DENSITY',F14.8,/)
     WRITE(2,2008) N301,NRWR,N00
2008   FORMAT(1H ,'N301=',2X,I10,2X,'NRWR=',I10,
  &       N00=',2X,I10)
24  CONTINUE
    close(2)
    close(8)
   STOP
   END
C**************************************************************************
   SUBROUTINE BINAV(RT,NPRS,NPART,G,XC,XHISTG,S,
  +RT1,RT2,NT2,XT1,XT2,YC2,YC)
   IMPLICIT REAL*8 (A-H,O-Z)
     COMMON/BLOCK/RX(2100),RY(2100),RZ(2100)
     DIMENSION RT(2),NPRS(2),N(2),NPART(2,50),G(2,100)
    DIMENSION RT1(2),RT2(2),NT2(2),XT1(2),XT2(2)
C CALCULATE AVERAGE DISTANCE SQUARED
C USING NEAREST IMAGE CONVENTION
   RMAX=0.5
    DO 5000 M=1,2
     RT(M)=0.0
     NPRS(M)=0
      NIM1=N(M)-1
     NB=N(M)
      DO 5001 I=1,NIM1
       IP1=I+1
       DO 5001 J=IP1,NB
        II=NPART(M,I)
        JJ=NPART(M,J)
        X=RX(II)-RX(JJ)
        Y=RY(II)-RY(JJ)
        IF(X.GT.0.5) X=X-1.0D0
        IF(X.LT.-0.5)X=X+1.0D0
       IF(Y.GT.YC2) Y=Y-YC
       IF(Y.LT.-YC2)Y=Y+YC
       RR=X*X+Y*Y
       R=SQRT(RR)
        IF(R.GE.RMAX) GOTO 5001
        NPRS(M)=NPRS(M)+1
       RT(M)=RT(M)+RR
         LL=INT((R-S)*XHISTG/(XC-S))+1
       G(M,LL)=G(M,LL)+1.0
5001    CONTINUE
5000  CONTINUE
     DO  5002 M=1,2
     NPRSM=NPRS(M)
      IF(NPRSM.EQ.0) GOTO 5002
      XNUM=0.5+0.5*SQRT(1.0+8.0*FLOAT(NPRSM))
     RRAV=RT(M)/FLOAT(NPRSM)
      RT1(M)=RT1(M)+RRAV
     RT2(M)=RT2(M)+RT(M)
     NT2(M)=NT2(M)+NPRSM
     XT1(M)=XT1(M)+XNUM
     XT2(M)=XT2(M)+XNUM*XNUM
5002  CONTINUE
    RETURN
    END
C**************************************************************************
    SUBROUTINE PRINTAV(XNAV,NPAIRS,DENSQ,NT2,XT1,XT2,
    +RT1,RT2,IS)
    IMPLICIT REAL*8 (A-H,O-Z)
     DIMENSION NPAIRS(2),DENSQ(2),NT2(2),XT1(2),XT2(2)
    DIMENSION RT1(2),RT2(2)
C PRINT OUT AVERAGES OF FIRST AND LAST BIN
C       WRITE(2,5005)
     DO 5003 M=1,2
       PRAV=FLOAT(NPAIRS(M))/XNAV
      DSQ=DENSQ(M)/XNAV
      XNTAV=FLOAT(NT2(M))/XNAV
       XXT1=XT1(M)/XNAV
       XXT2=XT2(M)/XNAV
```

```
      RRAV=RT1(M)/XNAV
      RRAV2=RT2(M)/FLOAT(NT2(M))
      RRAV3=RT2(M)/XNAV
      RRAV=RRAV/SS
      RRAV2=RRAV2/SS
      RRAV3=RRAV3/SS
      XPAV=DSQ 2.0*PRAV
      IF(XPAV.EQ.0.0) XPAV= 1.0
      FLUCT=(DSQ XPAV*XPAV)/XPAV
      FLUCT2=(XXT2 XXT1*XXT1)/XXT1
C       WRITE(2,5004)XNAV,PRAV,DSQ,XPAV,FLUCT,RRAV,RRAV2
C       WRITE(2,5004)XNAV,RRAV3,XNTAV,XXT1,XXT2,FLUCT2
5004      FORMAT(1H ,7(1X,G16.8),/)
5003  CONTINUE
5005  FORMAT(1H ,//,13H BIN AVERAGES)
   RETURN
   END
C*****************************************************************
   SUBROUTINE RADIAL(NHISTG,XC,XL,S,XNAV,NPAIRS,DENSQ,PI,AREA,G)
   IMPLICIT REAL*8 (A-H,O-Z)
   DIMENSION XN1(2)
   DIMENSION NPAIRS(2),DENSQ(2),G(2,100)
C CALCULATE RADIAL DISTRIBUTION FUNCTION
   NHIST2=NHISTG/2
   XHISTG=NHISTG
   DELR=(0.5*XC-S)*XL/XHISTG
   DO 5006 I=1,2
   DSQ=DENSQ(I)/XNAV
   PRAV=FLOAT(NPAIRS(I))/XNAV
   XN1(I)=DSQ 2.0*PRAV
C    WRITE(2,*)XN1(I)
C       WRITE(2,5010)
   CONST=XN1(I)*XN1(I)*PI*DELR*XNAV/AREA
   X2=0.0
   Y2=0.0
   SUM2=0.0
   X1=-0.5
   Y1=-1.0
   SUM1=-0.25
   DO 5006 L=1,NHIST2
   LL=L+NHIST2
   XX=FLOAT(L)
   XXX=FLOAT(LL)
     R1=(0.5*XC-S) * (2.0*XX-1.0) *XL/(XHISTG*2.0)
     R2=(0.5*XC-S) * (2.0*XXX-1.0)*XL/(XHISTG*2.0)
    R1=R1+1.0
    R2=R2+1.0
    R13=R1**3
    R23=R2**3
   GR1=G(I,L)/(CONST*R1)
   GR2=G(I,LL)/(CONST*R2)
    X1=X1+(GR1-1.0)*R1*DELR
    X2=X2+(GR2-1.0)*R2*DELR
   Y1=Y1+(GR1-1.0)*DELR
   Y2=Y2+(GR2-1.0)*DELR
    SUM1=SUM1+(GR1-1.0)*R13*DELR
    SUM2=SUM2+(GR2-1.0)*R23*DELR
C         WRITE(2,6000) R1,GR1,Y1,X1,SUM1,R2,GR2,Y2,X2,SUM2
6000         FORMAT(1H ,10(1X,G11.5))
5010      FORMAT(1H ,28HRADIAL DISTRIBUTION FUNCTION,/)
5006 CONTINUE
   RETURN
   END
C*****************************************************************
   SUBROUTINE GAUSS(XX,S,AM,V)
   IMPLICIT REAL*8 (A-H,O-Z)
   REAL*4 Y
   SUM=0.0
     DO 1 I=1,12
      Y = RAN(XX)
      SUM=SUM+Y
1   CONTINUE
     V=(SUM-6.0)*S+AM
    RETURN
    END
C
      SUBROUTINE WCOL(move,[illegible],RZ,VZ,aleft,aright
&            [illegible])
      implicit double precision (a-h,o-z)
c
      [illegible]
c
C CHECK WALL COLLISIONS
c
c                           left wall
      [illegible]
      if (move .eq. 2) then
      [illegible]
      [illegible]
      [illegible]
      else
      [illegible]
      [illegible]
      [illegible]
      endif
      if([illegible] .and. d .gt. 0.0) then
      [illegible]
      [illegible]
      endif
c                           right wall
      [illegible]
      if (move .eq. 0) then
      [illegible]
      [illegible]
      [illegible]
      else
      [illegible]
      disc= v*v + 2.0*aright*[illegible]
      d =(v - sqrt(disc))/aright
      endif
      if(d [illegible] .and. d .gt. 0.0) then
      [illegible]
      [illegible] 2
      endif
      return
      end
C*****************************************************************
   SUBROUTINE SCALE(N,XN,EXT,C)
   IMPLICIT REAL*8 (A-H,O-Z)
     COMMON/VELOC/VX(2100),VY(2100),VZ(2100),[illegible](2100)
c
   C=SQRT(3.0*XN/EKT)
   DO 1 I=1,N
     vx(i)=vx(i)*c
     vy(i)=vy(i)*c
     vz(i)=vz(i)*c
     [illegible](i)=[illegible](i)/[illegible]
1  CONTINUE
   RETURN
   END

      real function ranf(seed)

C
C         random no. generator
C         Peter Monson
```

```
C  . . . . . .

      double precision seed,d2p31m,d2p31

c*      implicit real*4 (a-h,o-z)

        data   d2p31m/2147483647.0d0/
        data   d2p31/2147483648.0d0/

        seed  =  dmod(16807.0d0*seed,d2p31m)
        ranf  =  seed/d2p31
        return
        end
```