**A CLASSICAL MOLECULAR APPROACH TO COMPUTER
SIMULATION OF BIOLOGICAL SORTING**

by

Donald Greenspan

**Department of Mathematics
University of Texas at Arlington
Arlington, Texas 76019**

Technical Report No. 133

May, 1980

**A CLASSICAL MOLECULAR APPROACH TO COMPUTER
SIMULATION OF BIOLOGICAL SORTING**

by

Donald Greenspan

**Department of Mathematics
University of Texas at Arlington
Arlington, Texas**

ABSTRACT

Steinberg's theory of sorting is modified by replacing the free energy minimization principle with dynamical equations of a molecular nature. Correct cellular sorting then follows in all cases where the mixture is in a liquid or a near-liquid state. Computer examples are described and discussed, primarily for two dimensional, but also for three dimensional, interactions.

1. INTRODUCTION

Mathematical modeling of sorting, or self-reorganization, of biological cells has received wide attention in recent years ([1], [3]-[5], [10]-[13]). Motivation is derived from a broad spectrum of exciting experimental results ([9], [10], [13]). Holtfreter, for example, began with a mixture of individual mesoderm, endoderm, and ectoderm cells and was able to induce self-reorganization into a normal tissue in which the mesoderm cells were interior to the tissue and the endoderm and ectoderm were at the periphery. Weiss and Taylor, more recently, cultured aggregates of cells from different organs and were able to induce organogenesis which approached the complexity of normal organization.

A very popular approach to the mathematical modeling of sorting is that of Steinberg [13], which is founded upon the hypotheses of cellular motility and differential adhesion, which are dominated by the principle of minimization of free energy. Computer implementation by others ([1], [3]-[5], [10]-[12]) have explored diverse dynamical possibilities, including special motility rules, extended zone effects, hexagonal tessellation, viscosity and interface tension. All have failed to establish the viability of the theory.

In all computer simulations to test Steinberg's theory, the fundamental problem considered was as follows. In the plane, consider a collection of cells which consist of only two or three different types. For illustrative purposes, consider a set of two types of cells, say, type A and type B. The initial positions of the cells are selected at random within a given square or rectangle. Confining all motions to be within the

given square or rectangle and using local cell interaction rules only, one must then induce the type A cells to organize into a central core and the type B cells to organize into a layer around this core.

In this paper, it is our intent to modify Steinberg's theory in a natural and suitable fashion so that self-reorganization always follows in the desired way. We will replace his gross principle of free energy minimization, which implies nothing about the mechanics of cellular interaction, by classical molecular-type force formulas, which do.

2. CLASSICAL MOLECULAR MECHANICS

For purposes of intuition, it will be important to review, first, how molecules interact.

Within a larger body, molecules interact only locally, that is, only with their nearest neighbors. This interaction is of the following general nature [2]. If two molecules are pushed together they repel each other, if pulled apart they attract each other, and mutual repulsion is of a greater order of magnitude than is mutual attraction. Mathematically, this behavior is often formulated as follows [9]. The magnitude F of the force $\vec{F}$ between two molecules which are locally r units apart is of the form

$$F = - \frac{G}{r^p} + \frac{H}{r^q} , \quad (2.1)$$

where, typically,

$$G > 0, \quad d > 0, \quad q > p \geq 7 . \quad (2.2)$$

The major problem in any simulation of a physical body is that there are too many component molecules to incorporate into the model. The classical mathematical approach is to replace the large, but finite, number of molecules by an infinite set of points. In so doing, the rich physics of molecular interaction is lost because every point always has an infinite number of neighbors which are arbitrarily close. A viable computer alternative is to replace the large number of molecules by a much smaller number of particles and then readjust the parameters in (2.1) to compensate ([6], [14]). It is this latter approach which we will follow.

3. THE COMPUTER ALGORITHM

The general idea outlined in Section 2 will be implemented in the following, constructive fashion.

Consider N particles P_i , $i = 1, 2, \dots, N$. For $\Delta t > 0$, let $t_k = k\Delta t$, $k = 0, 1, 2, \dots$. For each of $i = 1, 2, \dots, N$, let m_i denote the adhesive measure of P_i , and let P_i at t_k be located at $\vec{r}_{i,k} = (x_{i,k}, y_{i,k}, z_{i,k})$, have velocity $\vec{v}_{i,k} = (v_{i,k,x}, v_{i,k,y}, v_{i,k,z})$, and have acceleration $\vec{a}_{i,k} = (a_{i,k,x}, a_{i,k,y}, a_{i,k,z})$. Let position, velocity, and acceleration be related by the recursion formulas

$$\vec{v}_{i,1/2} = \vec{v}_{i,0} + \frac{1}{2}(\Delta t)\vec{a}_{i,0}, \quad (\text{starter formula}) \quad (3.1)$$

$$\vec{v}_{i,k+1/2} = \vec{v}_{i,k-1/2} + (\Delta t)\vec{a}_{i,k}, \quad k = 1, 2, 3, \dots \quad (3.2)$$

$$\vec{r}_{i,k+1} = \vec{r}_{i,k} + (\Delta t)\vec{v}_{i,k+1/2}, \quad k = 0, 1, 2, \dots \quad (3.3)$$

At t_k , let the force acting on P_i be $\vec{F}_{i,k} = (F_{i,k,x}, F_{i,k,y}, F_{i,k,z})$.

We relate force and acceleration by the dynamical equation

$$\vec{F}_{i,k} = m_i \vec{a}_{i,k} . \quad (3.4)$$

As soon as the precise structure of $\vec{F}_{i,k}$ is given, the motion of each P_i will be determined explicitly and recursively by (3.1)-(3.4) from given initial data. The force $\vec{F}_{i,k}$ is described now as follows.

Let $\vec{r}_{ij,k}$ be the vector from P_i to P_j at time t_k , so that $r_{ij,k} = |\vec{r}_{i,k} - \vec{r}_{j,k}|$ is the distance between the two particles. Then the force $\vec{F}_{ij,k}$ on P_i exerted by P_j at time t_k is assumed to be

$$\vec{F}_{ij,k} = \left(-\frac{G_{ij}}{(r_{ij,k})^p} + \frac{H_{ij}}{(r_{ij,k})^q} \frac{\vec{r}_{ij,k}}{r_{ij,k}} \right) , \quad (3.5)$$

which is consistent with (2.1). The total force $\vec{F}_{i,k}$ on P_i due to all other particles different from P_i is defined by

$$\vec{F}_{i,k} = \sum_{\substack{j=1 \\ j \neq i}}^N \left\{ \left(-\frac{G_{ij}}{(r_{ij,k})^p} + \frac{H_{ij}}{(r_{ij,k})^q} \frac{\vec{r}_{ij,k}}{r_{ij,k}} \right) \right\} . \quad (3.6)$$

Note finally that the introduction of an additional parameter D is essential to assure that particle interactions are local. We will require that whenever $r_{ij,k} > D$, then (3.5) must be replaced with

$$\vec{F}_{ij,k} = \vec{0} \quad (r_{ij,k} > D) . \quad (3.7)$$

Note that the vector form of equations (3.1)-(3.7) makes them applicable in any number of dimensions.

For the convenience of the reader, typical FORTRAN programs of the algorithm described above, in both two and three space dimensions, are given in Appendices 1 and 2 of Greenspan [8].

4. COMPUTER EXAMPLES

Since, in general, particles do not adhere when in a gaseous state and are rigid when in a solid state, sorting can only occur in a liquid, or near-liquid, state. Relative to this observation, the following preliminary remarks will be important.

For the time step $\Delta t = 0.0001$, in (3.5) set

$$p = 4, \quad q = 6 \quad (4.1)$$

$$G_{ij} = H_{ij} = 5m_i m_j \quad (4.2)$$

Let the local interaction parameter be given by

$$D = 2.3 \quad (4.3)$$

Then, if P_i is to be a liquid particle, the range of the speed v_i of P_i can be deduced for various adhesive measures m_i [7]. In particular,

$$m_i = 2000 \text{ implies } 100 < v_i < 170 \quad (4.4)$$

$$m_i = 4000 \text{ implies } 90 < v_i < 160 \quad (4.5)$$

$$m_1 = 6000 \text{ implies } 78 \leq v_1 \leq 140 \quad (4.6)$$

$$m_1 = 8000 \text{ implies } 65 \leq v_1 \leq 110 \quad (4.7)$$

$$m_1 = 10000 \text{ implies } 50 \leq v_1 \leq 80 \quad (4.8)$$

The parameter choices (4.1)-(4.3) will be assumed in all examples to be described, while choices of initial data will be guided by (4.4)-(4.8).

For our first example of sorting, we will consider planar motion within a square. Consider a set of 81 particles, 25 of which have adhesive measure $m_1 = 6000$ and 56 of which have adhesive measure $m_1 = 2000$. For convenience, let $P_1 - P_{25}$ each have $m_1 = 6000$, while $P_{26} - P_{81}$ each have $m_1 = 2000$. Next consider choosing initial positions for $P_1 - P_{81}$ interior to the square whose vertices are at $(5,5)$, $(-5,5)$, $(-5,-5)$, $(5,-5)$, and whose edge has length 10. For this purpose, set $\Delta x = \Delta y = 1$ and consider the 81 grid points which are interior to the square. Then each particle is assigned a distinct grid point position by a random process. The resulting positions are listed in Table 1 and are shown graphically in Figure 4.1 (a), where each shaded circle represents a particle for which $m_1 = 6000$, while each unshaded circle represents a particle for which $m_1 = 2000$. The particles $P_1 - P_{25}$ are then each assigned a speed of 70 while $P_{26} - P_{81}$ are each assigned a speed of 90. The velocity of each particle is fixed next, by a random process, in either the x or the y direction. The resulting initial velocities are also listed in Table 1.

As a final consideration, note that viscosity is an essential element

in liquid motion. Internal viscosity is accounted for by (3.6). Viscosity along the sides of the square is included in the following way. When a particle impacts on a side of the square, it will be reflected in the usual symmetric fashion, but viscous damping will be included by setting

$$v_r = (0.95)v_i, \quad (4.9)$$

where v_i is the incident speed, v_r is the reflected speed, and 0.95 is the damping coefficient.

The resulting dynamical behavior was generated from (3.1)-(3.7) on the DEC-20 system at the University of Texas. The results are shown in Figures 4.1 (b)-(f) at the respective times t_{2000} , t_{4000} , t_{8000} , t_{12000} , t_{24000} . In (b), at t_{2000} , clustering has begun and as it continues, as shown in (c), at t_{4000} , it is apparent that $P_1 - P_{25}$ are beginning to move towards each other. In (d), at t_{8000} , all but two of $P_1 - P_{25}$ have formed into a loose core while $P_{26} - P_{81}$ are still clustering, but now only with each other. In (e), at t_{12000} , $P_1 - P_{25}$ have formed a loose core while $P_{26} - P_{81}$ are beginning to form an outer layer around the core. In (f), at t_{24000} , the core has become relatively dense and the innermost portion of the outer layer is now firmly formed. The entire outer layer will become dense only when the wall viscosity reduces the high speeds of the outermost particles sufficiently to promote close adhesion. Since the amount of computer time necessary to induce this final effect can be relatively large [7], the calculations were terminated at t_{24000} , which required a total of three hours of computer time.

TABLE 1

i	x	y	vx	vy	i	x	y	vx	vy	i	x	y	vx	vy
1	1	-4	70	0	28	4	-3	-90	0	55	2	2	90	0
2	-4	-1	-70	0	29	-4	3	0	-90	56	-3	3	0	-90
3	1	0	0	70	30	4	0	0	90	57	4	-2	0	90
4	2	4	0	-70	31	-2	0	-90	0	58	0	4	90	0
5	-4	-4	0	70	32	-1	-4	90	0	59	-2	-2	0	-90
6	2	0	-70	0	33	-4	0	0	-90	60	4	1	-90	0
7	-1	4	0	70	34	2	-3	0	-90	61	0	-4	0	90
8	3	-1	-70	0	35	-3	2	90	0	62	-2	-1	90	0
9	4	2	-70	0	36	-1	2	0	90	63	2	-4	0	-90
10	-3	0	0	70	37	0	0	0	-90	64	-4	4	0	-90
11	1	1	-70	0	38	1	-3	-90	0	65	2	3	90	0
12	4	-1	0	70	39	-4	-3	0	90	66	-3	4	0	90
13	-2	-4	0	70	40	4	-4	0	-90	67	3	0	0	-90
14	2	-2	-70	0	41	-2	-3	90	0	68	-1	-3	-90	0
15	-2	1	0	70	42	3	-2	0	90	69	-4	-2	0	90
16	3	1	0	-70	43	1	2	0	-90	70	3	4	0	90
17	-4	2	0	70	44	-3	-3	0	-90	71	-3	-2	0	90
18	1	3	70	0	45	-1	0	90	0	72	0	1	0	-90
19	0	-1	-70	0	46	0	-2	0	90	73	-1	1	-90	0
20	-1	-2	0	70	47	-2	2	-90	0	74	0	-3	0	-90
21	2	1	-70	0	48	2	-1	-90	0	75	4	4	90	0
22	-1	3	0	70	49	-3	1	0	90	76	1	-1	-90	0
23	3	-4	0	70	50	3	-3	-90	0	77	-3	-1	0	90
24	1	-2	-70	0	51	0	2	0	-90	78	3	3	-90	0
25	-3	-4	0	70	52	-4	1	0	90	79	1	4	0	-90
26	0	3	0	90	53	3	2	90	0	80	-2	4	90	0
27	-1	-1	90	0	54	-2	3	-90	0	81	4	3	-90	0

(a) $t=t_0$ (d) $t=t_{8000}$ (b) $t=t_{2000}$ (e) $t=t_{12000}$ (c) $t=t_{4000}$ (f) $t=t_{24000}$

FIGURE 4.1

For our second example of sorting, we will consider again the planar motion of 81 particles within the same square as in the first example. The initial positions are the same as those of Table 1. However, this time let $P_1 - P_{15}$ have $m_1 = 10000$, let $P_{16} - P_{40}$ have $m_1 = 6000$, and let $P_{41} - P_{81}$ have $m_1 = 2000$, so that three sets will have to self-reorganize. Initially, the particles $P_1 - P_{15}$ are assigned speeds of 80, the particles $P_{16} - P_{40}$ are assigned speeds of 90, and the particles $P_{41} - P_{81}$ are assigned speeds of 100. The initial velocities are determined in an x or a y direction at random, as in the first example, and the results are given in Table 2. The initial particle configuration is shown in Figure 4.2 (a), where the shaded circles represent particles with $m_1 = 10000$, the unshaded circles represent particles with $m_1 = 6000$, and the shaded triangles represent particles with $m_1 = 2000$. Viscous damping at the walls is again implemented by (4.9).

The resulting dynamical behavior is shown in Figures 4.2 (b)-(f) at the respective times t_{3000} , t_{7000} , t_{12000} , t_{18000} , t_{32000} . In (b), at t_{3000} , clustering has begun. In (c), at t_{7000} , $P_1 - P_{15}$ show distinct movement toward each other, while $P_{41} - P_{81}$ show distinct motion outwards, and these continue through t_{12000} , as shown in (d). In (e), at t_{18000} , $P_1 - P_{15}$ have formed a dense core and $P_{16} - P_{40}$ have formed a relatively loose layer around the core. In (f), at t_{32000} , $P_{16} - P_{40}$ have now formed a dense layer around the core particles $P_1 - P_{15}$, and $P_{41} - P_{81}$ have formed a layer around $P_{16} - P_{40}$. Again, the computer time necessary for attaining a high density for this second layer did not warrant continuation past t_{32000} . The total computer time through t_{32000} was 4 hours.

TABLE 2

i	vx	vy	i	vx	vy	i	vx	vy	i	vx	vy
1	80	0	21	-90	0	41	100	0	61	0	100
2	-80	0	22	0	90	42	0	100	62	100	0
3	0	80	23	0	90	43	0	-100	63	0	-100
4	0	-80	24	-90	0	44	0	-100	64	0	-100
5	0	80	25	0	90	45	100	0	65	100	0
6	-80	0	26	0	90	46	0	100	66	0	100
7	0	80	27	90	0	47	-100	0	67	0	-100
8	-80	0	28	-90	0	48	-100	0	68	-100	0
9	-80	0	29	0	-90	49	0	100	69	0	100
10	0	80	30	0	90	50	-100	0	70	0	100
11	-80	0	31	-90	0	51	0	-100	71	0	100
12	0	80	32	90	0	52	0	100	72	0	-100
13	0	80	33	0	-90	53	100	0	73	-100	0
14	-80	0	34	0	-90	54	-100	0	74	0	-100
15	0	80	35	90	0	55	100	0	75	100	0
16	0	-90	36	0	90	56	0	-100	76	-100	0
17	0	90	37	0	-90	57	0	100	77	0	100
18	90	0	38	-90	0	58	100	0	78	-100	0
19	-90	0	39	0	90	59	0	-100	79	0	-100
20	0	90	40	0	-90	60	-100	0	80	100	0
									81	-100	0

(a) $t=t_0$ (d) $t=t_{12000}$ (b) $t=t_{3000}$ (e) $t=t_{18000}$ (c) $t=t_{7000}$ (f) $t=t_{32000}$

FIGURE 4.2

Variations in the parameters of the two examples described above were studied next in a number of additional computer examples, the results of which are summarized as follows. An increase in initial speeds decreased the rate of sorting, as did an increase in the wall damping factor to 0.99. When the choices of m_1 were close to each other, a longer time was required for sorting than when the choices of m_1 were more diverse. Decreasing initial speeds to half of those prescribed in (4.4)-(4.8) yielded early solidification and particle trapping, so that sorting could not be achieved. This same effect was attained by decreasing the wall damping factor radically.

Finally, note that several three dimensional examples, similar to the first example in this section, but set in a cube with only 64 particles, were also run. The results were entirely comparable to those of the first example. In three dimensions, the only problems encountered were the usual ones associated with the graphical display of the output. The program used and a listing of typical initial data are available in Appendices 2 and 3 of Greenspan [8].

5. REMARK

It should be observed that we and others have not included gravity into our models, though we could have. In general, gravity plays an essential role in fluid motions. However, in modeling sorting, the intermolecular type forces dominate gravity to the extent that the major effect of the latter force is to insure that the liquid under study keeps the shape of its container. However, this property was simulated by

having a closed top to our square boundary. If the top of the square had been left open, to form what is known as an open cavity, then it would have been essential to include the effects of gravity in (3.6).

REFERENCES

- [1] P. Antonelli, T. D. Rogers, and M. Willard, "Cell aggregation kinetics", *J. Theor. Biol.*, 41, 1973, pp. 1-21.
- [2] R. P. Feynman, R. B. Leighton, and M. Sands, *The Feynman Lectures on Physics*, Addison-Wesley, Reading, Mass., 1963.
- [3] N. Goel, R. D. Campbell, R. D. Gordon, R. Rosen, H. Martinez, and M. Ycas, "Self sorting of isotropic cells", *J. Theor. Biol.*, 28, 1970, pp. 423-468.
- [4] N. S. Goel and A. G. Leith, "Self-sorting of anisotropic cells", *J. Theor. Biol.*, 28, 1970, pp. 469-482.
- [5] R. Gordon, N. S. Goel, M. S. Steinberg, and L. L. Wiseman, "A rheological mechanism sufficient to explain the kinetics of cell sorting", *J. Theor. Biol.*, 37, 1972, pp. 43-73.
- [6] D. Greenspan, *Discrete Models*, Addison-Wesley, Reading, Mass., 1973.
- [7] D. Greenspan, "Computer studies of interactions of particles with differing masses", *J. Comp and Appl. Math.*, 3, 1977, pp. 145-154.
- [8] D. Greenspan, "A classical molecular approach to computer biological sorting", Tech. Rept. # 133, Dept. of Math., Univ. of Texas at Arlington, Arlington, Texas 76019.
- [9] J. O. Hirschfelder, C. F. Curtiss, and R. B. Bird, *Molecular Theory of Gases and Liquids*, Wiley, New York, 1954.

- [10] J. Holtfreter, "A study of the mechanics of gastrulation", *J. Exp. Biol.*, 94, 1943, pp. 261-318.
- [11] A. G. Leith and N. S. Goel, "Simulation of movement of cells during self sorting", *J. Theor. Biol.*, 33, 1971, pp. 171-188.
- [12] G. D. Mostow (Editor), *Mathematical Models for Cell Rearrangement*, Yale Univ. Press, New Haven, Conn., 1975.
- [13] M. S. Steinberg, "Reconstructing of tissues by dissociated cells", *Science*, 141, 1963, pp. 401-408.
- [14] J. von Neumann, "Proposal and analysis of a new treatment of hydrodynamical shock problems", in *The Collected Works of John von Neumann*, Vol. VI, No. 27, Pergamon, 1963.

C THIS IS SORT5 (SORTING OR SELF-REORGINATION)

DOUBLE PRECISION XO(81),YO(81),VXO(81),VYO(81),PMASS(81),XKE,
IX(81,2),Y(81,2),VX(81,2),VY(81,2),ACX(81),ACY(81),R2,R,FX,FY,F
C VARIABLES XO, ETC, HAVE BEEN PUNCHED OR, NOT ZERO.

OPEN (UNIT=7,FILE='SORT5.OUT',DEVICE='DSK')

OPEN (UNIT=21,FILE='SORT5.DAT')

OPEN (UNIT=5,DEVICE='LPT',FILE='SORT5.PRT',ACCESS='SEQUOT')

KPRINT=1000

DO 4 I=1,81

X(I,1)=0.

Y(I,1)=0.

VX(I,1)=0.

VY(I,1)=0.

X(I,2)=0.

Y(I,2)=0.

VX(I,2)=0.

VY(I,2)=0.

ACX(I)=0.

ACY(I)=0.

4 CONTINUE

C THE FIRST 15 PARTICLES ARE HEAVY.

DO 5 I=1,15

PMASS(I)=10000.

5 CONTINUE

DO 6 I=16,40

PMASS(I)=6000.

6 CONTINUE

DO 7 I=41,81

PMASS(I)=2000.

7 CONTINUE

C INPUT CARDS SEPARATE DATA BY 12 SPACES.

READ (21,10) (XO(I),YO(I),VXO(I),VYO(I),I=1,81)

10 FORMAT (2F12.5,2F12.6)

NM1= 80

K=1

DO 20 I=1,81

WRITE (5,15) I,XO(I),YO(I),VXO(I),VYO(I)

15 FORMAT (5X,15,2F12.5,2F12.6)

20 CONTINUE

C UPDATE

DO 30 I=1,81

X(I,1)=XO(I)

Y(I,1)=YO(I)

VX(I,1)=VXO(I)

VY(I,1)=VYO(I)

30 CONTINUE

GO TO 71

C RESET INITIAL DATA

65 DO 70 I=1,81

X(I,1)=X(I,2)

Y(I,1)=Y(I,2)

VX(I,1)=VX(I,2)

VY(I,1)=VY(I,2)

70 CONTINUE

C CALCULATION OF ACCELERATIONS

DO 701 I=1,81

ACX(I)=0.

ACY(I)=0.

701 CONTINUE

71 DO 77 I=1, 80

```

      IP1=I+1
      DO 76 J=IP1,81
      R2=(X(I,1)-X(J,1))**2+(Y(I,1)-Y(J,1))**2
      R=DSQRT(R2)
      IF (R.GT.2.3) GO TO 73
      F=PMASS(I)*PMASS(J)*((-5./R**4)+(5./R**6))
      GO TO 74
73     F=0.
74     FX=F*(X(I,1)-X(J,1))/R
      FY=F*(Y(I,1)-Y(J,1))/R
75     ACX(I)=ACX(I)+FX
      ACX(J)=ACX(J)-FX
      ACY(I)=ACY(I)+FY
      ACY(J)=ACY(J)-FY
76     CONTINUE
77     CONTINUE
      DO 770 I=1,81
      ACX(I)=ACX(I)/PMASS(I)
      ACY(I)=ACY(I)/PMASS(I)
770    CONTINUE
79     DO 80 I=1,81
      VX(I,2)=VX(I,1)+0.0001*ACX(I)
      VY(I,2)=VY(I,1)+0.0001*ACY(I)
      X(I,2)=X(I,1)+0.0001*VX(I,2)
      Y(I,2)=Y(I,1)+0.0001*VY(I,2)
80     CONTINUE
C NEXT IS REFLECTION FROM THE WALLS.
      DO 801 I=1,81
      IF (X(I,2).LT.5.0) GO TO 8001
      X(I,2)=10.0-X(I,2)
      VX(I,2)=-.95*VX(I,2)
      VY(I,2)=.95*VY(I,2)
8001    IF (X(I,2).GT.-5.0) GO TO 8002
      X(I,2)=-10.0-X(I,2)
      VX(I,2)=-.95*VX(I,2)
      VY(I,2)=.95*VY(I,2)
8002    IF (Y(I,2).LT.5.0) GO TO 8003
      Y(I,2)=10.0-Y(I,2)
      VY(I,2)=-.95*VY(I,2)
      VX(I,2)=.95*VX(I,2)
8003    IF (Y(I,2).GT.-5.0) GO TO 801
      Y(I,2)=-10.0-Y(I,2)
      VY(I,2)=-.95*VY(I,2)
      VX(I,2)=.95*VX(I,2)
801     CONTINUE
8043    K=K+1
C PRINT EVERY KPRINT STEPS.
      IF (MOD(K,KPRINT).GT.0) GO TO 82
      DO 810 I=1,81
      WRITE (5,81) K,I,X(I,2),Y(I,2),VX(I,2),VY(I,2)
81     FORMAT (5X,11Z,15,4F18.9)
810    CONTINUE
C CALCULATE KE BUT NORMALIZE WITH MASS DIVIDED BY 10**6.
      XKE=0.
      DO 8101 I=1,81
      XKE=XKE+0.5*PMASS(I)*(1./1000000.)*(VX(I,2)**2+VY(I,2)**2)
8101   CONTINUE
      WRITE (5,8102) XKE
8102   FORMAT (2X,F18.9)
C TERMINATION AFTER A FIXED NUMBER OF STEPS

```

```
82 IF (K.LT.8000) GO TO 65  
WRITE (7,10) (X(I,2),Y(I,2),VX(I,2),VY(I,2),I=1,81)  
STOP  
END
```

C THIS IS SORT8 (SORTING OR SELF-REORGINATION)

DOUBLE PRECISION XO(64),YO(64),VXO(64),VYO(64),PMASS(64),XKE,
IX(64,2),Y(64,2),VX(64,2),VY(64,2),ACX(64),ACY(64),R2,R,FX,FY,F,
IZO(64),VZO(64),Z(64,2),VZ(64,2),ACZ(64),FZ

C VARIABLES XO, ETC, HAVE BEEN PUNCHED ON, NOT ZERO.

OPEN (UNIT=7,FILE='SORT8.OUT',DEVICE='DSK')

OPEN (UNIT=21,FILE='SORT8.DAT')

OPEN (UNIT=5,DEVICE='LPT',FILE='SORT8.PRT',ACCESS='SEQUOUT')

KPRINT=1000

GRAVIT=0.0

DO 4 I=1,64

X(I,1)=0.

Y(I,1)=0.

Z(I,1)=0.

VX(I,1)=0.

VY(I,1)=0.

VZ(I,1)=0.

X(I,2)=0.

Y(I,2)=0.

Z(I,2)=0.

VX(I,2)=0.

VY(I,2)=0.

VZ(I,2)=0.

ACX(I)=0.

ACY(I)=0.

ACZ(I)=0.

4 CONTINUE

C THE FIRST 20 PARTICLES ARE HEAVY.

DO 5 I=1,20

PMASS(I)=6000.

5 CONTINUE

DO 6 I=21,64

PMASS(I)=2000.

6 CONTINUE

C INPUT CARDS SEPARATE DATA BY 12 SPACES.

READ (21,10) (XO(I),YO(I),ZO(I),VXO(I),VYO(I),VZO(I),I=1,64)

10 FORMAT (3F12.5,3F12.6)

NM1= 63

K=1

DO 20 I=1,64

WRITE (5,15) I,XO(I),YO(I),ZO(I),VXO(I),VYO(I),VZO(I)

15 FORMAT (5X,15,3F12.5,3F12.6)

20 CONTINUE

C UPDATE

DO 30 I=1,64

X(I,1)=XO(I)

Y(I,1)=YO(I)

Z(I,1)=ZO(I)

VX(I,1)=VXO(I)

VY(I,1)=VYO(I)

VZ(I,1)=VZO(I)

30 CONTINUE

GO TO 71

C RESET INITIAL DATA

65 DO 70 I=1,64

X(I,1)=X(I,2)

Y(I,1)=Y(I,2)

Z(I,1)=Z(I,2)

VX(I,1)=VX(I,2)

VY(I,1)=VY(I,2)

```

      VZ(I,1)=VZ(I,2)
70      CONTINUE
C CALCULATION OF ACCELERATIONS
      DO 701 I=1,64
        ACX(I)=0.
        ACY(I)=0.
        ACZ(I)=0.
701      CONTINUE
71      DO 77 I=1, 63
        IP1=I+1
        DO 76 J=IP1,64
          R2=(X(I,1)-X(J,1))**2+(Y(I,1)-Y(J,1))**2+(Z(I,1)-Z(J,1))**2
          R=DSQRT(R2)
          IF (R.GT.1.5) GO TO 73
          F=PMASS(I)*PMASS(J)*((-5./R**4)+(5./R**6))
          GO TO 74
73          F=0.
74          FX=F*(X(I,1)-X(J,1))/R
          FY=F*(Y(I,1)-Y(J,1))/R
          FZ=F*(Z(I,1)-Z(J,1))/R
75          ACX(I)=ACX(I)+FX
          ACX(J)=ACX(J)-FX
          ACY(I)=ACY(I)+FY
          ACY(J)=ACY(J)-FY
          ACZ(I)=ACZ(I)+FZ
          ACZ(J)=ACZ(J)-FZ
76          CONTINUE
77          CONTINUE
        DO 770 I=1,64
          ACX(I)=ACX(I)/PMASS(I)
          ACY(I)=ACY(I)/PMASS(I)
          ACZ(I)=(ACZ(I)/PMASS(I))-32.0*GRAVIT
770        CONTINUE
79      DO 80 I=1,64
        VX(I,2)=VX(I,1)+0.0001*ACX(I)
        VY(I,2)=VY(I,1)+0.0001*ACY(I)
        VZ(I,2)=VZ(I,1)+0.0001*ACZ(I)
        X(I,2)=X(I,1)+0.0001*VX(I,2)
        Y(I,2)=Y(I,1)+0.0001*VY(I,2)
        Z(I,2)=Z(I,1)+0.0001*VZ(I,2)
80      CONTINUE
C NEXT IS REFLECTION FROM THE WALLS.
      DO 801 I=1,64
        IF (X(I,2).LT.2.5) GO TO 8001
        X(I,2)=5.0-X(I,2)
        VX(I,2)=-.95*VX(I,2)
        VY(I,2)=.95*VY(I,2)
        VZ(I,2)=.95*VZ(I,2)
8001      IF (X(I,2).GT.-2.5) GO TO 8002
        X(I,2)=-5.0-X(I,2)
        VX(I,2)=-.95*VX(I,2)
        VY(I,2)=.95*VY(I,2)
        VZ(I,2)=.95*VZ(I,2)
8002      IF (Y(I,2).LT.2.5) GO TO 8003
        Y(I,2)=5.0-Y(I,2)
        VY(I,2)=-.95*VY(I,2)
        VX(I,2)=.95*VX(I,2)
        VZ(I,2)=.95*VZ(I,2)
8003      IF (Y(I,2).GT.-2.5) GO TO 8004
        Y(I,2)=-5.0-Y(I,2)

```

```

      VY(1,2)=-.95*VY(1,2)
      VX(1,2)=.95*VX(1,2)
      VZ(1,2)=.95*VZ(1,2)
8004  IF (Z(1,2).LT.2.5) GO TO 8005
      Z(1,2)=5.0-Z(1,2)
      VZ(1,2)=-.95*VZ(1,2)
      VX(1,2)=.95*VX(1,2)
      VY(1,2)=.95*VY(1,2)
8005  IF (Z(1,2).GT.-2.5) GO TO 801
      Z(1,2)=-5.0-Z(1,2)
      VZ(1,2)=-.95*VZ(1,2)
      VX(1,2)=.95*VX(1,2)
      VY(1,2)=.95*VY(1,2)
801   CONTINUE
8043  K=K+1
C PRINT EVERY KPRINT STEPS.
      IF (MOD(K,KPRINT).GT.0) GO TO 82
      DO 810 I=1,64
        WRITE (5,81) K,I,X(1,2),Y(1,2),Z(1,2),VX(1,2),VY(1,2),VZ(1,2)
81    FORMAT (5X,I12,15,6F18.9)
810   CONTINUE
C CALCULATE KE BUT NORMALIZE WITH MASS DIVIDED BY 10**6.
      XKE=0.
      DO 8101 I=1,64
        XKE=XKE+0.5*PMASS(I)*(1./1000000.)*(VX(I,2)**2+VY(I,2)**2
        1+VZ(I,2)**2)
8101  CONTINUE
      WRITE (5,8102) XKE
8102  FORMAT (2X,F18.9)
C TERMINATION AFTER A FIXED NUMBER OF STEPS
82    IF (K.LT.8000) GO TO 65
      WRITE (7,10) (X(1,2),Y(1,2),Z(1,2),VX(1,2),VY(1,2),VZ(1,2),
      1I=1,64)
      STOP
      END

```

i	x	y	z	vx	vy	vz
1	1.50000	-0.50000	0.50000	0.000000	-90.000000	0.000000
2	0.50000	0.50000	-1.50000	0.000000	0.000000	90.000000
3	0.50000	0.50000	0.50000	-90.000000	0.000000	0.000000
4	1.50000	0.50000	1.50000	90.000000	0.000000	0.000000
5	-0.50000	-0.50000	-0.50000	0.000000	0.000000	90.000000
6	1.50000	-1.50000	-0.50000	0.000000	0.000000	90.000000
7	-1.50000	-1.50000	1.50000	0.000000	-90.000000	0.000000
8	-0.50000	-0.50000	-1.50000	-90.000000	0.000000	0.000000
9	-0.50000	1.50000	0.50000	0.000000	0.000000	-90.000000
10	1.50000	1.50000	0.50000	0.000000	0.000000	90.000000
11	-1.50000	1.50000	-1.50000	-90.000000	0.000000	0.000000
12	1.50000	1.50000	-0.50000	0.000000	90.000000	0.000000
13	-1.50000	1.50000	1.50000	0.000000	0.000000	-90.000000
14	1.50000	0.50000	-0.50000	90.000000	0.000000	0.000000
15	0.50000	-0.50000	0.50000	0.000000	0.000000	90.000000
16	1.50000	-0.50000	-0.50000	0.000000	-90.000000	0.000000
17	0.50000	-0.50000	1.50000	0.000000	-90.000000	0.000000
18	1.50000	1.50000	-1.50000	90.000000	0.000000	0.000000
19	-0.50000	-1.50000	0.50000	0.000000	0.000000	90.000000
20	0.50000	0.50000	1.50000	0.000000	-90.000000	0.000000
21	0.50000	-1.50000	-1.50000	100.000000	0.000000	0.000000
22	0.50000	-1.50000	-0.50000	0.000000	100.000000	0.000000
23	0.50000	1.50000	0.50000	0.000000	-100.000000	0.000000
24	0.50000	-0.50000	-0.50000	-100.000000	0.000000	0.000000
25	0.50000	1.50000	1.50000	0.000000	0.000000	100.000000
26	1.50000	0.50000	-1.50000	0.000000	100.000000	0.000000
27	-1.50000	-1.50000	-1.50000	0.000000	0.000000	-100.000000
28	-0.50000	0.50000	0.50000	0.000000	100.000000	0.000000
29	0.50000	0.50000	-0.50000	0.000000	-100.000000	0.000000
30	-1.50000	-0.50000	0.50000	-100.000000	0.000000	0.000000
31	-1.50000	-0.50000	-1.50000	0.000000	0.000000	100.000000
32	-0.50000	-0.50000	1.50000	100.000000	0.000000	0.000000
33	0.50000	1.50000	-0.50000	100.000000	0.000000	0.000000
34	-0.50000	0.50000	1.50000	0.000000	100.000000	0.000000
35	-1.50000	1.50000	-0.50000	-100.000000	0.000000	0.000000
36	-0.50000	1.50000	1.50000	0.000000	-100.000000	0.000000
37	1.50000	-1.50000	0.50000	0.000000	100.000000	0.000000
38	1.50000	-0.50000	1.50000	0.000000	0.000000	-100.000000
39	1.50000	0.50000	0.50000	0.000000	0.000000	100.000000
40	0.50000	-1.50000	1.50000	-100.000000	0.000000	0.000000
41	-0.50000	-1.50000	-1.50000	0.000000	0.000000	100.000000
42	-0.50000	1.50000	-0.50000	-100.000000	0.000000	0.000000
43	-0.50000	-1.50000	-0.50000	0.000000	100.000000	0.000000
44	-0.50000	-1.50000	1.50000	0.000000	0.000000	-100.000000
45	-0.50000	0.50000	-1.50000	100.000000	0.000000	0.000000
46	1.50000	1.50000	1.50000	0.000000	100.000000	0.000000
47	-1.50000	-0.50000	-0.50000	-100.000000	0.000000	0.000000
48	1.50000	-1.50000	1.50000	0.000000	0.000000	-100.000000
49	-1.50000	-1.50000	0.50000	0.000000	-100.000000	100.000000
50	-1.50000	0.50000	-1.50000	100.000000	0.000000	0.000000
51	0.50000	-1.50000	0.50000	0.000000	0.000000	-100.000000
52	-1.50000	0.50000	-0.50000	0.000000	100.000000	0.000000
53	-0.50000	1.50000	-1.50000	0.000000	0.000000	-100.000000
54	-1.50000	0.50000	1.50000	100.000000	0.000000	0.000000

55	0.50000	-0.50000	-1.50000	0.000000	0.000000	-100.000000
56	-1.50000	-1.50000	-0.50000	0.000000	100.000000	0.000000
57	-1.50000	1.50000	0.50000	100.000000	0.000000	0.000000
58	1.50000	-1.50000	-1.50000	0.000000	0.000000	100.000000
59	-1.50000	0.50000	0.50000	0.000000	-100.000000	0.000000
60	1.50000	-0.50000	-1.50000	0.000000	-100.000000	0.000000
61	-1.50000	-0.50000	1.50000	100.000000	0.000000	0.000000
62	0.50000	1.50000	-1.50000	0.000000	-100.000000	0.000000
63	-0.50000	0.50000	-0.50000	100.000000	0.000000	0.000000
64	-0.50000	-0.50000	0.50000	0.000000	0.000000	-100.000000