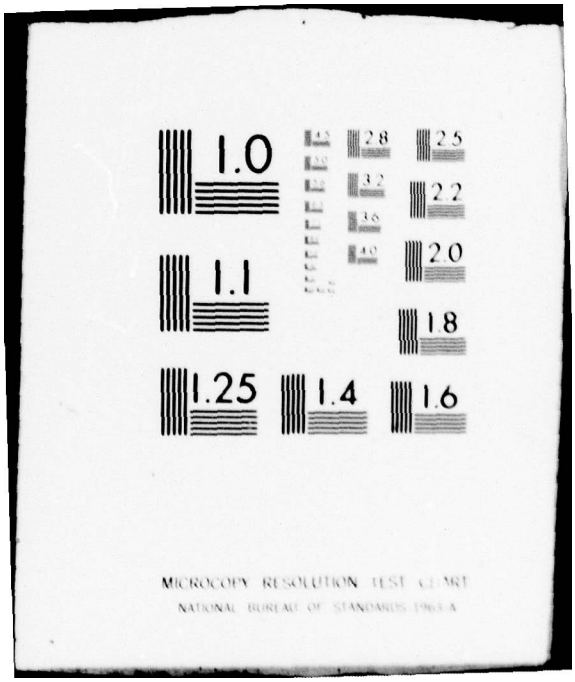
AD-A074 263 ARMY ARMAMENT RESEARCH AND DEVELOPMENT COMMAND ABERD--ETC F/G 20/11
SHOCK PROPAGATION IN THE THREE-DIMENSIONAL LATTICE. II. METHOD --ETC(U)
JUN 79 J D POWELL, J H BATTEH

UNCLASSIFIED ARBRL-TR-02175 SBIE-AD-E430 290 NL

| OF |
AD A074263

END
DATE FILMED
10-79
DDC


1.0
1.1
1.25
1.4
1.6
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.8
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963-A

LEVEL II

TECHNICAL REPORT ARBRL-TR-02175

SHOCK PROPAGATION IN THE THREE-DIMENSIONAL LATTICE. II. METHOD OF CALCULATION

John D. Powell
Jad H. Batteh

June 1979

US ARMY ARMAMENT RESEARCH AND DEVELOPMENT COMMAND
BALLISTIC RESEARCH LABORATORY
ABERDEEN PROVING GROUND, MARYLAND

Destroy this report when it is no longer needed.
Do not return it to the originator.

Secondary distribution of this report by originating or sponsoring activity is prohibited.

Additional copies of this report may be obtained from the National Technical Information Service, U.S. Department of Commerce, Springfield, Virginia 22161.

The findings in this report are not to be construed as an official Department of the Army position, unless so designated by other authorized documents.

The use of trade names or manufacturers' names in this report does not constitute indorsement of any commercial product.

AD74347

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

(12) 61 p.

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

(9) 1. REPORT NUMBER: Technical Report ARBRL-TR-02175

2. GOVT ACCESSION NO.

3. RECIPIENT'S CATALOG NUMBER

(6) 4. TITLE (and Subtitle): Shock Propagation in the Three-Dimensional Lattice. II. Method of Calculation.

5. TYPE OF REPORT & PERIOD COVERED: BRL Report

6. PERFORMING ORG. REPORT NUMBER

(10) 7. AUTHOR(s): John D. Powell and Jad H. Batteh

8. CONTRACT OR GRANT NUMBER(s)

9. PERFORMING ORGANIZATION NAME AND ADDRESS
US Army Ballistic Research Laboratory
ATTN: DRDAR-BLB
Aberdeen Proving Ground, MD 21005

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
(16) RDT&E 1L161102AH43

11. CONTROLLING OFFICE NAME AND ADDRESS
US Army Armament Research & Development Command
US Army Ballistic Research Laboratory
ATTN: DRDAR-BL, APG, MD 21005

(11) 12. REPORT DATE: JUNE 1979

13. NUMBER OF PAGES: 61

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)

15. SECURITY CLASS. (of this report): Unclassified

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

(14) ARBRL-TR-02175

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

(18) SBIE (19) AD-E430 290

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)
Shock propagation, computer molecular dynamics, lattice dynamics, nonequilibrium phenomena, solitary waves, solitons.

20. ABSTRACT (Continue on reverse side if necessary and identify by block number) (hmn)

A computer-molecular-dynamic (CMD) technique for simulating shock propagation in a three-dimensional, face-centered-cubic lattice has been developed. The first report in this sequence described the model used and the results of the calculation. This report contains the details of the calculation, a listing of the computer code, definitions of major symbols, and a brief description of the subprograms.

393 471

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

TABLE OF CONTENTS

1. INTRODUCTION

In the first report in this sequence[1] we presented results of numerical calculations in which we simulated shock propagation in a three-dimensional, face-centered-cubic (FCC) lattice. The details of the calculation were not discussed in order that the reader not be lost in a welter of detail, superfluous to understanding the significance of the results. In this report we describe more fully the calculations undertaken and the computer program used. The material herein is intended primarily for those who may wish to perform similar calculations.

We begin in Section 2 by describing the lattice used in the calculations. Included in the description are the labeling convention used to identify a particular atom, the method for finding its nearest neighbors, the method for calculating its potential interaction with other atoms in the lattice, and the method for calculating the lattice constant. In Section 3, we discuss how initial conditions of the lattice (prior to shock compression) are determined and how the shock wave affects the assumed periodicity of the lattice. The differential equations of motion satisfied by the atoms are also written down. In Section 4, we describe the thermodynamic variables (pressure, temperature, flow velocity, and density) calculated in the program and explain how the r calculation is carried out. The Appendix discusses the computer program used and provides a listing.

2. DESCRIPTION OF THE LATTICE

The model for the simulations consists of an infinite, three-dimensional, FCC lattice constructed of cells such as that shown in Figure 1. The cube edge, a_o, is hereafter referred to as the lattice constant.

Prior to compression, the crystal is assumed to obey periodic boundary conditions in all three Cartesian directions. Therefore, if a particle is located at point (x,y,z), there is a corresponding particle at the position $(x + \ell L_x a_o,\ y + m\, L_y a_o,\ z + n\, L_z a_o)$ where L_x, L_y and L_z are positive integers which define the periodicity of the lattice in the three directions, and ℓ, m, n are arbitrary integers. Furthermore, for any function F which depends on the particles' velocities and their displacements from equilibrium, we have

$$F(x+ \ell L_x a_o,\ y+mL_y a_o,\ z+nL_z a_o) = F(x,y,z)\ . \qquad (2.1)$$

1. J.D. Powell and J.H. Batteh, "Shock Propagation in the Three-Dimensional Lattice. I. Model and Results", BRL Report (to be published concurrently with this report).

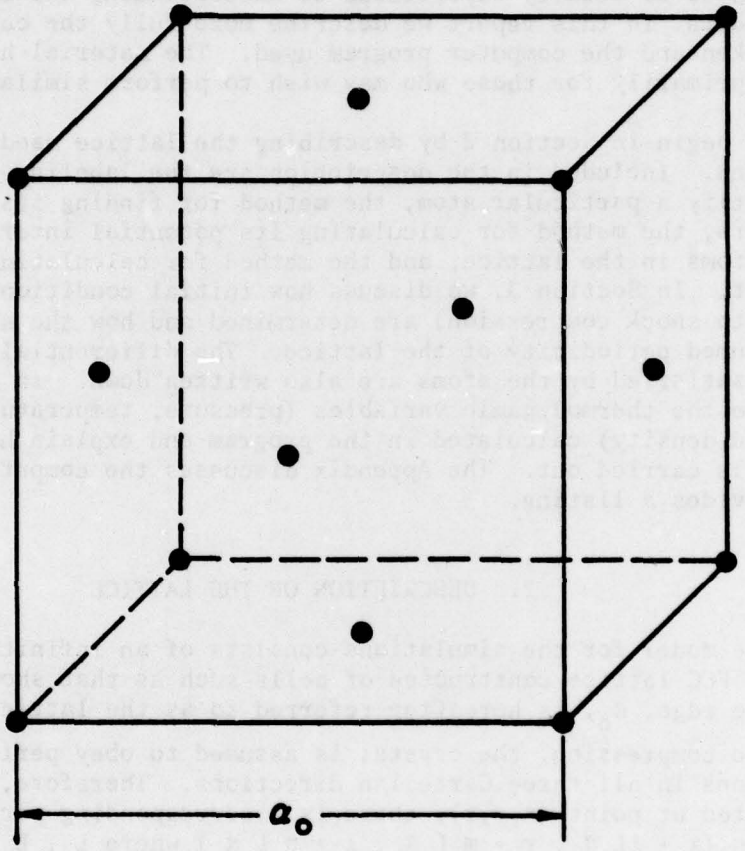


Figure 1. Cell of face-centered-cubic lattice.

It is obvious from Eq. (2.1) that, if the lattice is periodic, it is completely specified by the displacements and velocities of all atoms within a given period. We will henceforth refer to the period made up of atoms whose equilibrium positions satisfy the relations

$$\begin{aligned} 0 \leq x < L_x a_o \\ 0 \leq y < L_y a_o \\ 0 \leq z < L_z a_o \end{aligned} \tag{2.2}$$

as the "primary" lattice.

2.1 Location of Equilibrium Positions and Labeling Convention.

The crystal axes will be assumed to be coincident with the coordinate system. If the lattice is at equilibrium, then, it is evident that a plane of atoms located at $z=na_o$, where n is an integer, is oriented with respect to the x and y axes as shown in Figure 2a; a plane located at $z = (n+\frac{1}{2})a_o$, on the other hand, is oriented as shown in Figure 2b. For purposes of calculation it is most convenient to label each atom by a single set of indices (i,j,k) and to do so we adopt the following convention: Atoms in the first plane, located at z=0, whose x and y components are integral multiples of a_o will be labeled by k=1; those whose x and y components are half-integral multiples of a_o will be labeled by k=2; atoms in the second plane, located at $z=a_o/2$, whose x components are half-integral multiples of a_o and whose y components are integral multiples of a_o, will be labeled by k=3; those whose x components are integral multiples of a_o and whose y components are half-integral multiples of a_o are labeled by k=4. The process is then repeated so that atoms in the third plane, located at $z=a_o$, whose x and y components are integral multiples of a_o are labeled by k=5 and so forth. Thus, in general, the plane in which particle (i,j,k) lies is given by

$$n_{p\ell} = \left[\frac{k-1}{2}\right] + 1 \tag{2.3}$$

where the brackets denote integer division and the number of atoms in such a plane of the primary lattice is $2L_xL_y$. The value of i for a particle located at x=0 or $x=a_o/2$ is unity and values increase along the positive x axis. Similar considerations apply for the index j. This convention is not the only conceivable one, but does appear to be

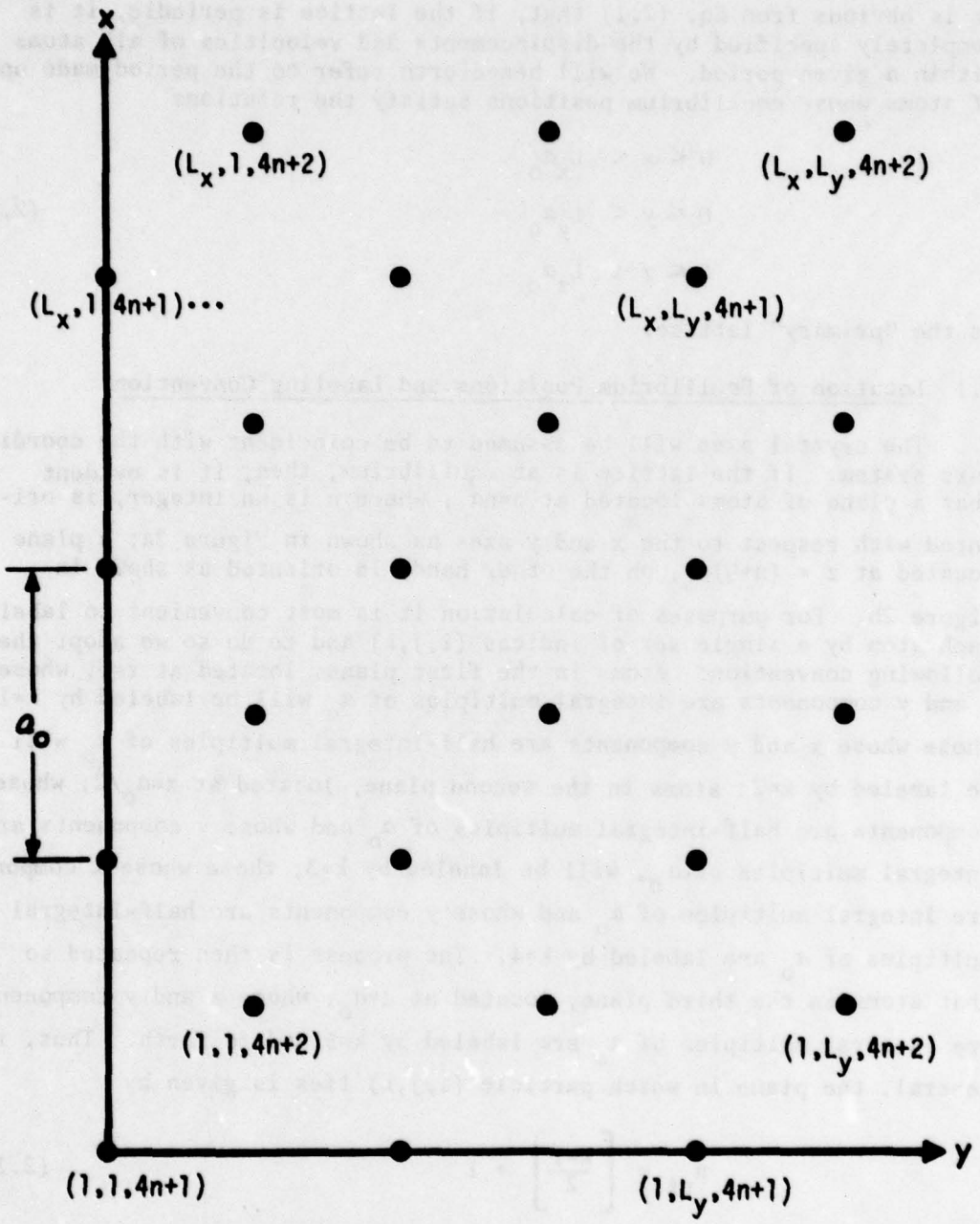


Figure 2a. Labeling scheme for the plane of atoms located at $z=na_o$. (For convenience, $L_x=4$, $L_y=3$.)

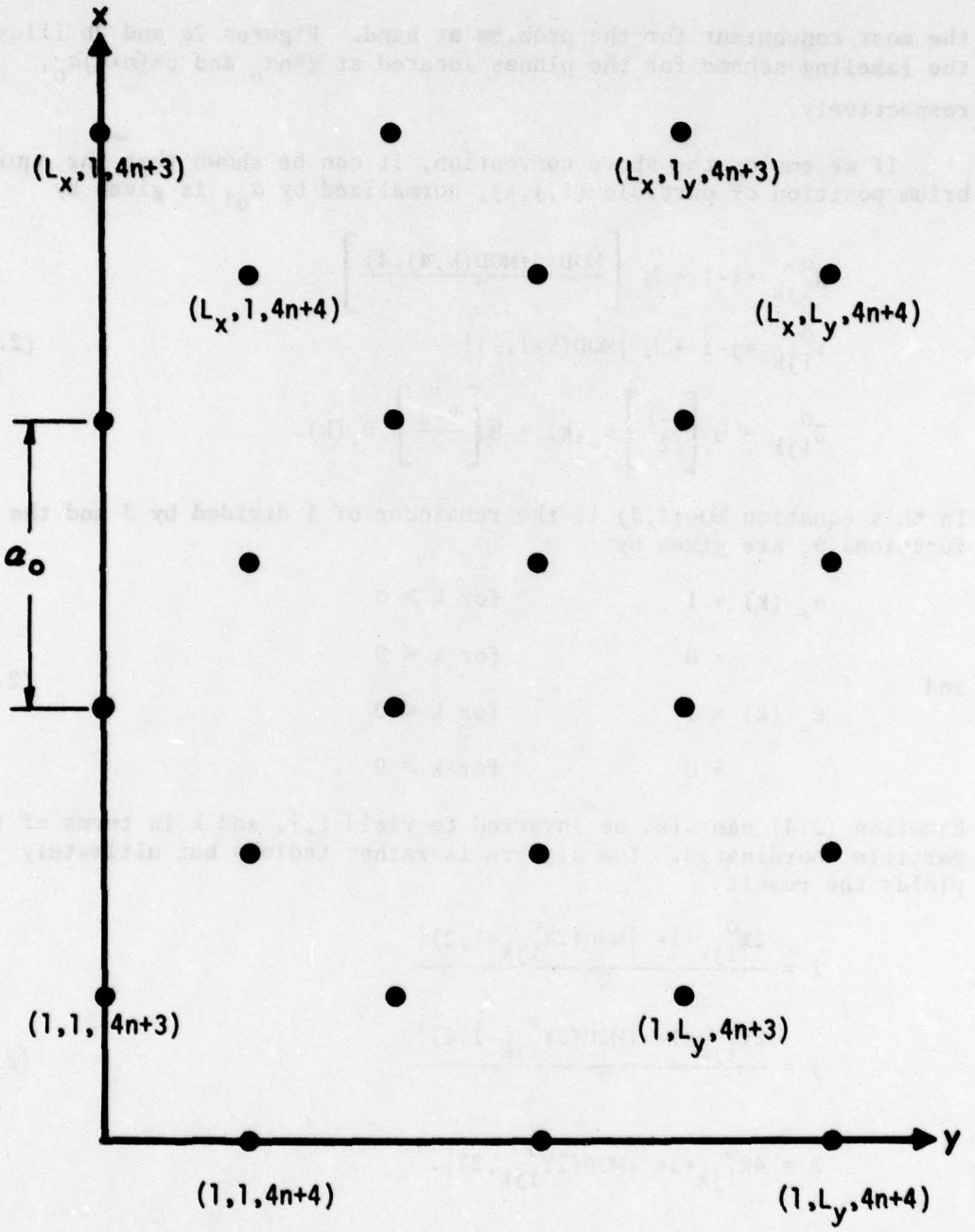


Figure 2b. Labeling scheme for the plane of atoms located at $z=(n+1/2)a_o$. (For convenience, $L_x=4, L_y=3$.)

the most convenient for the problem at hand. Figures 2a and 2b illustrate the labeling scheme for the planes located at $z=na_o$ and $z=(n+\frac{1}{2})a_o$, respectively.

If we employ the above convention, it can be shown that the equilibrium position of particle (i,j,k), normalized by a_o, is given by

$$\begin{aligned} X^o_{ijk} &= i-1 + \tfrac{1}{2}\left[\frac{\text{MOD}(4+\text{MOD}(k,4),4)}{2}\right] \\ Y^o_{ijk} &= j-1 + \tfrac{1}{2}\,|\text{MOD}(k-1,2)| \\ Z^o_{ijk} &= \tfrac{1}{2}\left[\frac{k-1}{2}\right]\theta_+(k) + \tfrac{1}{2}\left[\frac{k-2}{2}\right]\theta_-(k). \end{aligned} \tag{2.4}$$

In this equation MOD(I,J) is the remainder of I divided by J and the functions $\theta_\pm$ are given by

$$\begin{aligned} \theta_+(k) &= 1 \qquad \text{for } k > 0 \\ &= 0 \qquad \text{for } k \leq 0 \end{aligned}$$

and

$$\begin{aligned} \theta_-(k) &= 1 \qquad \text{for } k \leq 0 \\ &= 0 \qquad \text{for } k > 0 \ . \end{aligned} \tag{2.5}$$

Equation (2.4) can also be inverted to yield i,j, and k in terms of the particle coordinates. The algebra is rather tedious but ultimately yields the result

$$\begin{aligned} i &= \frac{2X^o_{ijk}+1+\,|\text{MOD}(2X^o_{ijk}-1,2)|}{2} \\ j &= \frac{2Y^o_{ijk}+1+\,|\text{MOD}(2Y^o_{ijk}-1,2)|}{2} \\ k &= 4Z^o_{ijk}+1+\,|\text{MOD}(2Y^o_{ijk},2)|. \end{aligned} \tag{2.6}$$

2.2 Relative Coordinates of Neighbors.

For all reasonable interatomic-force models, the force exerted on a particular atom by the remaining atoms in the lattice decreases rapidly with increasing separation distance. Consequently, when computing forces, it is necessary to consider only atoms in the immediate vicinity of the atom in question. Therefore, given an atom (i,j,k), it is

necessary to determine the values (i', j', k') for all its significant neighbors in order to know which particles exert an appreciable force on the atom.

The simplest method for determining an atom's neighbors is to note that, in the equilibrium lattice, the relative coordinates from atom (i,j,k) to one of its neighbors are independent of i,j, and k since the lattice possesses translational symmetry. For convenience, then, we consider atom (0,0,1) and note from Eq. (2.4) that the relative coordinates to atom (i', j', k') are given by

$$\Delta X^o = i' + \tfrac{1}{2}\left[\frac{\text{MOD}(4+\text{MOD}(k',4),4)}{2}\right]$$

$$\Delta Y^o = j' + \tfrac{1}{2}\,|\text{MOD}(k'-1,4)| \qquad (2.7)$$

$$\Delta Z^o = \tfrac{1}{2}\left[\frac{k'-1}{2}\right]\theta_+(k') + \tfrac{1}{2}\left[\frac{k'-2}{2}\right]\theta_-(k').$$

By letting i', j', k' vary over several values near i,j, and k (say, from -10 to 10) one can then use Eq. (2.7) to determine the relative coordinates to various neighbors.

A short computer program has been written which performs the above calculation and the results for the first five sets of nearest neighbors are shown in Table I. The first column in the table indicates which set of neighbors is being considered and the last column the number of atoms contained within the set (obtained by taking all possible combinations of sign for the relative coordinates). Column 5 indicates the distance between the two atoms given by

$$\text{DIST} = \left[(\Delta X^o)^2 + (\Delta Y^o)^2 + (\Delta Z^o)^2\right]^{\frac{1}{2}} \qquad (2.8)$$

Once i,j and k are given and the relative coordinates determined, values of i', j', and k' can then be found from Eq. (2.6).

2.3 Interatomic Potential and Lattice Constant.

In all our calculations we will assume that the atoms in the lattice interact through a Morse-type potential. Therefore, for a pair of isolated atoms the potential energy, normalized by the dissociation energy, D, of the pair, is given by

$$\phi = \left[e^{-R(r/r_o-1)} - 1\right]^2 . \qquad (2.9)$$

In this expression r is the separation distance of the pair, r_o the separation at the minimum of the potential, and R is a dimensionless parameter which is indicative of the nonlinearity of the potential.

TABLE I. Relative Coordinates of Neighbors of a Particular Atom in the FCC Lattice.

Neighbors	ΔX^o	ΔY^o	ΔZ^o	DIST	Number
1st nearest	$\pm\frac{1}{2}$	$\pm\frac{1}{2}$	0	$1/\sqrt{2}$	12
	$\pm\frac{1}{2}$	0	$\pm\frac{1}{2}$		
	0	$\pm\frac{1}{2}$	$\pm\frac{1}{2}$		
2nd nearest	± 1	0	0	1	6
	0	± 1	0		
	0	0	± 1		
3rd nearest	± 1	$\pm\frac{1}{2}$	$\pm\frac{1}{2}$	$\sqrt{3/2}$	24
	$\pm\frac{1}{2}$	± 1	$\pm\frac{1}{2}$		
	$\pm\frac{1}{2}$	$\pm\frac{1}{2}$	± 1		
4th nearest	0	± 1	± 1	$\sqrt{2}$	12
	± 1	0	± 1		
	± 1	± 1	0		
5th nearest	$\pm 3/2$	$\pm\frac{1}{2}$	0	$\sqrt{5/2}$	24
	$\pm 3/2$	0	$\pm\frac{1}{2}$		
	0	$\pm 3/2$	$\pm\frac{1}{2}$		
	0	$\pm\frac{1}{2}$	$\pm 3/2$		
	$\pm\frac{1}{2}$	$\pm 3/2$	0		
	$\pm\frac{1}{2}$	0	$\pm 3/2$		

When the atom pair is contained within a lattice the situation is more complicated because the atoms interact not only with one another, but also with the remaining atoms of the lattice. As noted previously, for purposes of computation it is most convenient to neglect the potential interaction for atoms sufficiently separated that their forces of interaction are negligible. Therefore, for a lattice in which all atoms are at rest in their equilibrium positions, we have

for the potential energy of atom (i,j,k)

$$\Phi_{i,j,k} = {}^{1}/_{2} \sum_{\ell,m,n}{}' \left[e^{-R(|\vec{r}^{\,o}_{ijk} - \vec{r}^{\,o}_{\ell mn}|/r_o - 1)} - 1 \right]^2 . \qquad (2.10)$$

In Eq. (2.10) the prime indicates that the sum runs only over atoms (ℓ,m,n) which are close enough to atom (i,j,k) to contribute significantly to the force and $\vec{r}^{\,o}_{ijk}$ denotes the equilibrium position of atom (i,j,k). The factor one-half is included so as not to count interactions twice when computing the total potential of the lattice.

If we define a nondimensional position vector according to

$$\vec{R}^{o}_{ijk} = \vec{r}^{\,o}_{ijk}/a_o \qquad (2.11)$$

the expression becomes

$$\Phi_{ijk} = 1/2 \sum_{\ell,m,n}{}' \left[e^{-R(A_o|\vec{R}^{o}_{ijk} - \vec{R}^{o}_{\ell mn}|-1)} - 1 \right]^2 \qquad (2.12)$$

where

$$A_o = a_o/r_o \qquad (2.13)$$

is the lattice constant normalized by the equilibrium separation for two isolated particles. The data in Table I can then be used to calculate the potential of particle (i,j,k) assuming up to fifth-nearest-neighbor interactions.

In order to generalize Eq. (2.12) to include the case in which the atoms are not in their equilibrium positions, we define a critical radius R_c such that interactions between atoms separated by a distance greater than R_c are negligible. The appropriate expression for the potential then becomes

$$\Phi_{ijk} = 1/2 \sum_{\ell,m,n} \left[e^{-R(A_o|\vec{R}_{ijk} - \vec{R}_{\ell mn}|-1)} - 1 \right]^2 \qquad (2.14)$$

$$|\vec{R}_{ijk} - \vec{R}_{\ell mn}| < R_c$$

where the sum extends over all neighbors which lie within a sphere of radius R_c centered at atom (i,j,k). The zero superscripts have been removed from $\vec{R}_{ijk}$ and $\vec{R}_{\ell mn}$ to denote that these vectors do not necessarily refer to the equilibrium positions of the atoms.

Equation (2.12) can be used to calculate the nondimensional lattice constant A_o in terms of R by noting that in the equilibrium configuration the lattice will readjust itself so as to minimize the potential with respect to A_o. Therefore, we must have

$$\frac{d\Phi_{ijk}}{dA_o} = -R \sum_{\ell,m,n}' |\vec{R}_{ijk} - \vec{R}_{\ell mn}| \left[e^{-2R(A_o|\vec{R}^o_{ijk} - \vec{R}^o_{\ell mn}| - 1)} - e^{-R(A_o|\vec{R}^o_{ijk} - \vec{R}^o_{\ell mn}| - 1)} \right] = 0 \quad . \tag{2.15}$$

A computer program has been written which solves this equation numerically for A_o as a function of R and the number of neighbors taken in the sum. Results are shown in Table II. The values of N listed across the top row indicate that the sum in Eq. (2.15) included all atoms through Nth nearest neighbors and values of R are listed in the first column.

TABLE II. Values of the Lattice Constant A_o as a Function of R and Number of Neighbors.

R \ N	1	2	3	4	5	6	7	8	9	10
.5	1.4142	1.2350	.9416	.8715	.7734	.7472	.6465	.6375	.5905	.5662
1.0	1.4142	1.2629	.9775	.9065	.8070	.7810	.6764	.6671	.6186	.5933
2.0	1.4142	1.3138	1.0827	1.0158	.9204	.8960	.7875	.7773	.7255	.6975
3.0	1.4142	1.3522	1.2170	1.1768	1.1227	1.1099	1.0523	1.0467	1.0207	1.0066
4.0	1.4142	1.3774	1.3172	1.3034	1.2898	1.2874	1.2796	1.2791	1.2772	1.2764
5.0	1.4142	1.3927	1.3681	1.3640	1.3610	1.3606	1.3596	1.3596	1.3594	1.3594
6.0	1.4142	1.4016	1.3916	1.3904	1.3897	1.3896	1.3895	1.3895	1.3895	1.3894
7.0	1.4142	1.4068	1.4027	1.4023	1.4021	1.4021	1.4021	1.4021	1.4021	1.4021

From the table it is evident that for small values of R the potential is extremely long-range. For example, for R less than about 3.0, A_o has not converged even though all atoms through tenth-nearest neighbors were employed in the calculation. Clearly the range of the potential extends beyond this point. As R increases, however, the range of the potential decreases significantly. For example, when R = 6.0, A_o has nearly converged to its asymptotic value when third-nearest neighbors are employed in the calculation. Obviously, the

higher-order neighbors contribute negligibly to the force. As $R \to \infty$, A_o approaches $\sqrt{2}$ because, in that case, the distance between nearest neighbors is r_o and from Table I

$$r_{o/a_o} = 1/\sqrt{2} \tag{2.16}$$

or $A_o = \sqrt{2}$.

3. GENERATION OF SHOCK WAVE

3.1 Initial Conditions and Method of Compression.

Before the lattice described in the previous section can be subjected to shock compression, the initial displacements and velocities of the atoms in the primary lattice must be specified. In this subsection, we will discuss the method for determining these initial conditions.

The most reasonable assumption for the initial state of the lattice is that it is in thermal equilibrium at temperature T. Therefore the velocities of the atoms within the lattice must be distributed according to a Maxwellian distribution.

Let us denote by $\vec{V}_{ijk}$ the velocity of particle (i,j,k), normalized to $\sqrt{D/m}$, where D is the dissociation energy for an isolated pair of atoms and m is the mass of the particle. Then the probability that the α^{th} Cartesian component of $\vec{V}_{ijk}$ lies between V and V+dV is given by the distribution function

$$f = \left(\frac{\gamma}{2\pi} \right)^{3/2} e^{-\gamma V^2/2} dV. \tag{3.1}$$

In Equation (3.1), γ is the ratio of the dissociation energy to the thermal energy, viz.,

$$\gamma = \frac{D}{k_B T} \tag{3.2}$$

where T is the absolute temperature and k_B is Boltzmann's constant. Furthermore, the total energy within the lattice is a constant of the motion and, if the lattice were harmonic, would be given by $3nk_BT$ where n is the total number of atoms in the lattice.

We made use of this information to obtain a set of initial conditions for the lattice in the following manner: First, a set of 3 n random numbers was generated from the distribution function in Eq. (3.1). These values were assigned as components of the velocity to each of the n particles in the primary lattice. An increment of velocity $d\vec{V}$ was then added to the velocity of each particle so that the condition

$$\sum_{i,j,k} \vec{V}_{ijk} = 0 \tag{3.3}$$

was satisfied. This insures that no net momentum is imparted to the lattice. We then noted that, even though the lattice was slightly anharmonic, its total energy in thermal equilibrium was still nearly equal to $3nk_BT$. In order to provide the lattice with this energy, we assigned the particles to their equilibrium positions and scaled the velocities by a common factor so as to be consistent with the total energy, i.e.,

$$\sum_{i,j,k} V_{ijk}^2 = {}^{6n}/_{\gamma} \, . \tag{3.4}$$

We then allowed the particles in the lattice to oscillate freely. Periodically, the velocities were sampled and their distribution plotted to demonstrate that the lattice was indeed in equilibrium and that it remained so. Figure 3 shows a typical result for a lattice with $L_x=L_y=L_z=4$ and n=256. Plotted on the vertical axis is the fraction of the total number of atoms whose velocity component V_α lies within the interval given by the horizontal axis. The Maxwellian character of the graphs is apparent and the general shapes were found to be constant in time.

When an appropriate set of initial conditions has been generated, the lattice is subjected to shock compression along the z axis. We assume that the shock wave does not disturb the periodicity of the lattice in the planes transverse to the axis of compression. Therefore, the x and y dimensions of the lattice for the shock wave calculations are taken to be the same as those of the primary lattice. It is clear, however, that the compression will disturb the periodicity in the z direction. Therefore, the calculation is performed as follows:

Beginning at z=0, we construct a semi-infinite lattice at temperature T from a series of segments whose initial conditions are identical to those of the primary lattice. Each segment contains $2L_z$ planes of atoms normal to the z axis, as shown in Figure 4. At time τ=0, the first plane is subjected to steady compression by assigning a velocity U_p in the positive z direction to each particle in that plane. This velocity is impressed upon the particles in the first plane throughout the calculation, thereby generating a shock wave which propagates along the z axis.

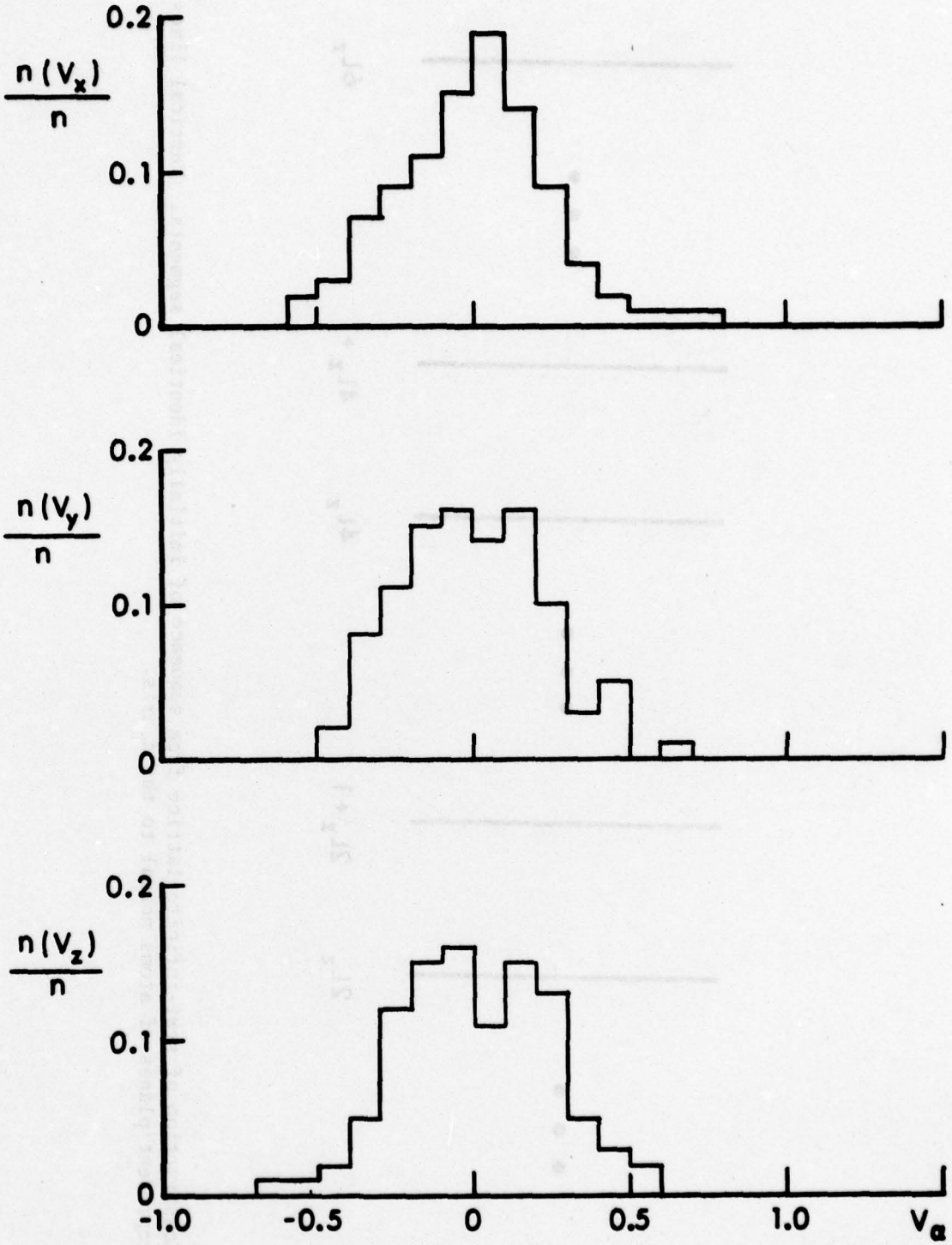


Figure 3. Velocity distribution function for equilibrium lattice.

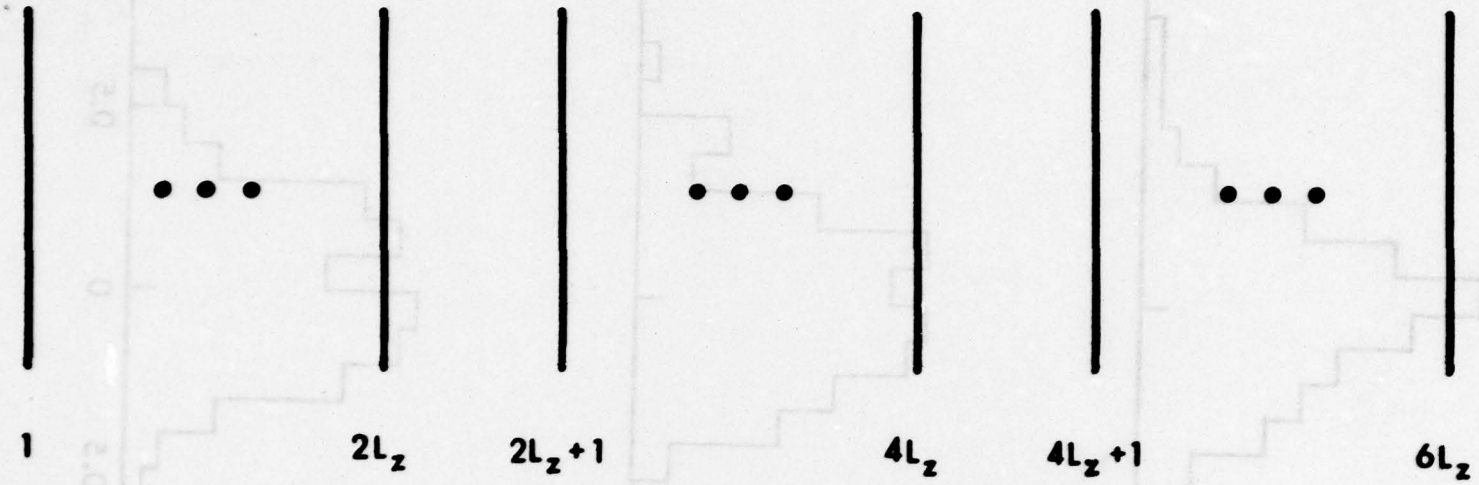


Figure 4. Construction of semi-infinite lattice from sequence of initially identical segments. Vertical lines represent planes of atoms normal to the z axis.

We now consider the j^{th} plane assumed to be located in the first segment. Before the shock front reaches this plane, the motion of a particular atom contained within it is the same as the motion of the corresponding atom in the $j+2L_z^{th}$ plane. The identical motion results because these atoms had the same initial conditions and were acted upon by the same forces. Working our way backwards from the j^{th} plane, we locate the first plane whose motion differs appreciably from that of the corresponding plane in the second segment. Thereby, we determine precisely the location of the shock front at any time. Furthermore, until the shock reaches the $2L_z^{th}$ plane, it is necessary to solve the equations of motion for only particles in the first two segments since the particles in all the remaining segments will have trajectories identical to the corresponding particles in the second segment. When the shock front nears the $2L_z^{th}$ plane, particles in the third segment are included in the computation. The shock front is located in the same manner as before and the process repeated as often as necessary to complete the calculation. It is therefore necessary to monitor at most $4L_z$ planes of particles ahead of the shock front at any time, the last $2L_z$ representing equilibrium conditions ahead of the shock[2].

3.2 Equations of Motion.

The preceding subsection contains a qualitative description of how the lattice is subjected to shock compression. In this subsection the equations of motion to be solved numerically for each atom in the lattice will be determined. As has been observed before, it will be most convenient to use nondimensional variables in all calculations. For convenience, Table III lists the normalizing factor for various quantities of interest. Hereafter, reference to these quantities will imply their nondimensional values.

The force exerted on particle (i,j,k) by the remaining particles in the lattice can be determined from the gradient of Eq. (2.14) with respect to $\vec{R}_{ijk}$. If we normalize the force by $2RD/r_o$ (see Table III) and employ nondimensional units for the time and for $\vec{R}_{ijk}$, the equation of motion for the particle becomes

$$\ddot{\vec{R}}_{ijk} = 2A_o R\, \vec{F}_{ijk} \tag{3.5}$$

2. We are indebted to D.H. Tsai for suggesting this method of calculation to us.

TABLE III. Normalizing Factors.

Quantity	Symbol	Normalizing Factor
Velocity of particle (i,j,k)	$\vec{V}_{ijk}$	$\sqrt{D/m}$
Nondimensional lattice constant	A_o	r_o
Position of particle (i,j,k)	$\vec{R}_{ijk}$	a_o
Potential energy of particle (i,j,k)	Φ_{ijk}	D
Kinetic energy of particle (i,j,k)	T_{ijk}	D
Density	ρ	ρ_o
Force exerted on particle (i,j,k)	$\vec{F}_{ijk}$	$2RD/r_o$
Time	τ	$\sqrt{m/D}\, a_o$
Element of stress tensor	$\sigma_{\alpha\beta}$	$4D/a_o^3$
Compression velocity	U_p	$\sqrt{D/m}$
Temperature	θ	D/k_B
Dissociation energy	D	
Mass of particle	m	
Equilibrium spacing for isolated pair of atoms	r_o	
Lattice constant	a_o	
Initial density ahead of shock	ρ_o	

Here, $\vec{F}_{ijk}$ is the nondimensional force exerted on the particle given explicitly by

$$\vec{F}_{ijk} = \sum_{\substack{\ell,m,n \\ |\vec{R}_{ijk}-\vec{R}_{\ell mn}|< R_c}} \left[e^{-2R(A_o|\vec{R}_{ijk}-\vec{R}_{\ell mn}| -1)} - e^{-R(A_o|\vec{R}_{ijk}-\vec{R}_{\ell mn}| -1)} \right] \times \frac{\vec{R}_{ijk}-\vec{R}_{\ell mn}}{|\vec{R}_{ijk}-\vec{R}_{\ell mn}|} \tag{3.6}$$

and each dot represents differentiation with respect to the dimensionless time, τ, related to the real time, t, by

$$\tau = \frac{t}{\sqrt{m/D}\, a_o} . \tag{3.7}$$

The sum in Eq. (3.6) runs only over values of ℓ,m, and n such that $|\vec{R}_{ijk} - \vec{R}_{\ell mn}| < R_c$.

For purposes of computation it is most convenient to convert Eqs. (3.5) into a set of first-order equations. This transformation can be accomplished by noting that the velocity of particle (i,j,k) is just the time derivative of the displacement. Equation (3.5) becomes, therefore,

$$\begin{aligned} \dot{\vec{V}}_{ijk} &= 2A_o R\, \vec{F}_{ijk} \\ \dot{\vec{R}}_{ijk} &= \vec{V}_{ijk} . \end{aligned} \tag{3.8}$$

We have, of course, twice the original number of equations.

For the equilibrium lattice, Eqs. (3.8) apply to every particle in the lattice. For the lattice undergoing shock compression, however, particles in the first plane are not acted on by any net force in the z direction. Therefore, we have

$$\begin{aligned} \left(\dot{\vec{V}}_{ijk}\right)_z &= 0 \\ & \qquad k = 1,2 \\ \left(\dot{\vec{R}}_{ijk}\right)_z &= U_p \end{aligned} \tag{3.9}$$

where U_p is the constant compression velocity and the subscript z denotes the component in the z direction. For the remaining particles in the lattice, Eqs. (3.8) apply.

In order to solve Eqs. (3.8) and (3.9), we employed a fourth-order Runge-Kutta technique[3]. To check the accuracy of the code and the appropriateness of the step size used we calculated the work done on the

3. B. Carnahan, H.A. Luther, and J.O. Wilkes, Applied Numerical Methods (Wiley, NY, 1969), Chap. 6.

lattice by the compression force and compared it at various times with the increase in the total energy of the lattice. For most calculations, a step size of the order .05/R was found to be acceptable.

4. THERMODYNAMIC VARIABLES

Once Eqs. (3.8) have been solved numerically to determine $\vec{V}_{ijk}$ and $\vec{R}_{ijk}$, the results can be used to calculate macroscopic quantities of interest in the lattice. These thermodynamic quantities vary with position and therefore the averages from which they are calculated are taken only over finite regions of the crystal. Hereafter, we will refer to such regions as "thermodynamic regions". We will assume that the regions are sufficiently small that the variable of interest varies negligibly throughout the region, and yet sufficiently large that the averages are meaningful and boundary effects negligible. For all thermodynamic variables, standard statistical-mechanical definitions will be employed and ensemble averages will be replaced by spatial averages. In the following discussion, we will assume that the thermodynamic region over which the average is taken begins just before some odd-numbered plane (normal to the z axis) and ends just before a later odd-numbered plane. Respective values of k are k_{min} and k_{max}. In the y and z directions the region is defined by the same bounds as the primary lattice. Therefore, in calculating averages in a thermodynamic region we include those particles whose indices lie within

$$\begin{aligned} 1 \leq i \leq L_x \\ 1 \leq j \leq L_y \\ k_{min} \leq k \leq k_{max} \end{aligned} \qquad (4.1)$$

The total number of atoms which satisfy conditions (4.1) is given by

$$n_R = (k_{max} - k_{min} + 1)\, L_x L_y. \qquad (4.2)$$

4.1 Average Velocity.

The shock will induce a flow in the direction of its propagation and, thus, it is necessary to calculate the average velocity as a function of position in the lattice. For the thermodynamic region specified by Eq. (4.1) the appropriate expression is

$$\langle \vec{V} \rangle = \frac{1}{n_R} \sum_{i=1}^{L_x} \sum_{j=1}^{L_y} \sum_{k=k_{min}}^{k_{max}} \vec{V}_{ijk} = \frac{1}{n_R} \sum_{R} \vec{V}_{ijk} \qquad (4.3)$$

where $\sum_R$ is a short-hand representation indicating that the sum extends only over the atoms contained in the region. One would expect that only $\langle V_z \rangle$ would be nonzero.

4.2 Density.

Prior to the time the lattice is compressed by the shock wave, its density is a constant and equal to $4/a_o^3$. This value can be obtained from Figure 1 if we note that one-eighth of each particle at the corners of the cube and one-half of each particle at the faces of the cube are contained within the volume of the cube. Compression by the shock does not affect the average separation between atoms in the x and y directions, but obviously does in the z direction. Thus, the final density, normalized by the original density, can be obtained by calculating the final separation along the z axis of planes associated with k_{min} and k_{max} and dividing by their equilibrium separation. In the dynamic case, the z coordinate of a plane will be defined as the mean z component of the atoms within it. Consequently, from Eq. (2.3) the initial separation of the planes defining the region in question is

$$\Delta Z_i = 0.5 \left\{ \left[\frac{k_{max}^{-1}}{2} \right] - \left[\frac{k_{min}^{-1}}{2} \right] \right\} , \tag{4.4}$$

and the final separation is

$$\Delta Z_f = \frac{1}{2L_x L_y} \sum_{i=1}^{L_x} \sum_{j=1}^{L_y} \left\{ Z_{i,j,k_{max}} + Z_{i,j,k_{max}-1} - Z_{i,j,k_{min}} - Z_{i,j,k_{min}+1} \right\} . \tag{4.5}$$

The final density, therefore, is

$$\rho = \frac{\Delta Z_i}{\Delta Z_f} . \tag{4.6}$$

4.3 Temperature.

The usual definition of temperature, normalized by the factor D/k_B, is given by the expression

$$\theta = \frac{1}{3n_R} \sum_R \left[\vec{V}_{ijk} - \langle \vec{V} \rangle \right]^2 . \tag{4.7}$$

It is also convenient to define a "component" of the temperature in each Cartesian direction. A reasonable definition is

$$\theta_\alpha = \frac{1}{3n_R} \sum_R \left[(V_{ijk})_\alpha - \langle V_\alpha \rangle \right]^2 . \qquad (4.8)$$

4.4 Stress.

The $\alpha\beta$ component of the stress tensor in a solid is defined as the α^{th} component of the force acting per unit area on a plane normal to the β direction. Such forces arise both from the interactions acting across the plane as well as from transfer of momentum due to fluctuations in the velocity of the atoms about the mean velocity. For dilute gases, the latter term is dominant and represents the pressure tensor. For solids or dense gases, however, the potential contribution must be accounted for.

Irving and Kirkwood[4] have derived an expression for the ensemble-averaged pressure tensor for a system in which the potential energy is not negligible. For purposes of computer-molecular-dynamic calculations, we assume that the ensemble average can be approximated by a spatial average over the appropriate thermodynamic region. Provided, then, that the range of the interatomic forces is much less than the dimensions of the region in question, the kinetic and potential contributions to the pressure tensor become [see Reference 4, Eqs. (5.13) and (5.14)]

$$\overleftrightarrow{\sigma}_K = \frac{\rho}{n_R} \sum_R (\vec{V}_{ijk} - \langle \vec{V} \rangle)(\vec{V}_{ijk} - \langle \vec{V} \rangle) \qquad (4.9)$$

and

$$\overleftrightarrow{\sigma}_V = \frac{RA_o}{n_R} \sum_R \sum_{\ell,m,n} (\vec{R}_{ijk} - \vec{R}_{\ell mn}) \vec{F}_{ijk;\,\ell mn} \qquad (4.10)$$

where $\vec{F}_{ijk;\ell mn}$ is the force exerted on particle (i,j,k) by particle (ℓ,m,n). In transcribing the equations from Reference 4, we have employed our nondimensional units and normalized the stress by the factor $4D/\alpha_o^3$ (see Table III). The total stress, of course, is given by

$$\overleftrightarrow{\sigma} = \overleftrightarrow{\sigma}_k + \overleftrightarrow{\sigma}_V \quad . \qquad (4.11)$$

4. J.H. Irving and J.G. Kirkwood, "The Statistical Mechanical Theory of Transport Processes. IV. The Equations of Hydrodynamics", J. Chem. Phys. 18, 817 (1950).

APPENDIX

In this appendix we briefly describe the computer program which performs the calculations described previously. The description is followed by a list of the major symbols used in the code, their definitions, and the corresponding symbol used in the text. In Section A.2 we describe the function of each of the sixteen subprograms in the program. Finally, a complete listing of the code is presented in Section A.3.

The program consists of a main routine and sixteen subprograms which will be discussed in the following section. It is capable of calculating a set of initial conditions for the primary lattice prior to shock compression and of performing the shock-wave calculation. At the end of each run the velocities and positions of each atom as well as the time are written on a tape. Thus, by using these values as input data, a succeeding calculation can begin where a previous one was terminated. The execution time for the program is approximately 7×10^{-3} seconds per particle per time step on a CDC 7600.

The basic function of the program is to solve numerically the classical equations of motion for every atom in the lattice and, from their solution, to calculate a number of quantities of interest. Included in the possible output are the mean displacements and velocities of planes of atoms normal to the z axis at a particular time; the velocity-time trajectories of individual planes; the velocity distribution function and various thermodynamic quantities calculated in macroscopically small, microscopically large regions of the lattice (referred to as thermodynamic regions); and the velocity and displacement of each atom in the lattice at a given time.

A number of input data must be supplied to the main routine. These data are listed and described as follows:

LX (statement 10) - one-half the total number of planes, normal to the x axis, of the primary lattice

LY (statement 10) - one-half the total number of planes, normal to the y axis, of the primary lattice

LZ (statement 10) - one-half the total number of planes, normal to the z axis, whose atomic equations of motion must be solved in the current calculation

NMAX (statement 14) - the final set of equilibrium nearest neighbors to a particle which are scanned to determine whether they lie within the critical radius of potential interaction.

NMIN (statement 14) - the last set of equilibrium nearest neighbors to a particle which are assumed to lie within the critical radius of potential interaction. Therefore, if NMIN=2 and NMAX=3, for example, all particles which are first- or second-nearest neighbors to a particular particle when the lattice is in equilibrium are assumed to lie within the critical radius in the dynamic lattice; those

which are fourth- or higher-nearest neighbors in equilibrium are assumed to be outside the critical radius; those which are third-nearest neighbors are tested to determine whether they are within the critical radius.

RCRIT (statement 14) - the radius of the imaginary sphere drawn about each particle whose equations of motion are solved. Atoms lying within the sphere are assumed to exert a force on the given atom; those outside do not.

H(statement 15) - the time step employed in the Runge-Kutta scheme for numerically solving the equations.

TAUMAX (statement 15) - the time at which present calculation is to stop.

IPRINT (statement 16) - the frequency of printout of all output data except velocity-time trajectories of specific planes. That is, IPRINT time steps are executed between printouts.

R (statement 17) - anharmonicity factor in the Morse potential

RE (statement 17) - lattice constant, a_o, normalized by single-pair, equilibrium separation constant, r_o.

GAMMA (statement 18) - ratio of dissociation energy, D, in Morse potential and thermal energy, $k_B T$, of lattice.

ICALC (statement 21) - index which determines which of three calculations is to be performed by the code. For ICALC=1, the program will calculate random initial conditions for the atoms in the primary lattice prior to shock compression; for ICALC=2, the program reads this initial data from TAPE 7 and performs the first shock-wave calculation; for ICALC=3, the program reads from TAPE 7 the positions and velocities of all atoms in the lattice as well as the time at the end of the last shock-wave calculation, and extends the shock-wave calculation to TAUMAX.

K21 (statement 22) - the number of planes, normal to the z axis, which make up a thermodynamic region. Thus if K21=8, for instance, thermodynamic properties of the lattice will be calculated for planes 1-8, 9-16, etc.

The remaining parameters are input data only for ICALC≠1:

LZLAST (statement 56) - one-half the number of planes, normal to the z axis, contained in the previous calculation. This will differ from LZ if one must increase the size of the lattice to perform the present shock-wave calculation (see discussion Section 3.1).

LZSEG (statement 56) - one-half the number of planes, normal to the z axis, contained in the primary lattice.

NSEG (statement 56) - the number of segments equal in size to the primary lattice which must be added to the lattice of the previous calculation to carry out the current calculation, i.e., LZ=LZLAST + NSEG*LZSEG.

UP (statement 57) - compression velocity (normalized by $\sqrt{D/m}$).

NPMAX (statement 58) - the total number of planes whose velocity-time trajectories are to be printed.

KPRINT (statement 58) - frequency of printout for velocity-time trajectories of particular planes .

NUMPL (I) (statement 61) - number of particular plane whose velocity-time trajectory is to be printed out ($1 \leq I \leq$ NPMAX).

DELT(I) (statement 61) - the time interval, beginning at time TAUI(I). for which the velocity-time trajectory of plane number NUMPL(I) is printed.

TAUI(I) (statement 61) - time at which printout of trajectory of plane NUMPL(I) is begun.

A.1. List of Variables.

This section contains in tabular form, a list of important symbols employed in the code, their definitions, and the corresponding symbol used in the text.

TABLE A.1. Variables in Computer Program.

Symbol in Code	Definition	Symbol in Text
DELT(I)	Time interval for trajectory printout of plane NUMPL(I)	
DENS	Particle density in thermodynamic region of lattice	ρ
DV	Velocity interval in calculation of distribution function	
DXAV	Average x component of displacement of particles in a plane	
DYAV	Average y component of displacement of particles in a plane	
DZAV	Average z component of displacement of particles in a plane	
ECHECK	Sum of initial energy in lattice and total work done	
EINIT	Initial energy in primary lattice	
EP	Potential energy of a single particle	Φ_{ijk}

TABLE A.1. (Continued)

Symbol in Code	Definition	Symbol in Text
EPAV	Average potential energy of single particle in thermodynamic region of lattice	
EPOT	Total potential energy in thermodynamic region of lattice	
ETHERM	Thermal (kinetic) energy in thermodynamic region of lattice	
ETOT	Total energy in lattice	
ETRANS	Translational (kinetic) energy in thermodynamic region of lattice	
FX	x component of force exerted on a particle	$(F_{ijk})_x$
FY	y component of force exerted on a particle	$(F_{ijk})_y$
FZ	z component of force exerted on a particle	$(F_{ijk})_z$
GAMMA	Ratio of dissociation to thermal energy in lattice	γ
H	Time step	
ICALC	Index to determine which calculation code is to perform	
IPRINT	Index to determine printout frequency for output	
K21	Number of planes (normal to z axis) contained within a thermodynamic region	
LX	One-half the number of planes of atoms in the lattice normal to the x axis	L_x
LY	One-half the number of planes of atoms normal to the y axis	L_y
LZ	One-half the number of planes of atoms normal to the z axis	
LZLAST	One-half the number of planes of atoms, normal to the z axis, in the previous calculation	
LZSEG	One-half the number of planes of atoms, normal to the z axis, in the primary lattice	L_z
NK1K2	Total number of atoms in thermodynamic region	n_R
NMAX	Largest set of equilibrium nearest neighbors scanned	

TABLE A.1. (Continued)

Symbol in Code	Definition	Symbol in Text
NMIN	Largest set of equilibrium nearest neighbors assumed to be within RCRIT	
NP	Number of particles per plane normal to the z axis in primary lattice	
NPLANE	Number of plane normal to the z axis	$n_{p\ell}$
NPMAX	Total number of planes whose trajectories are printed	
NSEG	Number of segments, equal in size to the primary lattice, added to lattice of previous calculation, i.e. LZ=LZLAST+NSEG*LZSEG	
NTOT	Total number of particles in lattice	
NUMPL(I)	One of NPMAX planes whose trajectories are printed	
PKXX	xx component of kinetic contribution to pressure tensor	$(\sigma_k)_{xx}$
PKYY	yy component of kinetic contribution to pressure tensor	$(\sigma_k)_{yy}$
PKZZ	zz component of kinetic contribution to pressure tensor	$(\sigma_k)_{zz}$
PVXX	xx component of potential contribution to pressure tensor	$(\sigma_v)_{xx}$
PVYY	yy component of potential contribution to pressure tensor	$(\sigma_v)_{yy}$
PVZZ	zz component of potential contribution to pressure tensor	$(\sigma_v)_{zz}$
PXX	xx component of pressure tensor	σ_{xx}
PYY	yy component of pressure tensor	σ_{yy}
PZZ	zz component of pressure tensor	σ_{zz}
R	Anharmonicity factor in Morse potential	R
RE	Lattice constant, a_o, normalized by single-pair equilibrium separation, r_o, in Morse potential	A_o
T	Temperature in thermodynamic region of lattice	θ
TAU	Time	τ

TABLE A.1. (Continued)

Symbol in Code	Definition	Symbol in Text
TAUI(I)	Time at which printout of trajectory of plane NUMPL(I) is begun	
TAUMAX	Maximum time for which calculation is carried out	
TX	x component of temperature in thermodynamic region	θ_x
TY	y component of temperature in thermodynamic region	θ_y
TZ	z component of temperature in thermodynamic region	θ_z
UP	Compression velocity	U_p
VX(I,J,K)	x component of velocity of particle (i,j,k)	$(V_{ijk})_x$
VY(I,J,K)	y component of velocity of particle (i,j,k)	$(V_{ijk})_y$
VZ(I,J,K)	z component of velocity of particle (i,j,k)	$(V_{ijk})_z$
VXAV	Average x component of velocity of particles in a plane normal to the z axis	
VYAV	Average y component of velocity of particles in a plane normal to the z axis	
VZAV	Average z component of velocity of particles in a plane normal to the z axis	
X(I,J,K)	x component of position of particle (i,j,k)	X_{ijk}
Y(I,J,K)	y component of position of particle (i,j,k)	Y_{ijk}
Z(I,J,K)	z component of position of particle (i,j,k)	Z_{ijk}

A.2. Description of Subprograms.

This section contains a list of the sixteen subprograms with a basic description of the function of each of the routines. The list (alphabetical) is as follows:

CALDIST - The subroutine calculates the components of the distance between two particles whose potential energy or force of interaction is being computed. The purpose is to determine whether one particle lies within the critical radius of the other.

CONVERT - The subroutine converts integers (ℓ,m,n), identifying a particle outside the primary lattice, to the corresponding atom inside the primary lattice identified by indices (LEQ,MEQ,NEQ).

DISTFN - The subroutine calculates the velocity distribution function in thermodynamic regions of the lattice. More precisely, it

determines the fractional number of atoms whose velocity at any time has components lying between VX and VX+DV, VY and VY+DV, and VZ and VZ+DV.

FORCE - The subroutine calculates the components of the force exerted on one particle by another, the distance between the atom being provided by CALDIST.

NABORS - The subroutine calculates the differences I-L, J-M, and K-N for any particle (I,J,K) and its neighbor (L,M,N). The values are stored in an array for future use in the program.

NRAN31 - The subroutine generates random numbers, consistent with a Gaussian distribution, for assigning velocities to atoms of the primary lattice.

OUTPUT - The subroutine prints the single - particle displacements and velocities for every atom whose equations of motion are solved. Also printed are the total sums of x,y, and z components of velocities.

PLANEAV - The subroutine averages the x,y, and z components of displacements and velocity for every atom in a single plane and prints the results. Also printed are the total sums of the components for all atoms in the lattice.

POTFOR - The subroutine calculates the potential energy of a single particle and the contribution to the potential part of the stress tensor from that particle.

SAVE - The subroutine saves current values of position for each atom in the vector (XS(I,J,K), YS(I,J,K), ZS(I,J,K)).

SCAN - The subroutine determines which neighbors between NMIN and NMAX lie within the critical radius of a given particle and calls the subroutines which calculate the potential of the particle as well as the force exerted on it. For the first iteration of the Runge-Kutta scheme, the nearest neighbors are calculated and are written on TAPE 8; for subsequent iterations atoms within the critical radius are assumed not to change.

SOLVE - The subroutine uses a fourth-order Runge-Kutta scheme to numerically solve the equations of motion for each atom in the lattice. It also calculates the work done on the lattice by the external driving force in a particular time interval.

SORT - The subroutine (used in conjunction with DISTFN) counts the number of atoms lying in each velocity interval.

START - The subroutine assigns atoms to their initial equilibrium positions and generates their initial velocities consistent with a Maxwellian distribution.

THERMO - The subroutine calculates the temperature, density, pressure tensor and mean velocity, averaged over thermodynamic regions of the lattice.

URAN31 - This function subprogram is called from NRAN31 and used to generate random numbers for initial conditions in the primary lattice.

A.3. Listing of Computer Program.

The final section of this appendix contains a complete listing of the main program and each of the sixteen subprograms.

```
0001:         PROGRAM BATTEH(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT,TAPE7,TAPE8)
0002:          COMMON/VEL/VX(04,04,300),VY(04,04,300),VZ(04,04,300)
0003:         COMMON/POSITS/XS(04,04,300),YS(04,04,300),ZS(04,04,300)
0004:         COMMON/PUSIT/X(04,04,300),Y(04,04,300),Z(04,04,300)
0005:         COMMON/NNBORS/IX(6,24,2),IY(6,24,2),IZ(6,24,2),NUM(6)
0006:         COMMON/SEARCH/NMAX,NMIN,NN,RCRIT,RCRIT2
0007:         COMMON/SIZE/LX,LY,LZ,NP,NTOT,LZ4,LZSEG4
0008:         COMMON/MISC/GAMMA,TAU,H,FX,FY,FZ,R,RE,EP,ICALC,ECHECK
0009:         DIMENSION NUMPL(20),II(20),TAU1(20),DELT(20),TAUI(20)
0010:         READ(5,100) LX,LY,LZ
0011:   C LX,LY,AND LZ ARE DIMENSIONS THIS CALCULATION. LX,LY, AND LZLAST ARE
0012:   C DIMENSIONS OF LAST CALCULATION, LX,LY, AND LZSEG ARE DIMENSIONS FOR A
0013:   C UNIT SEGMENT.
0014:         READ(5,101) NMAX,NMIN,RCRIT
0015:         READ(5,102) H,TAUMAX
0016:         READ(5,103) IPRINT
0017:         READ(5,104) R,RE
0018:         READ(5,105) GAMMA
0019:   C ICALC=1,2,OR3 FOR INITIAL-CONDITION CALCULATION,INITIAL SHOCK-WAVE
0020:   C CALCULATION,OR SUBSEQUENT SHOCK-WAVE CALCULATIONS,RESPECTIVELY.
0021:         READ(5,106) ICALC
0022:         READ(5,107) K21
0023:   C K21 IS THE NUMBER OF PLANES OVER WHICH THE DISTRIBUTION FUNCTION IS
0024:   C CALCULATED.
0025:         NINT=2*LZ/K21
0026:         NP=2*LX*LY
0027:         LZ4=4*LZ
0028:         NTOT=NP*LZ4/2
0029:         RCRIT2=RCRIT**2
0030:         IF(ICALC .EQ. 1) LZSEG=LZ
0031:         IF(ICALC .EQ. 1) LZSEG4=4*LZSEG
0032:         WRITE(6,108) LX,LY,LZ
0033:         WRITE(6,109) NMAX,NMIN,RCRIT
0034:         WRITE(6,110) H,TAUMAX
0035:         WRITE(6,111) R,RE
0036:         WRITE(6,112) GAMMA
0037:         CALL NABORS
0038:         IF(ICALC .NE. 1) GO TO 10
0039:         CALL START
0040:   C CALCULATE ZERO-POINT POTENTIAL ENERGY.
0041:         CALL SAVE
0042:         CALL SCAN(LX,LY,LZ4,2,1)
0043:         REWIND 8
0044:         EPTOT=EP*FLOAT(NTOT)
0045:   C CALCULATE INITIAL KINETIC ENERGY IN LATTICE.
0046:         EKTOT=0.0
0047:         DO 4 I=1,LX
0048:         DO 4 J=1,LY
0049:         DO 4 K=1,LZ4
0050:       4 EKTOT=EKTOT+0.5*(VX(I,J,K)**2+VY(I,J,K)**2+VZ(I,J,K)**2)
0051:         ECHECK=EKTOT+EPTOT
0052:         WRITE(6,113) EPTOT,EKTOT,ECHECK
0053:         TAU=0.0
0054:         GO TO 65
0055:   C BEGIN SEQUENCE FOR PERFORMING INITIAL SHOCK-WAVE CALCULATION.
```

```
0056:     10 READ(5,114) LZLAST,LZSEG,NSEG
0057:        READ(5,115) UP
0058:        READ(5,120) NPMAX,KPRINT
0059:        IF(NPMAX .EQ. 0) GO TO 13
0060:        DO 12 LL=1,NPMAX
0061:        READ(5,121)NUMPL(LL),DELT(LL),TAUI(LL)
0062:        II(LL)=0
0063:     12 TAU1(LL)=0.0
0064:     13 WRITE(6,116) UP
0065:        WRITE(6,119) LZSEG,LZLAST
0066:        LZLAS4=LZLAST*4
0067:        LZSEG4=LZSEG*4
0068:        DO 15 I=1,LX
0069:        DO 15 J=1,LY
0070:        DO 15 K=1,LZLAS4
0071:     15 READ(7) X(I,J,K),Y(I,J,K),Z(I,J,K),VX(I,J,K),VY(I,J,K),VZ(I,J,K)
0072:        IF(ICALC .EQ. 3) GO TO 40
0073:        REWIND 7
0074:        TAU=0.0
0075:        DO 25 I1=1,NSEG
0076:        DO 20 I=1,LX
0077:        DO 20 J=1,LY
0078:        DO 20 K=1,LZSEG4
0079:        KP=I1*LZSEG4+K
0080:        X(I,J,KP)=X(I,J,K)
0081:        Y(I,J,KP)=Y(I,J,K)
0082:        Z(I,J,KP)=Z(I,J,K)+0.5*FLOAT((KP-1)/2)-.5*FLOAT((K-1)/2)
0083:        VX(I,J,KP)=VX(I,J,K)
0084:        VY(I,J,KP)=VY(I,J,K)
0085:     20 VZ(I,J,KP)=VZ(I,J,K)
0086:     25 CONTINUE
0087:        DO 30 I=1,LX
0088:        DO 30 J=1,LY
0089:        VZ(I,J,1)=UP
0090:     30 VZ(I,J,2)=UP
0091:  C CALCULATE INITIAL ENERGY IN LATTICE.
0092:        EKTOT=0.0
0093:        EPTOT=0.0
0094:        DO 35 I=1,LX
0095:        DO 35 J=1,LY
0096:        DO 35 K=1,LZ4
0097:        EKTOT=EKTOT+0.5*(VX(I,J,K)**2+VY(I,J,K)**2+VZ(I,J,K)**2)
0098:        CALL SAVE
0099:        CALL SCAN(I,J,K,2,1)
0100:     35 EPTOT=EPTOT+EP
0101:        REWIND 8
0102:        ECHECK=EPTOT+EKTOT
0103:        WRITE(6,117) TAU,ECHECK
0104:        CALL PLANEAV
0105:        GO TO 65
0106:     40 READ(7) TAU,ECHECK,ETOT1
0107:        REWIND 7
0108:        IF(NSEG .EQ. 0) GO TO 60
0109:        DO 50 I1=1,NSEG
0110:        DO 45 I=1,LX
```

```
0111:          DO 45 J=1, Y
0112:          DO 45 K=1,LZSEG4
0113:          KP=LZLAS4+(I1-1)*LZSEG4+K
0114:          KPP=LZLAS4-LZSEG4+K
0115:          X(I,J,KP)=X(I,J,KPP)
0116:          Y(I,J,KP)=Y(I,J,KPP)
0117:          Z(I,J,KP)=Z(I,J,KPP)+ 0.5*FLOAT((KP-1)/2)-0.5*FLOAT((KPP-1)/2)
0118:          VX(I,J,KP)=VX(I,J,KPP)
0119:          VY(I,J,KP)=VY(I,J,KPP)
0120:       45 VZ(I,J,KP)=VZ(I,J,KPP)
0121:       50 CONTINUE
0122:  C CALCULATE ENERGY IN LATTICE.
0123:          EKTOT=0.0
0124:          EPTOT=0.0
0125:          DO 55 I=1,LX
0126:          DO 55 J=1,LY
0127:          DO 55 K=1,LZ4
0128:          EKTOT=EKTOT+0.5*(VX(I,J,K)**2+VY(I,J,K)**2+VZ(I,J,K)**2)
0129:          CALL SAVE
0130:          CALL SCAN(I,J,K,2,1)
0131:       55 EPTOT=EPTOT+EP
0132:          REWIND 8
0133:          ECHECK=ECHECK+EPTOT+EKTOT-ETOT1
0134:       60 CONTINUE
0135:          WRITE(6,117) TAU,ECHECK
0136:          CALL PLANEAV
0137:       65 M=0
0138:       70 M=M+1
0139:          IF(TAU .GE. TAUMAX-H/2.0) GO TO 90
0140:          CALL SOLVE
0141:          IF(NPMAX .EQ. 0) GO TO 75
0142:          DO 74 LL=1,NPMAX
0143:          IF(TAU.LT.TAUI(LL))GO TO 74
0144:          II(LL)=II(LL)+1
0145:          TAU1(LL)=TAU1(LL)+H
0146:          IF(TAU1(LL).GT.DELT(LL)) GO TO 74
0147:          IF(II(LL).NE.KPRINT) GO TO 74
0148:          KK=NUMPL(LL)
0149:          KK2=2*KK
0150:          KK2M1=2*KK-1
0151:          VZAV=0.0
0152:          VXAV=0.0
0153:          VYAV=0.0
0154:          DO 72 I=1,LX
0155:          DO 72 J= ,LY
0156:          VXAV=VXAV+VX(I,J,KK2)+VX(I,J,KK2M1)
0157:          VYAV=VYAV+VY(I,J,KK2)+VY(I,J,KK2M1)
0158:       72 VZAV=VZAV+VZ(I,J,KK2)+VZ(I,J,KK2M1)
0159:          VZAV=VZAV/FLOAT(NP)
0160:          VXAV=VXAV/FLOAT(NP)
0161:          VYAV=VYAV/FLOAT(NP)
0162:          VAV=SQRT(VXAV**2+VYAV**2+VZAV**2)
0163:          WRITE(6,122) KK,VZAV,TAU,VXAV,VYAV,VAV
0164:          II(LL)=0
0165:       74 CONTINUE
```

```
0166:      75 IF(M .NE. IPRINT) GO TO 70
0167:         ETOT=0.0
0168:         M=0
0169:         WRITE(6,117) TAU,ECHECK
0170:         IF(ICALC .NE. 1) GO TO 80
0171:         CALL OUTPUT
0172:         CALL OUTPUT
0173:      80 CALL PLANEAV
0174:         DO 85 I=1,NINT
0175:         K1=(I-1)*K21+1
0176:         K2=I*K21
0177:         CALL DISTFN(K1,K2,0,1)
0178:      85 CALL THERMO(K1,K2,ETOT)
0179:         WRITE(6,118) ETOT
0180:         DO 87 I=1,LX
0181:         DO 87 J=1,LY
0182:         DO 87 K=1,LZ4
0183:      87 WRITE(7) X(I,J,K),Y(I,J,K),Z(I,J,K),VX(I,J,K),VY(I,J,K),VZ(I,J,K)
0184:         WRITE(7) TAU,ECHECK,ETOT
0185:         REWIND 7
0186:         GO TO 70
0187:      90 STOP
0188:     100 FORMAT(3I3)
0189:     101 FORMAT(I3,I3,F7.3)
0190:     102 FORMAT(2F9.5)
0191:     103 FORMAT(I4)
0192:     104 FORMAT(2F9.5)
0193:     105 FORMAT(F9.5)
0194:     106 FORMAT(I3)
0195:     107 FORMAT(I3)
0196:     108 FORMAT(2X,8LX=8,I3,2X,8LY=8,I3,2X,8LZ=8,I3)
0197:     109 FORMAT(1X,8NMAX=8,I2,3X,8NMIN=8,I2,3X,8RCRIT=8,E14.6)
0198:     110 FORMAT(1X,8H=8,E14.6,3X,8TAUMAX=8,E14.6)
0199:     111 FORMAT(1X,8R=8,E14.6,3X,8RE=8,E14.6)
0200:     112 FORMAT(1X,8GAMMA=8,E14.6)
0201:     113 FORMAT(/1X,8ZERO-POINT POTENTIAL=8,E14.6,3X,8INITIAL KINETIC ENERG
0202:        2Y=8,E14.6,3X,8ECHECK=8,E14.6)
0203:     114 FORMAT(3I4)
0204:     115 FORMAT(F8.5)
0205:     116 FORMAT(1X,8UP=8,E14.6)
0206:     117 FORMAT(1H1,8TAU=8,E14.6,3X,8ECHECK=8,E14.6)
0207:     118 FORMAT(1X,8ETOT=8,E14.6)
0208:     119 FORMAT(1X,8LZSEG=8,I3,3X,8LZLAST=8,I3)
0209:     120 FORMAT(2I3)
0210:     122 FORMAT(1X,8PL NUM=8,I3,5X,8VZAV=8, E14.6,5X,8TAU=8,E14.6,5X,
0211:        28VXAV=8,E14.6,5X,8VYAV=8,E14.6,2X,8VAV=8,E14.6)
0212:     121 FORMAT(I3,2F9.5)
0213:         END
```

```
0001:          SUBROUTINE CALDIST(L,M,N,LEQ,MEQ,NEQ,I,J,K)
0002:          COMMON/STRESS/DISTX,DISTY,DISTZ,DIST,DIST2,C1,PVXX,PVYY,PVZZ
0003:          COMMON/POSITS/X(04,04,300),Y(04,04,300),Z(04,04,300)
0004:          IF(L.EQ.LEQ)GO TO 205
0005:          IPP=MOD(N,4)
0006:          IP=MOD(NEQ,4)/2-MOD(4+IPP,4)/2
0007:          DISTX=X(I,J,K)-X(LEQ,MEQ,NEQ)+.5*FLOAT(2*LEQ-2*L+IP)
0008:          GO TO 206
0009:      205 DISTX=X(I,J,K)-X(LEQ,MEQ,NEQ)
0010:      206 IF(M.EQ.MEQ)GO TO 215
0011:          IP=MOD(NEQ-1,2)-IABS(MOD(N-1,2))
0012:          DISTY=Y(I,J,K)-Y(LEQ,MEQ,NEQ)+.5*FLOAT(2*MEQ-2*M+IP)
0013:          GO TO 216
0014:      215 DISTY=Y(I,J,K)-Y(LEQ,MEQ,NEQ)
0015:      216 IF(NEQ.EQ.N)GO TO 220
0016:          IF(N.GT.0)IP=(N-1)/2
0017:          IF(N.LE.0)IP=(N-2)/2
0018:          DISTZ=Z(I,J,K)-Z(LEQ,MEQ,NEQ)+.5*FLOAT((NEQ-1)/2)-.5*FLOAT(IP)
0019:          GO TO 221
0020:      220 DISTZ=Z(I,J,K)-Z(LEQ,MEQ,NEQ)
0021:      221 DIST2=DISTX**2+DISTY**2+DISTZ**2
0022:          RETURN
0023:          END
```

```
0001:           SUBROUTINE CONVERT(L,M,N,LEQ,MEQ,NEQ)
0002:          COMMON/SIZE/LX,LY,LZ,NP,NTOT,LZ4,LZSE64
0003:          COMMON/STRESS/DISTX,DISTY,DISTZ,DIST,DIST2,CI,PVXX,PVYY,PVZZ
0004:    C SUBROUTINE CONVERTS L,M,AND N INTO EQUIVALENT VALUES FOR PERIODIC
0005:    C BOUNDARY CONDITIONS AND FINDS DISTANCE FROM SCANNED TO TESTED PARTICLE
0006:    C DETERMINE LEQ
0007:          IF(L .GE. 1 .AND.L.LE. LX) GO TO 105
0008:          IF(L.LT.1) GO TO 102
0009:          LEQ=MOD(L-1,LX)+1
0010:          GO TO 110
0011:      102 IN=1+IABS(L)/LX
0012:          LEQ=IN*LX-IABS(L)
0013:          GO TO 110
0014:      105 LEQ=L
0015:    C DETERMINE MEQ
0016:      110 IF(M.GE.1.AND.M.LE.LY) GO TO 118
0017:          IF(M.LT.1) GO TO 112
0018:          MEQ=MOD(M-1,LY)+1
0019:          GO TO 120
0020:      112 IN=1+IABS(M)/LY
0021:          MEQ=IN*LY-IABS(M)
0022:          GO TO 120
0023:      118 MEQ=M
0024:    C DETERMINE NEQ
0025:      120 IF(N.GE.1.AND.N.LE.LZ4) GO TO 128
0026:          IF(N.LT.1) GO TO 122
0027:          NEQ=MOD(N-1,LZ4)+1+LZ4-LZSE64
0028:          GO TO 130
0029:      122 IN=1+IABS(N)/LZ4
0030:          NEQ=IN*LZ4-IABS(N)
0031:          GO TO 130
0032:      128 NEQ=N
0033:      130 CONTINUE
0034:          RETURN
0035:          END
```

```
0001:          SUBROUTINE DISTFN(K1,K2,DV)
0002:    C     CALCULATES DISTRIBUTION FUNCTION FOR VX, VY, AND VZ WITH INTERVAL
0003:    C     DV IN THE REGION BETWEEN AND INCLUDING PLANES K1 AND K2.
0004:          COMMON/SIZE/LX,LY,LZ,NP,NTOT,LZ4,LZSEG4
0005:           COMMON/VEL/VX(04,04,300),VY(04,04,300),VZ(04,04,300)
0006:          COMMON/DIST/NIR(100),NIL(100),NIMAX
0007:          DO 1 J=1,100
0008:          NIL(J)=0
0009:        1 NIR(J)=0
0010:    C     CALCULATE TOTAL NUMBER OF PARTICLES BETWEEN K1 AND K2
0011:          LK=K2-K1+1
0012:          NK1K2=LK*NP
0013:          RNK1K2=FLOAT(NK1K2)
0014:          WRITE(6,99)
0015:          WRITE(6,100)K1,K2,NK1K2,DV
0016:    C     *** VX DISTRIBUTION ***
0017:          WRITE(6,101)
0018:          WRITE(6,102)
0019:          NIMAX=1
0020:          LZ1=2*K1-1
0021:          LZ2=2*K2
0022:          DO 2 K=LZ1,LZ2
0023:          DO 2 J=1,LY
0024:          DO 2 I=1,LX
0025:          V=VX(I,J,K)
0026:        2 CALL SORT(V,DV)
0027:          DO 3 I=1,NIMAX
0028:          VR=.5*DV*FLOAT(2*I-1)
0029:          VL=.5*DV*FLOAT(1-2*I)
0030:          RNR=FLOAT(NIR(I))/RNK1K2
0031:          RNL=FLOAT(NIL(I))/RNK1K2
0032:        3 WRITE(6,103)VR,RNR,VL,RNL
0033:    C     *** VY DISTRIBUTION ***
0034:          DO 4 I=1,100
0035:          NIL(I)=0
0036:        4 NIR(I)=0
0037:          NIMAX=1
0038:          DO 5 K=LZ1,LZ2
0039:          DO 5 J=1,LY
0040:          DO 5 I=1,LX
0041:          V=VY(I,J,K)
0042:        5 CALL SORT(V,DV)
0043:          WRITE( 6,104)
0044:          WRITE(6,102)
0045:          DO 6 I=1,NIMAX
0046:          VR=.5*DV*FLOAT(2*I-1)
0047:          VL=.5*DV*FLOAT(1-2*I)
0048:          RNR=FLOAT(NIR(I))/RNK1K2
0049:          RNL=FLOAT(NIL(I))/RNK1K2
0050:        6 WRITE(6,103)VR,RNR,VL,RNL
0051:    C     *** VZ DISTRIBUTION ***
0052:          DO 7 I=1,100
0053:          NIL(I)=0
0054:        7 NIR(I)=0
0055:          NIMAX=1
```

```
0056:         DO 8 K=LZ1,LZ2
0057:         DO 8 J=1,LY
0058:         DO 8 I=1,LX
0059:         V=VZ(I,J,K)
0060:       8 CALL SORT(V,DV)
0061:         WRITE(6,105)
0062:         WRITE(6,102)
0063:         DO 9 I=1,NIMAX
0064:         VR=.5*DV*FLOAT(2*I-1)
0065:         VL=.5*DV*FLOAT(1-2*I)
0066:         RNR=FLOAT(NIR(I))/RNK1K2
0067:         RNL=FLOAT(NIL(I))/RNK1K2
0068:       9 WRITE(6,103)VR,RNR,VL,RNL
0069:      99 FORMAT(/////2X,'DATA FROM SUBROUTINE DISTFN')
0070:     100 FORMAT(2X,'DISTRIBUTION FUNCTION FOR PARTICLES BETWEEN PLANES',I3
0071:        2,' AND ',I3/2X,'NUMBER OF PARTICLES IN SAMPLE = ',
0072:        3I5,5X,'DV = ',F13.6)
0073:     101 FORMAT(2X,'*** VX DISTRIBUTION ***')
0074:     102 FORMAT(3X,'VAVG',3X,'N/NTOT',4X,'VAVG',3X,'N/NTOT')
0075:     103 FORMAT(1X,2F7.4,3X,2F7.4)
0076:     104 FORMAT(2X,'*** VY DISTRIBUTION ***')
0077:     105 FORMAT(2X,'*** VZ DISTRIBUTION ***')
0078:         RETURN
0079:         END
```

```
0001:          SUBROUTINE FORCE
0002:          COMMON/MISC/GAMMA,TAU,H,FX,FY,FZ,R,RE,EP,ICALC,ECHECK
0003:          COMMON/MISC2/FORCEX,FORCEY,FORCEZ
0004:          COMMON/STRESS/DISTX,DISTY,DISTZ,DIST,DIST2,C1,PVXX,PVYY,PVZZ
0005:          C2=C1**2
0006:          IF(C1.EQ.0.0)GO TO 1
0007:          FORCEX=(C2-C1)*DISTX/DIST
0008:          FORCEY=(C2-C1)*DISTY/DIST
0009:          FORCEZ=(C2-C1)*DISTZ/DIST
0010:          GO TO 2
0011:        1 FORCEX=0.0
0012:          FORCEY=0.0
0013:          FORCEZ=0.0
0014:        2 FX=FX+FORCEX
0015:          FY=FY+FORCEY
0016:          FZ=FZ+FORCEZ
0017:          RETURN
0018:          END
```

```
0001:         SUBROUTINE NABORS
0002:   C FOR ANY PARTICLE (I,J,K) SUBROUTINE CALCULATES I-L,J-M, AND K-N WHERE
0003:   C(L,M,N) IS A NEIGHBOR. VALUES ARE STORED IN ARRAYS IX(I,J,1),IX(I,J,2)
0004:   CIY(I,J,1) ETC. WHERE I HERE DENOTES UST,2ND, 3RD ETC. AND J THE
0005:   C PARTICULAR ONE OF THE NEIGHBORS. TWO EXPRESSIONS ARE USED TO
0006:   C CALCULATE IX,IY, AND IZ DEPENDING ON THE VALUE OF K OF THE SCANNED
0007:   C PARTICLE, AND THE 3RD INDEX DETERMINES WHICH EXPRESSION IS USED.
0008:         COMMON/SEARCH/NMAX,NMIN,NN,RCRIT,RCRIT2
0009:         COMMON/NNBORS/IX(6,24,2),IY(6,24,2),IZ(6,24,2),NUM(6)
0010:         DO 50 I=1,6
0011:      50 NUM(I)=0
0012:         DO 100 LL=1,21
0013:         DO 100 MM=1,21
0014:         DO 100 NNN=1,21
0015:         L=LL-11
0016:         M=MM-11
0017:         N=NNN-11
0018:         I=L**2+M**2+N**2+L*N+L*M+M*N
0019:         IF(I .GT. NMAX) GO TO 100
0020:         NUM(I)=NUM(I)+1
0021:         IRX2=L+M
0022:         IRY2=L+N
0023:         IRZ2=M+N
0024:         K=NUM(I)
0025:         IX(I,K,1)=(IRX2-1+IABS(MOD(IRX2-1,2)))/2
0026:         IX(I,K,2)=(IRX2+IABS(MOD(IRX2,2)))/2
0027:         IY(I,K,1)=(IRY2-1+IABS(MOD(IRY2-1,2)))/2
0028:         IY(I,K,2)=(IRY2+IABS(MOD(IRY2,2)))/2
0029:         IZ(I,K,1)=2*IRZ2+IABS(MOD(IRY2,2))
0030:         IZ(I,K,2)=2*IRZ2-1+IABS(MOD(IRY2-1,2))
0031:     100 CONTINUE
0032:         RETURN
0033:         END
```

```
0001:          SUBROUTINE NRAN31(X1,X2,I)
0002:          COMMON/MISC/GAMMA,TAU,H,FX,FY,FZ,R,RE,EP,ICALC,ECHECK
0003:          X3=SQRT(-2.0*ALOG(URAN31(I)))
0004:          X4=6.2831853072*URAN31(I)
0005:          X2=X3*SIN(X4)
0006:          X1=X3*COS(X4)
0007:          X2=X2/SQRT(GAMMA)
0008:          X1=X1/SQRT(GAMMA)
0009:          RETURN
0010:          END
```

```
0001:       SUBROUTINE OUTPUT
0002:  C SUBROUTINE PRINTS SINGLE-PARTICLE DISPLACEMENTS AND VELOCITIES
0003:       COMMON/SIZE/LX,LY,LZ,NP,NTOT,LZ4,LZSEG4
0004:        COMMON/VEL/VX(04,04,300),VY(04,04,300),VZ(04,04,300)
0005:       COMMON/POSIT/X(04,04,300),Y(04,04,300),Z(04,04,300)
0006:       COMMON/MISC/GAMMA,TAU,H,FX,FY,FZ,R,RE,EP,ICALC,ECHECK
0007:       WRITE(6,101)
0008:       SUMVX=0.
0009:       SUMVY=0.
0010:       SUMVZ=0.
0011:       WRITE(6,102)
0012:       DO 5 K=1,LZ4
0013:       DO 5 J=1,LY
0014:       DO 5 I=1,LX
0015:       DX=X(I,J,K)-FLOAT(I)+1.-.5*FLOAT(MOD(K,4)/2)
0016:       DY=Y(I,J,K)-FLOAT(J)+1.-.5*FLOAT(MOD(K-1,2))
0017:       DZ=Z(I,J,K)-0.5*FLOAT((K-1)/2)
0018:       SUMVX=SUMVX+VX(I,J,K)
0019:       SUMVY=SUMVY+VY(I,J,K)
0020:       SUMVZ=SUMVZ+VZ(I,J,K)
0021:       NPLANE=(K-1)/2+1
0022:     5 WRITE(6,103)NPLANE,I,J,K,DX,DY,DZ,VX(I,J,K),VY(I,J,K),VZ(I,J,K)
0023:       WRITE(6,104)SUMVX,SUMVY,SUMVZ
0024:   101 FORMAT(/////2X,8DATA FROM SUBROUTINE OUTPUT8)
0025:   102 FORMAT(2X,8PL8,6X,8I8,6X,8J8,6X,8K8,9X,8DX8,12X,8DY8,13X,8DZ8,13X
0026:      2,8VX8,13X,8VY8,13X,8VZ8)
0027:   103 FORMAT(1X,4(I4,3X),6(E12.4,3X))
0028:   104 FORMAT(1X,8SUMVX = 8,E14.6,3X,8SUMVY = 8,E14.6,3X,8SUMVZ = 8,E14.
0029:      26)
0030:       RETURN
0031:       END
```

```
0001:         SUBROUTINE PLANEAV
0002:   C SUBROUTINE CALCULATES AVERAGE VALUES OF POSITION AND VELOCITY FOR
0003:   C PARTICLES IN A GIVEN PLANE.
0004:         COMMON/SIZE/LX,LY,LZ,NP,NTOT,LZ4,LZSEG4
0005:          COMMON/VEL/VX(04,04,300),VY(04,04,300),VZ(04,04,300)
0006:         COMMON/POSIT/X(04,04,300),Y(04,04,300),Z(04,04,300)
0007:         WRITE(6,100)
0008:         WRITE(6,101)
0009:         SUMVX=0.
0010:         SUMVY=0.
0011:         SUMDX=0.0
0012:         SUMDY=0.0
0013:         SUMVZ=0.
0014:         DO 30 K=1,LZ4,2
0015:         DZAV=0.
0016:         DYAV=0.0
0017:         DXAV=0.0
0018:         VXAV=0.
0019:         VYAV=0.
0020:         VZAV=0.
0021:         DO 20 J=1,LY
0022:         DO 20 I=1,LX
0023:         DZAV=DZAV+Z(I,J,K)+Z(I,J,K+1)-.5*FLOAT((K-1)/2)-.5*FLOAT(K/2)
0024:         DXAV=DXAV+X(I,J,K)+X(I,J,K+1)-2.0*FLOAT(I)+2.0
0025:        2-0.5*FLOAT(MOD(K+1,4)/2)-0.5*FLOAT(MOD(K,4)/2)
0026:         DYAV=DYAV+Y(I,J,K)+Y(I,J,K+1)-2.0*FLOAT(J)+2.0
0027:        2-0.5*FLOAT(MOD(K,2))-0.5*FLOAT(MOD(K-1,2))
0028:         VXAV=VXAV+VX(I,J,K)+VX(I,J,K+1)
0029:         VYAV=VYAV+VY(I,J,K)+VY(I,J,K+1)
0030:      20 VZAV=VZAV+VZ(I,J,K)+VZ(I,J,K+1)
0031:          SUMVX=SUMVX+VXAV
0032:         SUMVY=SUMVY+VYAV
0033:         SUMDX=SUMDX+DXAV
0034:         SUMDY=SUMDY+DYAV
0035:         SUMVZ=SUMVZ+VZAV
0036:         DZAV=DZAV/FLOAT(NP)
0037:         VXAV=VXAV/FLOAT(NP)
0038:         VYAV=VYAV/FLOAT(NP)
0039:         VZAV=VZAV/FLOAT(NP)
0040:         DXAV=DXAV/FLOAT(NP)
0041:         DYAV=DYAV/FLOAT(NP)
0042:         NPLANE=(K-1)/2+1
0043:      30 WRITE(6,102) NPLANE,DZAV,VZAV,VXAV,VYAV,DXAV,DYAV
0044:         WRITE(6,103) SUMVZ,SUMVX,SUMVY,SUMDX,SUMDY
0045:     103 FORMAT(//2X,@SUMS@,T24,E14.6,3X,E14.6,3X,E14.6,3X,E14.6,3X,E14.6)
0046:     100 FORMAT(/////2X,@DATA FROM SUBROUTINE PLANEAV@)
0047:     101 FORMAT(1X,@PL@,9X,@DZAV@,13X,@VZAV@,13X,@VXAV@,13X,@VYAV@,13X,
0048:        2@DXAV@,13X,@DYAV@)
0049:     102 FORMAT(1X,I3,3X,E14.6,3X,E14.6,3X,E14.6,3X,E14.6,3X,E14.6,3X,
0050:        2E14.6)
0051:         RETURN
0052:         END
```

```
0001:          SUBROUTINE POTFOR(N,ISCAN1)
0002:          COMMON/MISC2/FORCEX,FORCEY,FORCEZ
0003:          COMMON/STRESS/DISTX,DISTY,DISTZ,DIST,DIST2,C1,PVXX,PVYY,PVZZ
0004:          COMMON/MISC/GAMMA,TAU,H,FX,FY,FZ,R,RE,EP,ICALC,ECHECK
0005:          DIST=SQRT(DIST2)
0006:          C1=EXP(-R*(RE*DIST-1.0))
0007:          IF(ICALC.NE.1.AND.N.LT.1) C1=0.0
0008:          IF(ISCAN1.EQ.2) GO TO 135
0009:          CALL FORCE
0010:          IF(ISCAN1.NE.3) GO TO 149
0011:          PVXX=PVXX+DISTX*FORCEX
0012:          PVYY=PVYY+DISTY*FORCEY
0013:          PVZZ=PVZZ+DISTZ*FORCEZ
0014:      135 C=C1
0015:          IF(ICALC .NE.1.AND.N.LT.1) C=0.0
0016:          EP=EP-0.5+0.5*(C-1.0)**2
0017:      149 CONTINUE
0018:          RETURN
0019:          END
```

```
0001:         SUBROUTINE SAVE
0002:   C SUBROUTINE SAVES CURRENT VALUES OF POSITION IN VECTOR AS FOR USE IN
0003:   C SUBROUTINE SCAN
0004:         COMMON/SIZE/LX,LY,LZ,NP,NTOT,LZ4,LZSE64
0005:         COMMON/POSIT/X(64,64,300),Y(64,64,300),Z(64,64,300)
0006:         COMMON/POSITS/XS(64,64,300),YS(64,64,300),ZS(64,64,300)
0007:         DO 100 K=1,LZ4
0008:         DO 100 J=1,LY
0009:         DO 100 I=1,LX
0010:         XS(I,J,K)=X(I,J,K)
0011:         YS(I,J,K)=Y(I,J,K)
0012:     100 ZS(I,J,K)=Z(I,J,K)
0013:         RETURN
0014:         END
```

```
0001:         SUBROUTINE SCAN(I,J,K,ISCAN1,ISCAN2)
0002:   C IF ISCAN2=1, SUBROUTINE DETERMINES THE NEAREST NEIGHBORS TO PARTICLE
0003:   C (I,J,K) . IF ISCAN2=2, NEAREST NEIGHBORS ARE  ASSUMED TO BE THOSE
0004:   C CALCULATED AT SOME PREVIOUS TIME. IF ISCAN1=1, THE FORCE EXERTED ON
0005:   C PARTICLE (I,J,K) BY ITS NEIGHBORS IS RETURNED TO THE CALLING PROGRAM.
0006:   C IF ISCAN1=2, THE POTENTIAL ENERGY IS RETURNED. IF ISCAN1=3, BOTH ARE
0007:   C RETURNED AND CONTRIBUTIONS TO THE PRESSURE  TENSOR CALCULATED.
0008:         COMMON/SIZE/LX,LY,LZ,NP,NTOT,LZ4,LZSE64
0009:         COMMON/STRESS/DISTX,DISTY,DIST2,DIST,DIST2,C1,PVXX,PVYY,PVZZ
0010:         COMMON/POSITS/X(64,64,300),Y(64,64,300),Z(64,64,300)
0011:         COMMON/MISC/GAMMA,TAU,H,FX,FY,FZ,R,RE,EP,ICALC,ECHECK
0012:         COMMON/SEARCH/NMAX,NMIN,NN,RCRIT,RCRIT2
0013:         COMMON/NNBORS/IX(6,24,2),IY(6,24,2),IZ(6,24,8),NUM(6)
0014:         DIMENSION NUMNN(64,64,300)
0015:         FX=0.0
0016:         FY=0.0
0017:         FZ=0.0
0018:         EP=0.
0019:         PVXX=0.0
0020:         PVYY=0.0
0021:         PVZZ=0.0
0022:         KK=MOD(K,4)+1
0023:         GO TO (5,6,7,8),KK
0024:       5 IN3X=1
0025:         IN3Y=2
0026:         IN3Z=2
0027:         GO TO 9
0028:       6 IN3X=1
0029:         IN3Y=1
0030:         IN3Z=1
0031:         GO TO 9
0032:       7 IN3X=2
0033:         IN3Y=2
0034:         IN3Z=2
0035:         GO TO 9
0036:       8 IN3X=2
0037:         IN3Y=1
0038:         IN3Z=1
0039:       9 CONTINUE
0040:         IF(NMIN .EQ. 0) GO TO 151
0041:     100 DO 150 L1=1,NMIN
0042:         M1MAX=NUM(L1)
0043:         DO 149 M1=1,M1MAX
0044:         L=I+IX(L1,M1,IN3X)
0045:         M=J+IY(L1,M1,IN3Y)
0046:         N=K+IZ(L1,M1,IN3Z)
0047:         CALL CONVERT(L,M,N,LEQ,MEQ,NEQ)
0048:         CALL CALDIST(L,M,N,LEQ,MEQ,NEQ,I,J,K)
0049:         CALL POTFOR(N,ISCAN1)
0050:     149 CONTINUE
0051:     150 CONTINUE
0052:     151 CONTINUE
0053:         IF(NMAX.EQ.NMIN) GO TO 500
0054:         IF(ISCAN2.EQ.2) GO TO 400
0055:     200 NUMNN(I,J,K)=0
```

```
0056:          NMINP1=NMIN+1
0057:          DO 350 L1=NMINP1,NMAX
0058:          M1MAX=NUM(L1)
0059:          DO 349 M1=1,M1MAX
0060:          L=I+IX(L1,M1,IN3X)
0061:          M=J+IY(L1,M1,IN3Y)
0062:          N=K+IZ(L1,M1,IN3Z)
0063:          CALL CONVERT(L,M,N,LEQ,MEQ,NEQ)
0064:          CALL CALDIST(L,M,N,LEQ,MEQ,NEQ,I,J,K)
0065:          IF(DIST2 .GT. RCRIT2 .OR. DIST2 .LE. 1.0E-10) GO TO 349
0066:          NUMNN(I,J,K)=NUMNN(I,J,K)+1
0067:          WRITE(8) L,M,N,LEQ,MEQ,NEQ
0068:          CALL POTFOR(N,ISCAN1)
0069:      349 CONTINUE
0070:      350 CONTINUE
0071:          GO TO 500
0072:      400 NIJK=NUMNN(I,J,K)
0073:          IF(NIJK .EQ. 0) GO TO 500
0074:          DO 450 L1=1,NIJK
0075:          READ(8) L,M,N,LEQ,MEQ,NEQ
0076:          CALL CALDIST(L,M,N,LEQ,MEQ,NEQ,I,J,K)
0077:          CALL POTFOR(N,ISCAN1)
0078:      450 CONTINUE
0079:      500 CONTINUE
0080:          RETURN
0081:          END
```

```
0001:        SUBROUTINE SOLVE
0002:  C SUBROUTINE SOLVES EQUATIONS OF MOTION TO OBTAIN VALUES OF THE
0003:  C POSITION AND VELOCITY AT TIME TAU + H GIVEN THEIR VALUES AT TIME
0004:  C TAU. THE PROCEDURE IS VIA A FOURTH-ORDER RUNGE-KUTTA METHOD.
0005:        COMMON/SIZE/LX,LY,LZ,NP,NTOT,LZ4,LZSEG4
0006:         COMMON/VEL/VX(04,04,300),VY(04,04,300),VZ(04,04,300)
0007:        COMMON/POSIT/X(04,04,300),Y(04,04,300),Z(04,04,300)
0008:        COMMON/POSITS/XS(04,04,300),YS(04,04,300),ZS(04,04,300)
0009:        COMMON/SEARCH/NMAX,NMIN,NN,RCRIT,RCRIT2
0010:        COMMON/MISC/GAMMA,TAU,H,FX,FY,FZ,R,RE,EP,ICALC,ECHECK
0011:        COMMON/DUMMY/PX(04,04,300),PY(04,04,300),PZ(04,04,300),PVX(04,04,
0012:       2300),PVY(04,04,300),PVZ(04,04,300),XI(04,04,300),YI(04,04,300),
0013:       3ZI(04,04,300),VXI(04,04,300),VYI(04,04,300),VZI(04,04,300)
0014:        LEVEL 2,PX,PY,PZ,PVX,PVY,PVZ,XI,YI,ZI,VXI,VYI,VZI
0015:        DIMENSION FZI(04,04,02)
0016:  C SAVE VALUES OF POSITION AND VELOCITY AT BEGINNING OF INTERVAL IN
0017:  C VECTORS RI AND VI AND INITIALIZE PBS.
0018:        DO 50 I=1,LX
0019:        DO 50 J=1,LY
0020:        DO 50 K=1,LZ4
0021:        XI(I,J,K)=X(I,J,K)
0022:        YI(I,J,K)=Y(I,J,K)
0023:        ZI(I,J,K)=Z(I,J,K)
0024:        VXI(I,J,K)=VX(I,J,K)
0025:        VYI(I,J,K)=VY(I,J,K)
0026:        VZI(I,J,K)=VZ(I,J,K)
0027:        PX(I,J,K)=0.0
0028:        PY(I,J,K)=0.0
0029:        PZ(I,J,K)=0.0
0030:        PVX(I,J,K)=0.0
0031:        PVY(I,J,K)=0.0
0032:     50 PVZ(I,J,K)=0.0
0033:  C END SAVE AND INITIALIZATION SEQUENCE.
0034:        DO 900 M=1,4
0035:  C SAVE CURRENT VALUES OF POSITION IN RS.
0036:        CALL SAVE
0037:  C BEGIN SEQUENCE FOR SOLVING EQUATIONS.
0038:        REWIND 8
0039:        DO 200 K=1,LZ4
0040:        DO 200 J=1,LY
0041:        DO 200 I=1,LX
0042:        IF(M .NE. 1) GO TO 60
0043:        CALL SCAN(I,J,K,1,1)
0044:        GO TO 61
0045:     60 CALL SCAN(I,J,K,1,2)
0046:  C SCAN RETURNS VALUES OF FX,FY, AND FZ, THE COMPONENTS OF FORCES ACTING
0047:  C ON THE PARTICLE BEING SCANNED.
0048:     61 FX=2.0*RE*R*FX
0049:        FY=2.0*RE*R*FY
0050:        FZ=2.0*RE*R*FZ
0051:        IF(K .EQ. 1 .AND. M .EQ. 1) FZI(I,J,1)=-FZ
0052:        IF(K .EQ. 2 .AND. M .EQ. 1) FZI(I,J,2)=-FZ
0053:        IF(ICALC .NE. 1 .AND. K .EQ. 1) FZ=0.0
0054:        IF(ICALC .NE. 1 .AND. K .EQ. 2) FZ=0.0
0055:        GO TO (105,110,115,120),M
```

```
0056:     105 PVX(I,J,K)=PVX(I,J,K)+FX/6.0
0057:         PVY(I,J,K)=PVY(I,J,K)+FY/6.0
0058:         PVZ(I,J,K)=PVZ(I,J,K)+FZ/6.0
0059:         PX(I,J,K)=PX(I,J,K)+VX(I,J,K)/6.0
0060:         PY(I,J,K)=PY(I,J,K)+VY(I,J,K)/6.0
0061:         PZ(I,J,K)=PZ(I,J,K)+VZ(I,J,K)/6.0
0062:         X(I,J,K)=XI(I,J,K)+H*VX(I,J,K)/2.0
0063:         Y(I,J,K)=YI(I,J,K)+H*VY(I,J,K)/2.0
0064:         Z(I,J,K)=ZI(I,J,K)+H*VZ(I,J,K)/2.0
0065:         VX(I,J,K)=VXI(I,J,K)+H*FX/2.0
0066:         VY(I,J,K)=VYI(I,J,K)+H*FY/2.0
0067:         VZ(I,J,K)=VZI(I,J,K)+H*FZ/2.0
0068:         GO TO 200
0069:     110 PVX(I,J,K)=PVX(I,J,K)+FX/3.0
0070:         PVY(I,J,K)=PVY(I,J,K)+FY/3.0
0071:         PVZ(I,J,K)=PVZ(I,J,K)+FZ/3.0
0072:         PX(I,J,K)=PX(I,J,K)+VX(I,J,K)/3.0
0073:         PY(I,J,K)=PY(I,J,K)+VY(I,J,K)/3.0
0074:         PZ(I,J,K)=PZ(I,J,K)+VZ(I,J,K)/3.0
0075:         X(I,J,K)=XI(I,J,K)+H*VX(I,J,K)/2.0
0076:         Y(I,J,K)=YI(I,J,K)+H*VY(I,J,K)/2.0
0077:         Z(I,J,K)=ZI(I,J,K)+H*VZ(I,J,K)/2.0
0078:         VX(I,J,K)=VXI(I,J,K)+H*FX/2.0
0079:         VY(I,J,K)=VYI(I,J,K)+H*FY/2.0
0080:         VZ(I,J,K)=VZI(I,J,K)+H*FZ/2.0
0081:         GO TO 200
0082:     115 PVX(I,J,K)=PVX(I,J,K)+FX/3.0
0083:         PVY(I,J,K)=PVY(I,J,K)+FY/3.0
0084:         PVZ(I,J,K)=PVZ(I,J,K)+FZ/3.0
0085:         PX(I,J,K)=PX(I,J,K)+VX(I,J,K)/3.0
0086:         PY(I,J,K)=PY(I,J,K)+VY(I,J,K)/3.0
0087:         PZ(I,J,K)=PZ(I,J,K)+VZ(I,J,K)/3.0
0088:         X(I,J,K)=XI(I,J,K)+H*VX(I,J,K)
0089:         Y(I,J,K)=YI(I,J,K)+H*VY(I,J,K)
0090:         Z(I,J,K)=ZI(I,J,K)+H*VZ(I,J,K)
0091:         VX(I,J,K)=VXI(I,J,K)+H*FX
0092:         VY(I,J,K)=VYI(I,J,K)+H*FY
0093:         VZ(I,J,K)=VZI(I,J,K)+H*FZ
0094:         GO TO 200
0095:     120 PVX(I,J,K)=PVX(I,J,K)+FX/6.0
0096:         PVY(I,J,K)=PVY(I,J,K)+FY/6.0
0097:         PVZ(I,J,K)=PVZ(I,J,K)+FZ/6.0
0098:         PX(I,J,K)=PX(I,J,K)+VX(I,J,K)/6.0
0099:         PY(I,J,K)=PY(I,J,K)+VY(I,J,K)/6.0
0100:         PZ(I,J,K)=PZ(I,J,K)+VZ(I,J,K)/6.0
0101:         X(I,J,K)=XI(I,J,K)+H*PX(I,J,K)
0102:         Y(I,J,K)=YI(I,J,K)+H*PY(I,J,K)
0103:         Z(I,J,K)=ZI(I,J,K)+H*PZ(I,J,K)
0104:         VX(I,J,K)=VXI(I,J,K)+H*PVX(I,J,K)
0105:         VY(I,J,K)=VYI(I,J,K)+H*PVY(I,J,K)
0106:         VZ(I,J,K)=VZI(I,J,K)+H*PVZ(I,J,K)
0107:     200 CONTINUE
0108:     900 CONTINUE
0109:         REWIND 8
0110:   C CALCULATE WORK DONE IN INTERVAL.
```

```
0111:          CALL SAVE
0112:          WORK =0.0
0113:          DO 910 J=1,LY
0114:          DO 910 I=1,LX
0115:          CALL SCAN(I,J,1,1,2)
0116:          FZ=-2.0*RE*R*FZ
0117:      910 WORK=WORK+0.5*(FZI(I,J,1)+FZ)*(Z(I,J,1)-ZI(I,J,1))
0118:          DO 919 J=1,LY
0119:          DO 919 I=1,LX
0120:          CALL SCAN(I,J,2,1,2)
0121:          FZ=-2.0*RE*R*FZ
0122:      919 WORK=WORK+0.5*(FZI(I,J,2)+FZ)*(Z(I,J,2)-ZI(I,J,2))
0123:          ECHECK=ECHECK+WORK
0124:          REWIND 8
0125:      920 TAU=TAU+H
0126:          RETURN
0127:          END
```

```
0001:          SUBROUTINE SORT(V,DV)
0002:          COMMON/DIST/NIR(100),NIL(100),NIMAX
0003:          DO 1 J=1,100
0004:          V1=DV*FLOAT(J-1)
0005:          V2=DV*FLOAT(J)
0006:          IF(V1.LE.V.AND.V.LT.V2) GO TO 2
0007:          V1N=DV*FLOAT(1-J)
0008:          V2N=DV*FLOAT(-J)
0009:          IF(V2N.LE.V.AND.V.LT.V1N) GO TO 3
0010:        1 CONTINUE
0011:          GO TO 4
0012:        2 NIR(J)=NIR(J)+1
0013:          IF(J.GT.NIMAX)NIMAX=J
0014:          GO TO 4
0015:        3 NIL(J)=NIL(J)+1
0016:          IF(J.GT.NIMAX)NIMAX=J
0017:        4 CONTINUE
0018:          RETURN
0019:          END
```

```
0001:          SUBROUTINE START
0002:    C     SUBROUTINE ASSIGNS PARTICLES TO INITIAL POSITIONS AND ASSIGNS THEM
0003:    C     RANDOM VELOCITIES ACCORDING TO A MAXWELLIAN DISTRIBUTION WITH MEAN ZERO
0004:    C     AND STANDARD DEVIATION = 1/SQRT(GAMMA).
0005:          COMMON/SIZE/LX,LY,LZ,NP,NTOT,LZ4,LZSEG4
0006:           COMMON/VEL/VX(04,04,300),VY(04,04,300),VZ(04,04,300)
0007:          COMMON/POSIT/X(04,04,300),Y(04,04,300),Z(04,04,300)
0008:          COMMON/MISC/GAMMA,TAU,H,FX,FY,FZ,R,RE,EP,ICALC,ECHECK
0009:    C     CALCULATE INITIAL POSITIONS
0010:          DO 7 K=1,LZ4
0011:          DO 7 J=1,LY
0012:          DO 7 I=1,LX
0013:          X(I,J,K)=FLOAT(I)-1.0+0.5*FLOAT(MOD(K,4)/2)
0014:          Y(I,J,K)=FLOAT(J)-1.0+0.5*FLOAT(MOD(K-1,2))
0015:        7 Z(I,J,K)=0.5*FLOAT((K-1)/2)
0016:    C     ASSIGN INITIAL VELOCITIES
0017:          IRAN=0
0018:          KSCALE=0
0019:          SUMVX=0.
0020:          SUMVY=0.
0021:          SUMVZ=0.
0022:          SUMV2=0.
0023:          DO 8 K=1,LZ4
0024:          DO 8 J=1,LY
0025:          DO 8 I=1,LX
0026:          CALL NRAN31(X1,X2,IRAN)
0027:          VX(I,J,K)=X1
0028:          VY(I,J,K)=X2
0029:          CALL NRAN31(X1,X2,IRAN)
0030:          VZ(I,J,K)=X1
0031:          SUMVX=SUMVX+VX(I,J,K)
0032:          SUMVY=SUMVY+VY(I,J,K)
0033:        8 SUMVZ=SUMVZ+VZ(I,J,K)
0034:    C     ADD VELOCITY INCREMENTS TO INSURE NO NET MOMENTUM
0035:          DVX=-SUMVX/FLOAT(NTOT)
0036:          DVY=-SUMVY/FLOAT(NTOT)
0037:          DVZ=-SUMVZ/FLOAT(NTOT)
0038:          C=1.
0039:        9 CONTINUE
0040:          DO 10 K=1,LZ4
0041:          DO 10 J=1,LY
0042:          DO 10 I=1,LX
0043:          VX(I,J,K)=C*VX(I,J,K)+DVX
0044:          VY(I,J,K)=C*VY(I,J,K)+DVY
0045:          VZ(I,J,K)=C*VZ(I,J,K)+DVZ
0046:          SUMVX=SUMVX+VX(I,J,K)
0047:          SUMVY=SUMVY+VY(I,J,K)
0048:          SUMVZ=SUMVZ+VZ(I,J,K)
0049:       10 SUMV2=SUMV2+VX(I,J,K)**2+VY(I,J,K)**2+VZ(I,J,K)**2
0050:          IF(KSCALE.GT.0)GO TO 11
0051:    C     SCALE VELOCITIES TO GIVE SUMV2=6*NTOT/GAMMA
0052:          C=6.*FLOAT(NTOT)/(GAMMA*SUMV2)
0053:          C=SQRT(C)
0054:          KSCALE=1
0055:          DVX=0.
```

```
0056:         DVY=0.
0057:         DVZ=0.
0058:         SUMVX=0.
0059:         SUMVY=0.
0060:         SUMVZ=0.
0061:         SUMV2=0.
0062:         GO TO 9
0063:      11 EINIT=SUMV2/2.
0064:         WRITE(6,12)
0065:         WRITE(6,13)EINIT,SUMVX,SUMVY,SUMVZ
0066:      12 FORMAT(2X,@DATA FROM SUBROUTINE START@)
0067:      13 FORMAT(2X,@INITIAL VALUES OF KINETIC ENERGY AND NET VELOCITIES@,
0068:        24E14.6)
0069:         RETURN
0070:         END
```

```
0001:         SUBROUTINE THERMO(K1,K2,ETOT)
0002:   C     CALCULATES AVERAGE VALUES OF FLOW VARIABLES BETWEEN PLANES K1 AND K2
0003:         COMMON/POSIT/X(04,04,300),Y(04,04,300),Z(04,04,300)
0004:         COMMON/POSITS/XS(04,04,300),YS(04,04,300),ZS(04,04,300)
0005:         COMMON/STRESS/DISTX,DISTY,DISTZ,DIST,DIST2,C1,PVXX,PVYY,PVZZ
0006:         COMMON/SIZE/LX,LY,LZ,NP,NTOT,LZ4,LZSEG4
0007:          COMMON/VEL/VX(04,04,300),VY(04,04,300),VZ(04,04,300)
0008:         COMMON/MISC/GAMMA,TAU,H,FX,FY,FZ,R,RE,EP,ICALC,ECHECK
0009:   C     CALCULATION OF DENSITY NORMALIZED TO ZERO-TEMPERATURE VALUE.
0010:         DENS=0.
0011:         ZK1=0.
0012:         ZK2=0.
0013:         LZ1=2*K1-1
0014:         LZ2=2*K2
0015:         DO 1 J=1,LY
0016:         DO 1 I=1,LX
0017:         ZK1=ZK1+Z(I,J,LZ1)+Z(I,J,LZ1+1)
0018:       1 ZK2=ZK2+Z(I,J,LZ2)+Z(I,J,LZ2-1)
0019:         DZ=(ZK2-ZK1)/FLOAT(NP)
0020:         T1=.5*FLOAT(K2-K1)
0021:         DENS=T1/DZ
0022:   C     CALCULATION OF AVERAGE VELOCITY.
0023:         VAV=0.
0024:         NK1K2=NP*(K2-K1+1)
0025:         DO 2 K=LZ1,LZ2
0026:         DO 2 J=1,LY
0027:         DO 2 I=1,LX
0028:       2 VAV=VAV+VZ(I,J,K)
0029:         VAV=VAV/FLOAT(NK1K2)
0030:         ETRANS=FLOAT(NK1K2)*VAV**2/2.
0031:   C     CALCULATION OF TEMPERATURES.
0032:         TX=0.
0033:         TY=0.
0034:         TZ=0.
0035:         T=0.
0036:         DO 3 K=LZ1,LZ2
0037:         DO 3 J=1,LY
0038:         DO 3 I=1,LX
0039:         TX=TX+VX(I,J,K)**2
0040:         TY=TY+VY(I,J,K)**2
0041:         VZ1=VZ(I,J,K)-VAV
0042:     *   3 TZ=TZ+VZ1**2
0043:         ETHERM=(TX+TY+TZ)/2.
0044:         TX=TX/FLOAT(NK1K2*3)
0045:         TY=TY/FLOAT(NK1K2*3)
0046:         TZ=TZ/FLOAT(NK1K2*3)
0047:         T=TX+TY+TZ
0048:   C CALCULATE KINETIC CONTRIBUTION TO PRESSURE TENSOR
0049:         PKXXAV=3.0*DENS*TX
0050:         PKYYAV=3.0*DENS*TY
0051:         PKZZAV=3.0*DENS*TZ
0052:   C CALCULATE AVERAGE POTENTIAL ENERGY AND POTENTIAL CONTRIBUTION TO
0053:   C PRESSURE TENSOR
0054:         EPAV=0.
0055:         PVXXAV=0.
```

```
0056:         PVYYAV=0.
0057:         PVZZAV=0.
0058:         CALL SAVE
0059:         DO 4 K=LZ1,LZ2
0060:         DO 4 J=1,LY
0061:         DO 4 I=1,LX
0062:         CALL SCAN(I,J,K,3,1)
0063:         PVXXAV=PVXXAV+R*RE*DENS*PVXX/FLOAT(NK1K2)
0064:         PVYYAV=PVYYAV+R*RE*DENS*PVYY/FLOAT(NK1K2)
0065:         PVZZAV=PVZZAV+R*RE*DENS*PVZZ/FLOAT(NK1K2)
0066:       4 EPAV=EPAV+EP
0067:         REWIND 8
0068:         EPOT=EPAV
0069:         ETOT=ETOT+ETRANS+ETHERM+EPOT
0070:         EPAV=EPAV/FLOAT(NK1K2)
0071:         PXXAV=PKXXAV+PVXXAV
0072:         PYYAV=PKYYAV+PVYYAV
0073:         PZZAV=PKZZAV+PVZZAV
0074:         WRITE(6,99)
0075:         WRITE(6,100)K1,K2,NK1K2
0076:         WRITE(6,101)DENS,VAV,EPAV
0077:         WRITE(6,102)ETRANS,ETHERM,EPOT
0078:         WRITE(6,103) TX,PKXXAV,PVXXAV,PXXAV
0079:         WRITE(6,104) TY,PKYYAV,PVYYAV,PYYAV
0080:         WRITE(6,105) TZ,PKZZAV,PVZZAV,PZZAV
0081:         WRITE(6,106) T
0082:      99 FORMAT(/////2X,@DATA FROM SUBROUTINE THERMO-VALUES AVERAGED BETWE
0083:        2EN PLANES K1 AND K2@)
0084:     100 FORMAT(1H0,@K1=@,I4,2X,@K2=@,I4,3X,@NK1K2=@,I4)
0085:     101 FORMAT(5X,@DENSITY=@,E13.6,3X,@VZAVG=@,E13.6,3X,@EPAVG=@,E13.6)
0086:     102 FORMAT(6X,@ETRANS=@,E13.6,2X,@ETHERM=@,E13.6,4X,@EPOT=@,E13.6)
0087:     103 FORMAT(3X,@TX=@,E13.6,3X,@PKXX=@,E13.6,3X,@PVXX=@,E13.6,3X,@PXX=@
0088:        2,E13.6)
0089:     104 FORMAT(3X,@TY=@,E13.6,3X,@PKYY=@,E13.6,3X,@PVYY=@,E13.6,3X,@PYY=@
0090:        2,E13.6)
0091:     105 FORMAT(3X,@TZ=@,E13.6,3X,@PKZZ=@,E13.6,3X,@PVZZ=@,E13.6,3X,@PZZ=@
0092:        2,E13.6)
0093:     106 FORMAT(3X,@T=@,E13.6)
0094:         RETURN
0095:         END
```

```
0001:          FUNCTION  URAN31(I)
0002:          IF(I)10,11,10
0003:   11     I=11111111
0004:      10  J=I
0005:          J=J*25
0006:          J=J-(J/67108864)*67108864
0007:          J=J*25
0008:          J=J-(J/67108864)*67108864
0009:          J=J*5
0010:          J=J-(J/67108864)*67108864
0011:          A1=J
0012:          I=J
0013:          URAN31=A1/67108864.
0014:          RETURN
0015:          END
```

DISTRIBUTION LIST

No. of Copies	Organization
12	Commander Defense Documentation Center ATTN: DDC-DDA Cameron Station Alexandria, VA 22314
1	Commander US Army Materiel Development and Readiness Command ATTN: DRCDMD-ST 5001 Eisenhower Avenue Alexandria, VA 22333
1	Commander US Army Aviation Research and Development Command ATTN: DRSAV-E P.O. Box 209 St. Louis, MO 63166
1	Director US Army Air Mobility Research and Development Laboratory Ames Research Center Moffett Field, CA 94035
1	Commander US Army Electronics Research and Development Command Technical Support Activity ATTN: DELSD-L Fort Monmouth, NJ 07703
1	Commander US Army Communications Rsch and Development Command ATTN: DRDCO-PPA-SA Fort Monmouth, NJ 07703
2	Commander US Army Missile Research and Development Command ATTN: DRDMI-R DRDMI-YDL Redstone Arsenal, AL 35809
1	Commander US Army Tank Automotive Rsch and Development Command ATTN: DRDTA-UL Warren, MI 48090
2	Commander US Army Armament Research and Development Command ATTN: DRDAR-TSS (2 cys) Dover, NJ 07801
1	Commander US Army Armament Materiel Readiness Command ATTN: DRSAR-LEP-L, Tech Lib Rock Island, IL 61299
2	Commander US Army Armament Research and Development Command ATTN: DRDAR-LCN, Dr. P. Harris DRDAR-LCE, Dr. F. Owens Dover, NJ 07801
1	Director Army Materials & Mechanics Research Center ATTN: Dr. R. Harrison Watertown, MA 02172
1	Director US Army TRADOC Systems Analysis Activity ATTN: ATAA-SL, Tech Lib White Sands Missile Range, NM 88002
3	Commander US Army Research Office ATTN: Dr. M. Ciftan Dr. E. Saibel Dr. J. Chandra P.O. Box 12211 Research Triangle Park, NC 27709

DISTRIBUTION LIST

No. of Copies	Organization
1	Commander Army Research & Standardization (Group) Europe Electronics Branch Dr. Alfred K. Nedoluha Box 65 FPO, NY, NY 09510
1	Director Naval Surface Weapons Center ATTN: Dr. D.J. Pastine White Oak, MD 20910
6	Director Lawrence Livermore Laboratory ATTN: Dr. W. Hoover, Dr. A. Karo Dr. C.M. Tarver, Dr. E.L. Lee, Dr. W. VonHolle, Dr. F.E. Walker Livermore, CA 94550
2	Director Los Alamos Scientific Lab ATTN: Dr. B.L. Holian Dr. G.K. Straub Los Alamos, NM 87544
3	Director National Bureau of Standards ATTN: Dr. H. Prask Dr. S. Trevino Dr. D. Tsai Gaithersburg, MD 20760
1	Director NASA Goddard Space Flight Ctr Mail Code 624 ATTN: Dr. J.E. Allen Greenbelt, MD 20771
1	Science Applications, Inc. ATTN: Mr. S. Howie 6600 Powers Ferry Rd, Suite 220 Atlanta, GA 30339
1	City College of New York Department of Physics ATTN: Professor M. Lax 138th St. & Convent Ave. New York, NY 10031
1	Massachusetts Institute of Technology Dept. of Nuclear Engineering ATTN: Professor S. Yip Cambridge, MA 02139
1	Princeton University Astrophysics Department Professor M.D. Kruskal Princeton, NJ 08540
1	Queens College ATTN: Professor A. Paskin Flushing, NY 11973
1	State University of New York Department of Mechanics ATTN: Professor J. Tasi Stony Brook, NY 11790
2	University of Arizona Department of Mathematics ATTN: Dr. D. McLaughlin Dr. H. Flaschka Tucson, AZ 85721
1	University of California at Irvine Department of Physics ATTN: Dr. A. Maradudin Irvine, CA 92664
1	University of Delaware Department of Physics ATTN: Prof. Fred Williams Newark, DE 19711

DISTRIBUTION LIST

No. of Copies	Organization
2	University of Florida Department of Engineering ATTN: Prof. K.T. Millsaps Prof. B.M. Leadon Gainesville, FL 32603
2	University of Florida Department of Physics and Astronomy ATTN: Prof. J.W. Dufty Prof. C.F. Hooper Gainesville, FL 32603
1	University of Massachusetts Department of Physics ATTN: Professor R. Guyer Amherst, MA 01002
1	University of Nebraska ATTN: Professor J.R. Hardy Lincoln, NE 68588
1	University of Pittsburgh ATTN: Prof. N.J. Zabusky Pittsburgh, PA 15260
1	University of Rochester Department of Physics and Astronomy ATTN: Professor E. Montroll Rochester, NY 14627
1	University of Tennessee Space Institute ATTN: Prof. D.R. Keefer Tullahoma, TN 37388
1	University of Wisconsin Department of Electrical and Computer Engineering ATTN: Prof. A. Scott Madison, WI 53706
1	Washington State University Shock Dynamics Laboratory ATTN: Prof. G. Duvall Pullman, WA 99163

Aberdeen Proving Ground

Dir, USAMSAA
ATT: Dr. J. Sperrazza
DRXSY-MP, H. Cohen
Cdr, USATECOM
ATTN: DRSTE-TO-F
Dir, Wpns Sys Concepts Team
Bldg. E3516, EA
ATTN: DRDAR-ACW