SANDIA REPORT

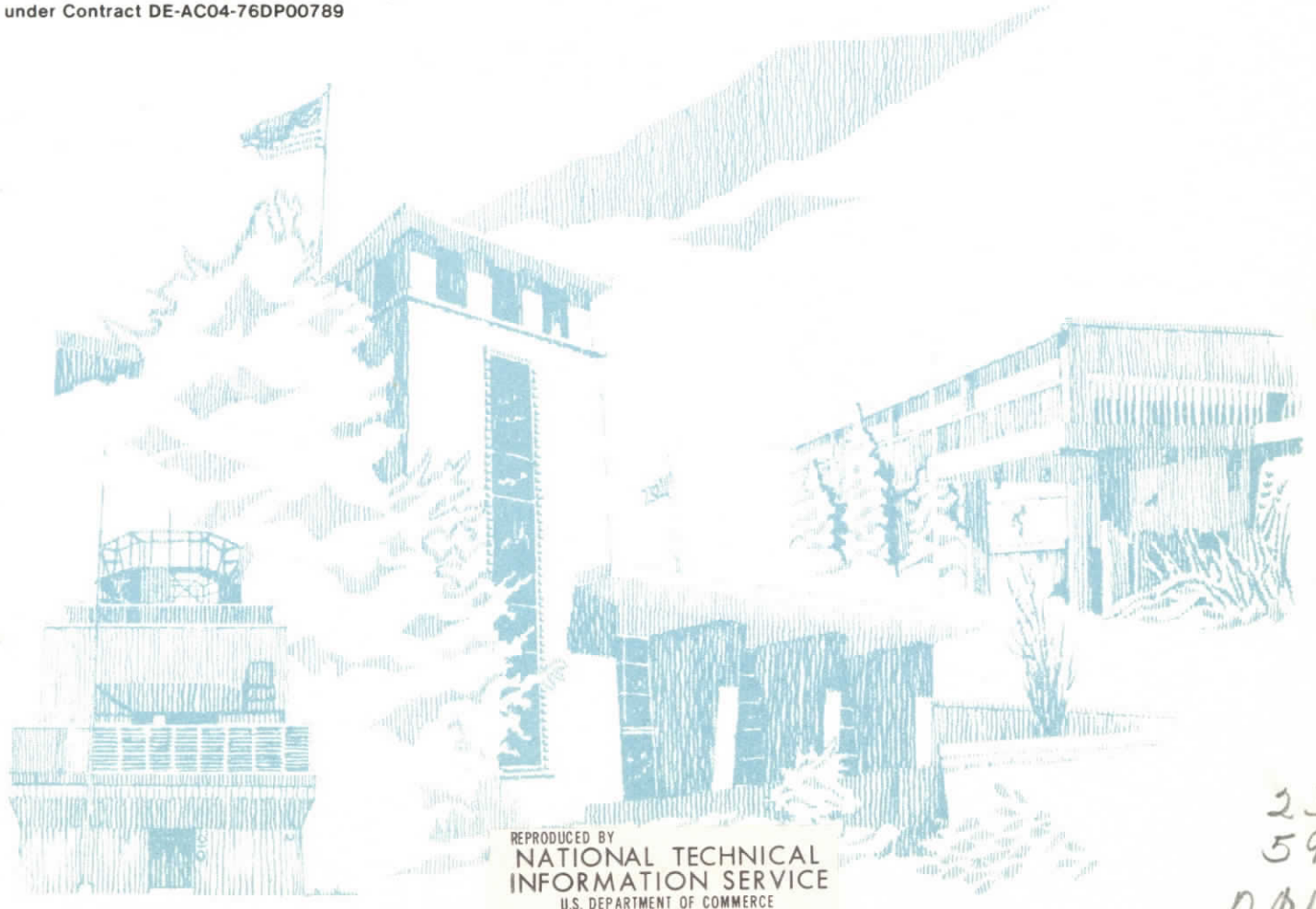
Printed March 1983

DE83009221
SAND-82-2373

BLOCKS, A Block Motion Code for Geomechanics Studies

Lee M. Taylor

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550
for the United States Department of Energy
under Contract DE-AC04-76DP00789



SF 2900-Q(6-82)

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

Printed in the United States of America
Available from
National Technical Information Service
U.S. Department of Commerce
5285 Port Royal Road
Springfield, VA 22161

NTIS price codes
Printed copy: A04
Microfiche copy: A01

SAND82-2373

Unlimited Distribution

BLOCKS, A Block Motion Code For Geomechanics Studies

Lee M. Taylor

Sandia National Laboratories
Albuquerque, New Mexico 87185

ABSTRACT

A block motion code, BLOCKS, has been developed for use in geological investigations. In this model originally proposed by Cundall in 1971, jointed rock is treated as an assemblage of blocks. The equations of motion are solved for each block using an explicit integration operator. The displacements and rotations of each block and the resulting collisions with nearby blocks are calculated in the model.

A complete theoretical description of the model is presented along with a detailed description of the computational algorithms used in the computer program. Several example problems are presented. A user's manual and complete documentation of the code are included.

TABLE OF CONTENTS

LIST OF FIGURES

1.0 INTRODUCTION

This report describes work at Sandia National Laboratories to develop a block motion code for use in geomechanics investigations. Specifically, the code has been developed for use in rubblization studies associated with the design of oil shale retorts and for use in subsidence prediction above coal mines. The model also has potential applications in a number of other geomechanics problems.

A block motion code tracks the motion of a collection of individual polygonal blocks and monitors the collisions between them. The blocks can be treated as rigid blocks or can be endowed with the ability to deform. Both capabilities are present in the model described here.

A block motion code models the rock mass as a collection or set of discontinuous blocks instead of as a continuous medium. Traditional continuum mechanics approaches for the calculation of the falling or gross motion of rock rubble or blocks have had a limited amount of success. Finite difference and finite element methods are generally unable to solve these problems in a tractable manner because of the highly jointed and layered nature of rock formations. Movements within the formation create slip planes and thereby allow large blocks of the formation to undergo rotations and translations. Thus, finite element methods require an inordinate number of slidelines to model the discontinuous nature of the problem. A novel approach by Benzley and Krieg [1] called the Rubble model has been used in the prediction of subsidence with some success.

The concept of a block motion model was first proposed by Cundall [2] in 1971. The model was intended for highly jointed rock where the deformations of the intact rock were negligible compared to the displacements which occur across the pre-existing joint systems. This model was further generalized by Cundall [3] and a code called RBM was developed in 1974. The code used explicit time integration and treated the blocks as rigid bodies. RBM was written in machine language for a

NOVA mini-computer. Cundall, et al [4] translated the code into FORTRAN and at the same time added several improvements to the model. The concept of simply deforming blocks was incorporated in this new code known as SDEM [4]. Also, a version of the code, called CRACK [4], was developed which allowed blocks to split into two separate blocks based on a tensile failure criteria. Walton [5] expanded upon the original RBM code and added a different contact force-displacement law.

The code described in this report is now known as BLOCKS. It had its origins in the SDEM code [4]. Approximately 40 percent of the original FORTRAN from SDEM still exists in BLOCKS. The program has seen extensive changes in the storage scheme and code architecture to facilitate vectorization on the CRAY. Also, a number of options have been added to the program. The mechanics of the model are basically the same as those found in SDEM. The descriptions of the algorithms and mechanics contained in the main body of this report are intended to document the code and some of what follows parallels descriptions that can be found in [4]. Where possible, the difference between SDEM and BLOCKS are pointed out and explained.

2.0 GOVERNING EQUATIONS

2.1 Centroidal Motions

The differenced equations of motion for the centroid of each block are given by

$$\ddot{X}^{t+\Delta t}_{nb} = F^{t}_{x_{nb}} / M_{nb} + g_x$$

$$\ddot{Y}^{t+\Delta t}_{nb} = F^{t}_{y_{nb}} / M_{nb} + g_y \qquad (1)$$

$$\ddot{\theta}^{t+\Delta t}_{nb} = T^{t}_{nb} / I_{nb}$$

where X, Y, and θ are the positions and rotation, respectively, of the block centroid. The subscript, nb, refers to a specific block number. Also, F_x, F_y, and T are the sum of the applied forces and moments on the block centroid, g_x and g_y are accelerations due to gravity, and M and I are the mass and rotatory moment of inertia for the block. Superposed dots represent derivatives with respect to time.

Equations (1) are integrated using an explicit forward difference operator to obtain centroidal velocities.

$$\dot{X}^{t+\Delta t}_{nb} = \dot{X}^{t}_{nb} + \Delta t\ \ddot{X}^{t+\Delta t}_{nb}$$

$$\dot{Y}^{t+\Delta t}_{nb} = \dot{Y}^{t}_{nb} + \Delta t\ \ddot{Y}^{t+\Delta t}_{nb} \qquad (2)$$

$$\dot{\theta}^{t+\Delta t}_{nb} = \dot{\theta}^{t}_{nb} + \Delta t\ \ddot{\theta}^{t+\Delta t}_{nb}$$

and centroidal positions

$$X_{nb}^{t+\Delta t} = X_{nb}^{t} + \Delta t\ \dot{X}_{nb}^{t+\Delta t}$$

$$Y_{nb}^{t+\Delta t} = Y_{nb}^{t} + \Delta t\ \dot{Y}_{nb}^{t+\Delta t} \qquad (3)$$

In equations (2) and (3), Δt is the time step.

2.2 Critical Time Step Size

All explicit integration schemes have a critical time step size which is determined by the highest eigenvalue in the system. For time steps larger than this critical value, the calculations become unstable. The program computes the critical time step size by finding the minimum value of

$$\Delta t_{cr} = 2\sqrt{M_{nb}/K} \qquad (4)$$

for all the blocks. In equation (4), K is either the spring constant for the normal direction of contact, K_n, or the spring constant for the shear direction, K_s; whichever is largest.

The critical time step given by equation (4) is based upon a single degree of freedom mass and spring system. For a densely packed system of blocks, the system is much stiffer than that implied by equation (4). Hence, the program requires that the user input a fraction of the critical time step. Experience has shown that, for densely packed assemblages of blocks, a value of, at most, 0.10 Δt_{cr} is acceptable.

At the present time, the critical time step size is based upon the contact law only and does not include any effects due to deformability of the blocks.

2.3 Simple Deformability

To account for the transfer of kinetic energy into strain energy, the concept of simple deformability has been introduced into the code.

An expression for the average applied stress on an arbitrary block in terms of the applied forces on the block can be derived as follows. The average applied stress on a block with volume, V, is defined by

$$\sigma_{ij}^{A} = \frac{1}{V} \int_{V} \sigma_{ij} \, dV \tag{5}$$

For the body shown in Figure (1), a virtual work equation can be written as

$$\int_{V} \sigma_{ij} \, \delta u_{i,j} \, dV = \sum_{n=1}^{nc} f_i^n \, \delta u_i^n \tag{6}$$

where f_i is the $n^{\underline{th}}$ applied force vector and δu_i^n is the vertical displacement vector at the point of application of that force. Here, we use the usual index notation with summation implied by repeated subscripts and differentiation indicated by commas. The virtual displacement field is arbitrary and is chosen as

$$\delta u_i = \phi_{ij} \, x_j + C_i \tag{7}$$

Where ϕ_{ij} is an arbitrary constant tensor, C_i is an arbitrary constant vector and x_j is the position vector of any point in the body with respect to the centroid of the body. Using equations (5) and (7) in equation (6) gives

$$\phi_{ij} \left[\sigma_{ij}^{A} \, V - \sum_{n=1}^{nc} f_i^n \, x_j^n \right] = 0 \tag{8}$$

or, since ϕ_{ij} is arbitrary

$$\sigma^A_{ij} = \frac{1}{V} \sum_{n=1}^{nc} f^n_i \, x^n_j \tag{9}$$

Cundall, et al [4] draw a direct analogy between a single degree of freedom spring-mass system and deformation response of a block to applied stress. They obtain an evolution equation for strain rates given by

$$\dot{\varepsilon}^{t+\Delta t}_{ij} = \dot{\varepsilon}^{t}_{ij} + \frac{\sigma^A_{ij} - \sigma^I_{ij}}{M^e(\ell_j)} \cdot \Delta t \tag{10}$$

Here, σ^I_{ij} is the internal stresses in the block while σ^A_{ij} are the average external applied stresses given by equation (9). The effective mass $M^e(\ell_j)$, is not the actual mass of the block, but is given by

$$M^e(\ell_j) = \frac{\ell_j \, M}{4A} \tag{11}$$

In equation (11), ℓ_j is projection of the block on the $j^{\underline{th}}$ coordinate direction and A is the area of the block. Hence, the effective mass is different for different directions. Cundall, et al present the derivation of equation (11) in [4]. Using this effective mass allows a stress wave to propagate through the assemblage of spring-masses at the same rate as through a continuum.

The applied stress is given by equation (9) and the strain rates by equation (10). The internal stress is calculated from the strain rates and a constitutive law

$$(\sigma^I_{ij})^{t+\Delta t} = (\sigma^I_{ij})^t + C_{ijkl} \, \dot{\varepsilon}^{t+\Delta t}_{ij} \cdot \Delta t \tag{12}$$

At the present time, only linear elasticity is included in the code.

2.4 Updating of Corner Positions

If the centroidal displacements, rotations and strain rates are known, the new positions of the corners can be calculated. The components of the vector which connect the $n^{\underline{th}}$ corner of the block to the centroid of the block are

$$\bar{X}_n = X_n - X_{nb} \qquad \bar{Y}_n = Y_n - Y_{nb} \tag{13}$$

In equation (13), the subscript nb refers to a block number and the subscript n refers to a specific corner in that block.

The displacements of the corner are

$$\begin{aligned} \Delta X_n &= \left[\dot{X}_{nb} + \dot{\varepsilon}^{11}_{nb}\ \bar{X} + (\dot{\varepsilon}^{12}_{nb} - \dot{\theta}_{nb}) \cdot \bar{Y}_n\right] \cdot \Delta t \\ \Delta Y_n &= \left[\dot{Y}_{nb} + \dot{\varepsilon}^{22}_{nb}\ \bar{Y} + (\dot{\varepsilon}^{21}_{nb} - \dot{\theta}_{nb}) \cdot \bar{X}_n\right] \cdot \Delta t \end{aligned} \tag{14}$$

and the new coordinates of the corner are

$$X_n^{t+\Delta t} = X_n^t + \Delta X_n \qquad Y_n^{t+\Delta t} = Y_n^t + \Delta Y_n \tag{15}$$

The integration used in equation (14) requires that $\dot{\theta}$ must be small compared to the time step size, Δt. This condition is satisfied for the types of problems considered here. The BLOCKS model is not intended for use in problems where the blocks have an extremely high angular velocity.

If the blocks are rigid (i.e., no deformability), the terms in equations (14) associated with strain rates are zero and can be removed.

2.5 Force-Displacement Law for Contacts

The BLOCK code retains the same corner to edge contacts found in the original SDEM [4] code. The contact law is written in a rate form. For the corner to edge contact shown in Figure 2, the relative velocity of the corner with respect to the edge is calculated and denoted by $\dot{X}_i$. This velocity vector has components $\dot{X}_n$ normal and $\dot{X}_s$ tangential to the edge. The normal and tangential (or shear) contact force can then be integrated as

$$F_n^{t+\Delta t} = F_n^{\,t} + \Delta t \quad \dot{X}_n \cdot K_n + \dot{X}_n \cdot C$$

$$F_s^{t+\Delta t} = F_s^{\,t} + \Delta t \quad \dot{X}_s \cdot K_s + \dot{X}_s \cdot C \qquad (16)$$

The spring constants, K_n and K_s, are input by the user. The damping coefficient, C, is defined in section 2.6. Coulomb friction is used in the model. The program checks to see if

$$F_s \leq \mu F_n \qquad (17)$$

If so, the forces are unchanged. If not, F_s is set to μF_n.

Writing the contact law in a rate form, as is done here, reduces the amount of data which must be stored for each contact. There is no need to keep track of the position of the corners with respect to where it first contacts the edge. The contact law is still just a spring-mass-dashpot system for each contact and a corner must penetrate an edge to develop a reaction force.

2.6 Damping Coefficient

The damping in equation (16) represents a viscous damper or dashpot in parallel with the contact spring. The original SDEM code used Rayleigh damping which is easily understood in the

context of an elastic, continuous system. Rayleigh damping uses a damping coefficient which is made up of a mass proportional part and a stiffness proportional part. The mass proportional part tends to damp out large global motions of the assemblage such as side to side motions. Stiffness proportional damping damps out the individual relative block motions which tend to cause the blocks to "rattle".

Rayleigh damping requires the user to specify which frequencies he wishes to damp. For a collection of blocks it can be quite difficult to determine which parameters to input for Rayleigh damping. The constrained, densley packed, assemblages of blocks for which the BLOCK code is intended, do not exhibit the global "sloshing" modes for which mass proportional damping is useful. But they do exhibit the tendency to "rattle". For these reasons, the specification of the amount of damping has been simplified.

For a collision of two blocks, with masses m_1 and m_2 respectively; separated by a spring and dashpot, the critical damping coefficient is given by

$$C_{cr} = 2\sqrt{\frac{m_1 m_2 K}{m_1 + m_2}} \tag{18}$$

where K is the spring constant in the normal direction. The code now requires the user to input the fraction of this critical damping. For every contact, a damping coefficient is calculated using equation (18) and the user supplied fraction.

The user now only has to supply one input value for which he may have some intuitive feel. Recall that with critical damping, there is no oscillation in the response; the two masses would stick together. For fractions of critical damping less than 1.0, the masses would rebound. A value of 0.5 has been used for most of the examples calculated so far. This represents a considerable amount of damping when compared to the damping found in normal structures and reflects the behavior of rock to rock contact. Rocks generally do not bounce.

3.0 Contact Search Algorithm

3.1 Boxing

For even a moderate number of blocks, the amount of time spent searching for corner to edge contacts becomes very large if the search proceeds on the basis that every corner could hit any given edge. To reduce this search time, the concept of boxing the corners is used in the code. Essentially, this process consists of defining a grid of Eulerian boxes on the domain in which the blocks reside. The blocks then fall through these boxes. Each corner is assigned to the box in which it currently resides. As the blocks move, a corner may cross a box boundary and be re-assigned to a new box. The search for potential contacts is then restricted to corners within the box (or boxes) in which the edge resides with a considerable reduction in search times. If a corner crosses the edge of the domain (no longer resides in a box), the block to which it belongs becomes fixed. This is not intended as a boundary condition, it just simplifies the coding. A boundary that a block can bounce off of can be defined using fixed blocks or using a symmetry plane.

3.2 Potential Contact List

The program maintains a list of potential contacts. A potential contact is a corner which has come "close enough" to an edge that it will probably hit the edge. A tolerance is input by the user which defines the distance. If a corner is within two times the tolerance of the edge, it is "close enough". Generally, a value for this tolerance which is slightly smaller than half the minimum length of a side suffices. Once a potential contact is identified and entered into the list, it is checked every time step to determine whether the corner has, in fact, penetrated the edge. Also, once in the list, the potential contact remains there no matter how far the corner and edge drift apart. The potential

contacts are all stored in a linked list in the global storage array as shown in Appendix D beginning at pointer M6. Obviously, for a large number of blocks, this list can become quite long.

The potential contact list is not updated every time step. If it were, the tolerance discussed above could be set to zero. Instead a fictitious displacement is calculated since the last update. This displacement is the sum of the maximum corner displacement for each time step. When this displacement exceeds the input tolerance, new potential contacts are searched for. Thus, when there is rapid motion of the blocks, the potential contact list is updated often. When there is relatively slow motion of the blocks, the list is updated infrequently. The use of a tolerance and the fictitious displacement trades storage space for computation time. With the tolerance, some potential contact records in the list represent corners that will never penetrate the edge with which they are paired. On the other hand, the relatively expensive and time consuming search for potential contacts is performed much less frequently. Note that a corner may be paired with several edges as entries in the list.

3.3 Contact Checks

Each potential contact is checked every time step to determine if the corner has penetrated the edge. There are several checks which must be made and unfortunately, this adds to the computational time of the analysis. These checks were required to correct some of the obvious deficiencies in the original SDEM code. In the original code, blocks sometimes had a tendency to "run over" each other. Also, for densely packed systems of blocks, there are special problems associated with some potential contacts which will be discussed below.

The first, and easiest, check is to determine if the corner has penetrated the edge. Figure 3 illustrates this case (greatly exaggerated). The area of the shaded triangle is given by

$$A = \frac{1}{2} (x^1 y^c + x^c y^2 + x^2 y^1 - y^1 x^c - y^c x^2 - y^2 x^1) \qquad (19)$$

If the area given by equation (19) is negative, the corner has not penetrated the edge and there is no contact. If the area is positive, the code then checks to determine whether the corner is between the two corners of the edge. If not, there is no contact.

The last check is associated with the dense packing of blocks. Figure 4a illustrated the usual sort of contact such as would be detected by the checks described above. With dense packing, it is common to have two blocks initially positioned as shown in Figure 4b. When the code performs an update of the potential contact list, it will create a potential contact between corner 3 of block 1 and edge 3 of block 2 (this edge contains corners 2 and 3). Now, this is a spurious contact and arises only because of the tolerance used to detect when a corner is "close enough" to an edge (see Section 3.2). To see the difficulties associated with this spurious contact, invision a motion in which block 2 is fixed and block 1 falls under gravity. Due to roundoff, the first check described above may very well calculate a positive area and the second check may find that corner 3 of block 1 is just to the right of corner 2 of block 2. As a result, the corner will "hang up" and the block will not slide down the first side of block 2 as it should.

To detect this form of spurious contact, the following procedure is performed. The outward normal to the edge, $\hat{n}$, and unit vectors along the two edges, $\hat{t}_1$, and $\hat{t}_2$, are formed (see Figure 4c). If both $\hat{t}_1 \cdot \hat{n} \geq 0$ and $\hat{t}_2 \cdot \hat{n} > 0$, then

this is not a spurious contact. Actually, you can not check on zero since a case such as shown in Figure 4d may arise. In this case, because the corners may penetrate small but unequal amounts, some tolerance must be included. The code uses a tolerance associated with an angle of -10°. Hence,

$$\hat{t}_1 \cdot \hat{n} \geq \sin(-10^\circ) \qquad (20)$$
$$\hat{t}_2 \cdot \hat{n} \geq \sin(-10^\circ)$$

indicates that this is in fact a true contact.

4.0 Example Problems

4.1 Crater Formation

The first example problem represents the explosive formation of a crater in oil shale. This is a basic calculation needed for rubblization studies associated with an oil shale retort. Figure 5 shows the geometry of the model and the original blocking of the rock mass. The shaded area represents the borehole in which the explosive is detonated. The explosive loading is represented by an initial impulse and a force-time load applied to the blocks nearest the borehole. The impulse is converted to an initial block velocity.

Figure 6 shows the resulting crater formation after 500 msec. Only half the model was analyzed due to symmetry.

4.2 Subsidence Prediction

The increase in the demand for coal will eventually lead to mining of coal in more populated regions of the United States and a resultant need for numerical methods for predicting subsidence above coal mines. Schuler and Sutherland present a review of current subsidence predictive methods in Ref. 6. In their report, they present a scale model test for an idealized mine structure. Figure 7a shows a BLOCK model for the idealized mine structure (see also Ref. 7). Each layer has a random distribution of blocks, with 850 blocks in the model. The RIGID option was used, which results in approximately a 10 percent decrease in computer time. Only half the model shown was analyzed due to symmetry. Figure 7b shows the final block orientations. The model correctly predicts the non-critical subsidence and predicts the width of the subsidence trough quite well. It overpredicts the subsidence depth by 17 percent.

References

1. S. E. Benzley and R. D. Krieg, "A Continuum Finite Element Approach for Rock Failure and Rubble Formation," International Journal for Numerical and Analytical Methods in Geomechanics, Vol 6, p 277-286, 1982.

2. P. A. Cundall, "A Computer Model for Simulating Progressive Large Scale Movements in Blocking Rock Systems," Proceedings of Symposium on Rock Fracture (ISRM), paper II-8, 1971.

3. P. A. Cundall, "Rational Design of Tunnel Supports: A Computer Model for Rock Mass Behavior Using Interactive Graphics for the Input and Output of Geometrical Data," U.S. Army Corps of Engineers Technical Report, MRD-2-74, 1974.

4. P. A. Cundall, J. Marti, P. Beresford, N. Last, M. Asgain, and T. Maini, "Computer Modeling of Jointed Rock Masses," Dames and Moore, Los Angeles, Technical Report N-78-4, 1978.

5. O. R. Walton, Particle Dynamics Modeling of Geological Materials, PhD Thesis, University of California, Davis, 1980.

6. H. J. Sutherland and K. W. Schuler, "A Review of Subsidence Prediction Research Conducted at Sandia National Laboratories," SAND82-0017, Albuquerque, New Mexico, 1982.

References (Con't)

7. H. S. Sutherland, K. W. Schuler, and S. E. Benzley, "Numerical and Physical Simulations of Strata Movements Above Idealized Mine Structures, In Situ (in publication).

8. Z. E. Beisinger and C. M. Stone, "TPLO2: A Flexible X-Y Plotting Program for Use with Finite Element Software," SAND80-2508, Sandia National Laboratories, Albuquerque, New Mexico, 1980.

The input data for BLOCKS is read in two parts. There is a certain amount of data which must be read at the beginning of the run and which must remain constant. This is called the preliminary data and is described in section A.1. Following the preliminary data is the problem definition data described in section A.2. These cards may appear in whatever order is appropriate for the particular problem. Unlike the preliminary data, none of these cards is required. Many of these cards require additional cards (i.e., CREATE, Section A.2.1).

The amount of output created by the code is controlled by data cards in the problem definition set. See sections A.2.17 and A.2.18. The user has control over the frequency of both the printed output and the output written on the plot tape.

The following convention is used throughout this users manual. Words in upper case letters represent keywords and must appear on the data card. Words in lower case letters represent data for which values must be entered. Free field input is used throughout with data separated by commas.

A.1.0 Preliminary Data

Almost all the following cards are required at the beginning of a new run (the exceptions are STIFF, RIGID, and RESTART). The cards may be read in any order, except that a START card and its associated title card must be first and an END card must be last.

In a restart, the only preliminary data card allowed is the RESTART card (an END card is not required).

A.1.1 START, nblokm, ncmax

nblokm = maximum number of blocks

ncmax = maximum number of corners per block

title card (separate card, 80 characters allowed)

A.1.2 XLIMITS, xmin, xmax

xmin = minimum x coordinate of outer box boundaries

xmax = maximum x coordinate of outer box boundaries

A.1.3 YLIMITS, ymin, ymax

ymin = minimum y coordinate of outer box boundaries

ymax = maximum y coordinate of outer box boundaries

A.1.4 BOXES, iboxes, jboxes, ctol

iboxes = number of boxes in the x direction

jboxes = number of boxes in the y direction

ctol = contact tolerance (default = .1)

The size of the boxes is determined by:

bsizex = (xmax-xmin)/iboxes

bsizey = (ymax-ymin)/jboxes

A.1.5 STIFF, stifn, stifs

stifn = contact siffness in the normal direction (default = 10^6)

stifs = contact stiffness in the tangential direction (default = 10^6)

A.1.6 FRACTION, tfrac

tfrac = fraction of the critical time step.

A.1.7 MATERIAL, mat, E, G, ρ

mat = material number

E = Young's modulus

G = Shear modulus

ρ = mass density

More than one of these cards may be used.

A.1.8 FUNCTION, ntimep

t_1, p_1

t_2, p_2

. .
. .
. .

t_{ntimep}, p_{ntimep}

ntimep = number of time function data cards to read.

t_i, p_i = function (time history) data point.

A.1.9 RIGID

Indicates that blocks are nondeforming.

A.1.10 END

Indicates the end of preliminary data. (Not used for a RESTART)

A.1.11 RESTART

Indicates this is a restart of a previous problem. No other preliminary data is required.

A.2.0 Problem Definition Data

The data cards described below are used to define the problem. The cards may appear in whatever order is appropriate for the particular problem. Unlike the preliminary data, none of these cards is required. Many of these cards require additional cards (i.e., CREATE).

A.2.1 CREATE, nc, mat, ifix

nc = number of corners
mat = material number (see SEc. 1.7)
ifix = flag which allows block to be fixed.
if ifix ≠ 0 block is fixed (default = 0)

Following the CREATE card must be two cards containing the x and y coordinates of the corners of the block (in clockwise order):

$x_1, x_2, \ldots, x_{nc}$

$y_1, y_2, \ldots, y_{nc}$

A.2.2 CLOOP, nloop, radd, zadd

This command is for use in conjunction with the CREATE command (A.2.1). It specifies that the set of CREATE cards which must follow next will be repeated nloop times and the coordinates of the block are shifted radd and zadd each time. This allows the user to create a number of similar blocks quite easily.

A.2.3 READ9

This card tells the program to read tape9 which contains blocks as defined by QMESH.

A.2.4 | LAYER, ± ns, mat, x_1m y_1, x_2, y_2, x_3, y_3, x_4, y_4, rcommand |

This card is used to define a layer of blocks. There are several ways to use this option. You can generate evenly spaced blocks, randomly spaced blocks, or define the spacing with additional cards.

ns = number of blocks along the layer.

If ns > 0, the spacing is read as described below.
If ns < 0, the spacing is either uniform or random depending on the rcommand.

x_i, y_i, i = 1,4 are the corners of the layer as shown in Figure A.1

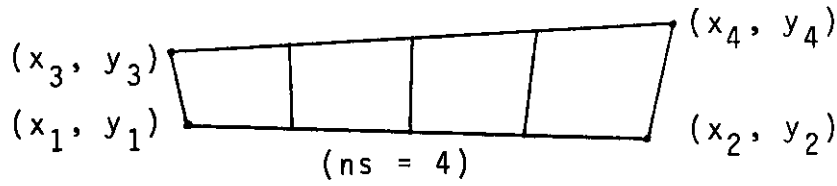


Figure A.1

rcommand = command to initiate random spacing. If blank even spacing is used. If rcommand = RANDOM, the spacing is random. Currently, blocks are limited from having an aspect ratio of less than 0.2 or greater than 3.0.

The blocks are first spaced along the middle of the layer randomly. Then the angular orientation between ANGPLUS and ANGMIN (see A.2.5) is set randomly. If ANGPLUS and ANGMIN are zero (default), the sides of the blocks are normal to the midsurface of the layer.

When ns > 0 on the LAYER card, two additional cards must be read, first, the spacing on the bottom line and second, the spacing on the top line.

sb_1, sb_2, sb_3, ..., sb_{ns}

st_1, st_2, st_3, ..., st_{ns}

If ns > 2 and sb_3 to sb_{ns} are omitted (likewise for st_3 to st_{ns}), the spacing is given by sb_1, sb_2, sb_2, sb_2, ...

A.2.5 | ORIENT, angmin, angplus |

Sets the orientation of the block sides for the RANDOM option on the LAYER card.

angmin = minimum angle (defalut = 0.0)
angplus = maximum angle (default 0.0)

NOTE: both the same gives all block sides the same orientation.

A.2.6 | MERGE, aratio |

aratio = minimum aspect ratio allowed for blocks created with the RANDOM option on the LAYER card.

This card causes small, slender blocks to be absorbed into the next block. Applies only to the RANDOM option of the LAYER card. Note: this will usually reduce the number of blocks.

A.2.7 **SPLANE, X_1, Y_1, X_2, Y_2**

Defines a symmetry plane.

X_1, Y_1 = coordinates of $1^{\underline{st}}$ point defining the plane.

X_2, Y_2 = coordinates of $2^{\underline{2nd}}$ point defining the plane.

NOTE: The orientation of the plane is defined by the order of the two points. The normal, $\hat{n}$, to the plane must point towards the blocks being restrained by the symmetry plane as shown in Figure A.2.

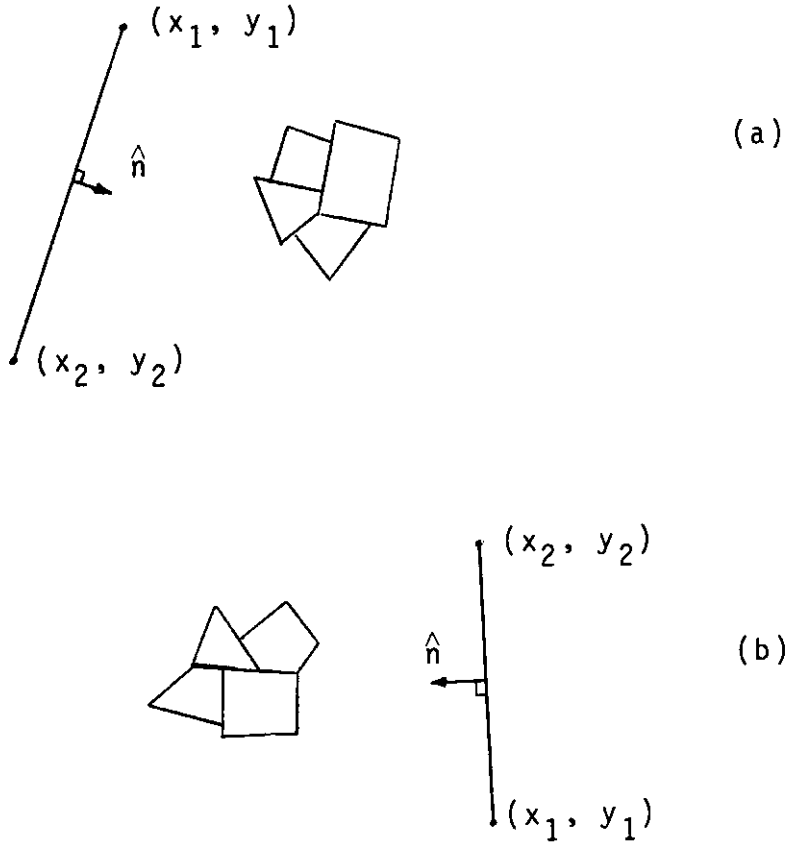


Figure A.2

A.2.8 GRAVITY, gravx, gravy

gravx = acceleration due to gravity in the x direction.

gravy = acceleration due to gravity in the y direction.

A.2.9 FIX, nblok

nblok = number of block to be fixed. This can also be accomplished with ifix on a CREATE card.

A.2.10 PRESSBC, nblok, nside, p_1, p_2, t_{arriv}

nblok = block number

nside = side number (counting clockwise)

p_1 = pressure multiplier on first node on side, multiplies function defined in preliminary data (see Section 1.8)

p_2 = pressure multiplier on second node on side.

t_{arrive} = pressure arrival time

A.2.11 FORCE, nblok, fx, fy, t_{arrive}

nblok = block number

fx = multiplier of x component of applied force on block centroid, multiplies function defined in preliminary data (see section 1.8)

fy = multiplier of y component of applied force at block centroid, multiplies function defined in preliminary data

t_{arrive} = force arrival time

A.2.12 FRICTN, fric

fric = coefficient of friction between blocks, (default = 0.0)

A.2.13 SETVEL, nb_1, nb_2, vx, vy, $\dot{\theta}$

nb_1 = block number

nb_2 = block number

If nb1 < nb_2, then all blocks numbered nb_1 to nb_2 get these initial velocities

If nb_2 = 0, just block nb_1 gets these initial velocities.

If nb_2 < 0, blocks nb_1 to nblocks get these initial velocities.

vx = velocity in x direction

vy = velocity in y direction

$\dot{\theta}$ = angular velocity

A.2.14 SETVELM, mat, vx, vy, $\dot{\theta}$

mat = material number (see Sec. 1.7)

vx, vy, $\dot{\theta}$ = same as Sec. 2.13

All blocks with material number mat get these initial velocities.

This card is handy for defining initial conditions on blocks that were read from a QMESH tape (see Sec. 2.2) where the order or numbering of the blocks is unknown.

A.2.15 DAMPING, β

Defines the amount of damping in the problem.

β = fraction of critical damping.

A.2.16 DUMP

Prints core by blocks, boxes and contacts. Useful for debugging, but generates a considerable amount of output if there are very many blocks.

A.2.17 OUTPUT, dtout, nb, nb_2, ...nb_{16}

dtout = time interval at which output is to be printed

nb_i = block numbers to be printed, if absent, all blocks are printed ($i \leq 16$)

A.2.18 PLOTS, dtplot

dtplot = time interval at which plot tape is to be written.

A.2.19 MOTION, dt

dt = time interval for calculating the blocks motion

This card causes the code to run for a given interval of time.

A.2.20 CYCLE, ncycle

ncycle = number of time steps to be run.

This card causes the code to run for a given number of time steps.

A.2.21 STOP

Stops program execution. Restart tape is written.

Appendix B — Subroutine Descriptions

BLOCKS	Main program. Contains all the data statements which define default values for variables. Zeros the core storage and material property arrays.
BOX	This subroutine determines in which box each corner resides. It also creates am empty list of potential contacts from the end of the boxing data to the end of the core storage. The critical time step is calculated here. Called from NEXT.
CREATE	Calculates the properties for a block and stores them in the global storage array. Called by NEXT, QMESH, and LAYER.
CYCLE	This subroutine is the driver for the program. It contains the time step loop. Called by NEXT.
DUMP	An output routine which dumps the contents of the global storage array. This subroutine is used primarily for debug purposes and generates a considerable amount of output for larger problems. Called by NEXT.
FFLD	Free field reader. Called by SETUP, NEXT, and LAYER.

FINISH	This subroutine writes the restart tape. For a normal termination it is called by NEXT. Some errors result in it being called from other portions of the code.
FORD	This subroutine calculates the contact forces between blocks. It contains several checks to insure that blocks do not run over one another. Called by CYCLE.
LAYER	An input subroutine which defines layers of blocks. Called by NEXT.
LISTI	Prints a list of the input data at the beginning of execution. Called by BLOCKS.
MOTION	Calculates the strain rates for deforming blocks. The subroutine also updates the position of the block corners. Called by CYCLE.
NEXT	An input subroutine which defines the problem. Called by BLOCKS.
OUT	An output subroutine which prints the position, velocities, stresses, etc., for selected blocks at regular time intervals. Called by CYCLE.
PRESS	This subroutine calculates the centroidal forces and moments on a block due to pressure applied to a side. Called by CYCLE.

QMESH	A subroutine to read block definitions from a QMESH run off TAPE9. Called by NEXT.
REBOX	This subroutine reboxes a corner which has crossed over a box boundary. Called by MOTION.
SETUP	A subroutine to read the preliminary data. Called by BLOCKS.
STRESS	This subroutine contains the constitutive law which relates strain rates to stress rates. Currently it has only linear elasticity. Called by CYCLE.
UPDAT	This subroutine updates the list of potential contacts between corners and edges. Called by CYCLE.
WPLTP	A subroutine which writes the plot tape for subsequent post-processing. Called by CYCLE.

Appendix C Variable Definitions

There are four common blocks in the program which contain the variables which define the problem. The contents of these common blocks are what is dumped onto the restart tape (see Appendix G).

Blank common contains the global storage array. The dimension of the array must be increased to do larger problems. Note: the user must change the value of M7 in the DATA statement in MAIN to reflect the correct length of the global storage array. The three other common blocks and the definition of the variables they contain are:

```
COMMON/CINTGR/NBLOKM,NCMAX,NBOXES,M3A,M3B,M3C,M3D,M3E,M3,M4
    M5,M6,M7,NBLOKS,NCTOT,NCYC,MCYCLE,NEMPT,IBOXES,JBOXES,
    NBR,NPR,NUPDAT,IBR,MAXMAT,IBCFLG,NLIST,ILIST(16),NPLOT
    M1A,M1B,M1C,M1D,M1E,M1F,M1G,M1H,NBCP,NTIMEP,IUFLG,IUDKNT,
    IRIGID,ISEED
```

IBCFLG	Flag which shows whether pressure-time history data is defined.
IBOXES	Number of boxes in x direction.
IBR	Not used.
ILIST(16)	List of blocks to have printed output.
IRIGID	Flag which indicates whether simple deforming capability is on or off.
ISEED	Seed for random number generator.
IUDKNT	Counts the number of time steps between calls to UPDAT.

IUFLG	Flag which tells the code to call UPDAT.
JBOXES	Number of boxes in the y direction.
MAXMAT	Maximum number of material types allowed.
MCYCLE	Total number of time steps performed.
NBCP	Number of pressure boundary conditions applied.
NBLOKM	Maximum number of blocks allowed.
NBLOKS	Number of blocks currently defined.
NBOXES	Number of boxes, equal to IBOXES*JBOXES.
NBR	Not used.
NCMAX	Maximum number of corners allowed per block.
NCTOT	Total number of corners, equal to NCMAX*NBLOCKS. Note that some blocks may actually have less than NCMAX corners.
NCYC	Number of cycles to be run as defined on the CYCLE data card (see A.2.20).
NEMPT	Pointer to the next available contact record for a new contact in the global storage array.
NLIST	Number of blocks defined in the ILIST array.

NPLOT	Flag which tells the code that the initial plot data has already been written.
NPR	Not used.
NTIMEP	Not used.
NUPDAT	Total number of calls to UPDAT made.

M1A,M1B,....M3E,M4,M5,M6,M7 are pointers to data in the global storage array (see Appendix D).

```
COMMON/CREAL/HED(10),TFRAC,TDEL,BSIZEX,BSIZEY,UDMAX,TIME,CTOL,
  UMOST,STIFN,STIFS,FRIC,BETA,GRAVX,GRAVY,XL,YL,XU,YU,
  DTIME,DTOUT,DTPLOT,TEND,TOUT,TPLOT,PT(50,2)ANGM,ANGP
```

ANGM	Minimum angle for random orientation.
ANGP	Maximum angle for random orientation.
BETA	Fraction of critical damping
BSIZEX	Box size in x direction.
BSIZEY	Box size in y direction.
CTOL	Contact tolerance.
DTIME	Increment of time to be run.
DTOUT	Increment of time between printed output.
DTPLOT	Increment of time between writing on the plot tape.

FRIC	Coefficient of friction.
GRAVX	Gravity in the x direction.
GRAVY	Gravity in the y direction.
HED(10)	Heading for the problem.
PT(50,2)	Pressure-time history.
STIFN	Normal contact stiffness.
STIFS	Shear contact stiffness.
TDEL	Time step size.
TFRAC	Fraction of the critical time step.
TEND	Time at which analysis ends.
TIME	Current time, equals MCYCLE*TDEL
TOUT	Time for next printed output.
TPLOT	Time for next write to plot tape.
UDMAX	Maximum displacement of any corner during time step.
UMOST	Accumulation of UDMAXs since last call to UPDAT.
XL	Minimum x coordinate of problem.
XU	Maximum x coordinate of problem.

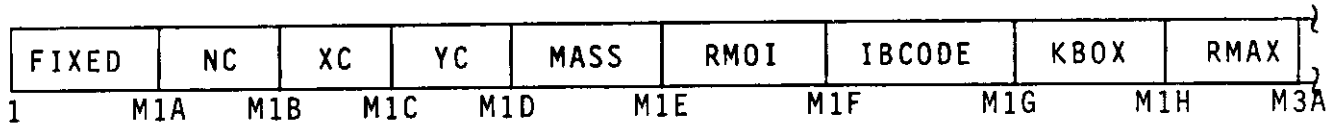
YL	Minimum y coordinate of problem.
YU	Maximum y coordinate of problem.

COMMON/MATPRO/PROP(10,8)

PROP	Material property data

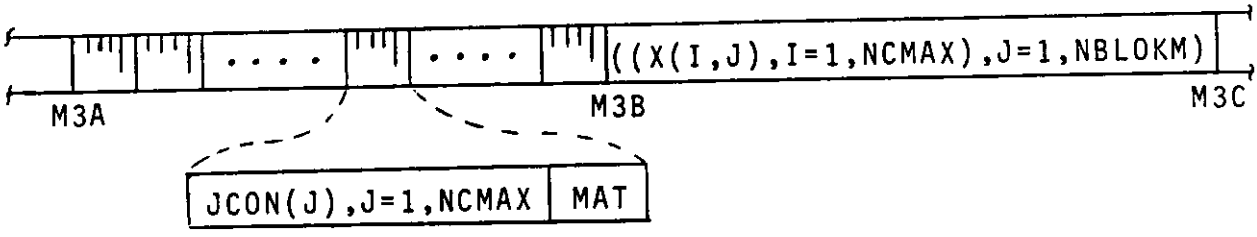
Appendix D Storage

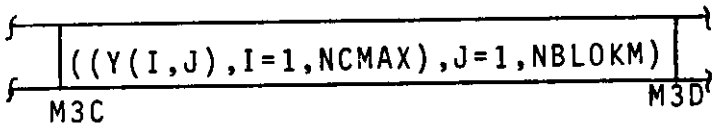
The program uses a global storage array to store the data for the problem. The location of the data is described below.



FIXED = fixed/free flag, 0.0 for fixed blocks, 1.0 for free blocks.
NC = number of corners in block
XC = x coordinate of block centroid
YC = y coordinate of block centroid
MASS = mass of block
RMOI = moment of inertia of block
IBCODE = (not used)
KBOX = contains information on boxing of corners, dimensioned ncmax by nblockm, each entry is 1000*IBOX + JBOX.
RMAX = (not used)

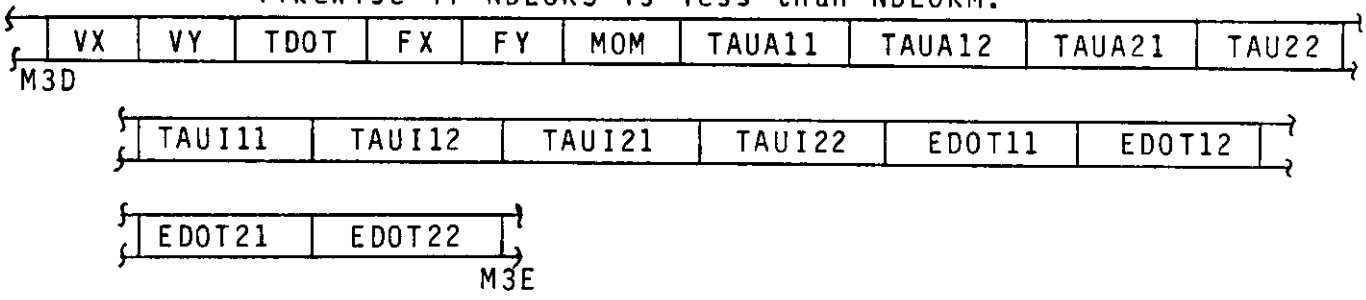
Each of the variables described above is a vector of length nblockm (except KBOX, which is a two-dimensional array).





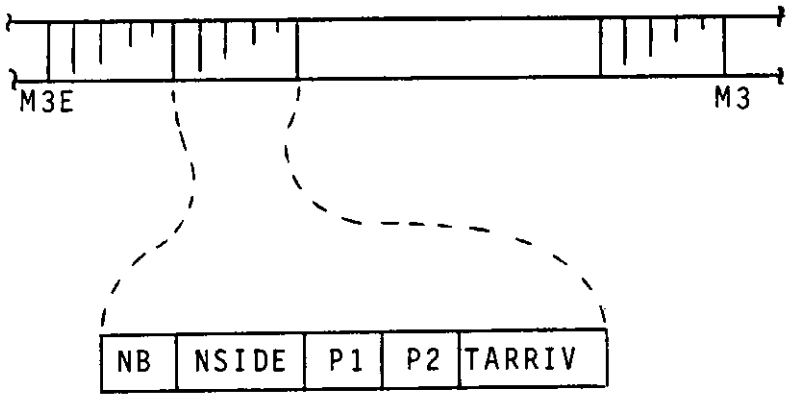
JCON = connectivity for each block
MAT = material number for each block
x = x coordinates of corners
y = y coordinates of corners

NOTE: If NC is less than the maximum number of corners, NCMAX, then the rest of the record is just zero, likewise if NBLOKS is less than NBLOKM.



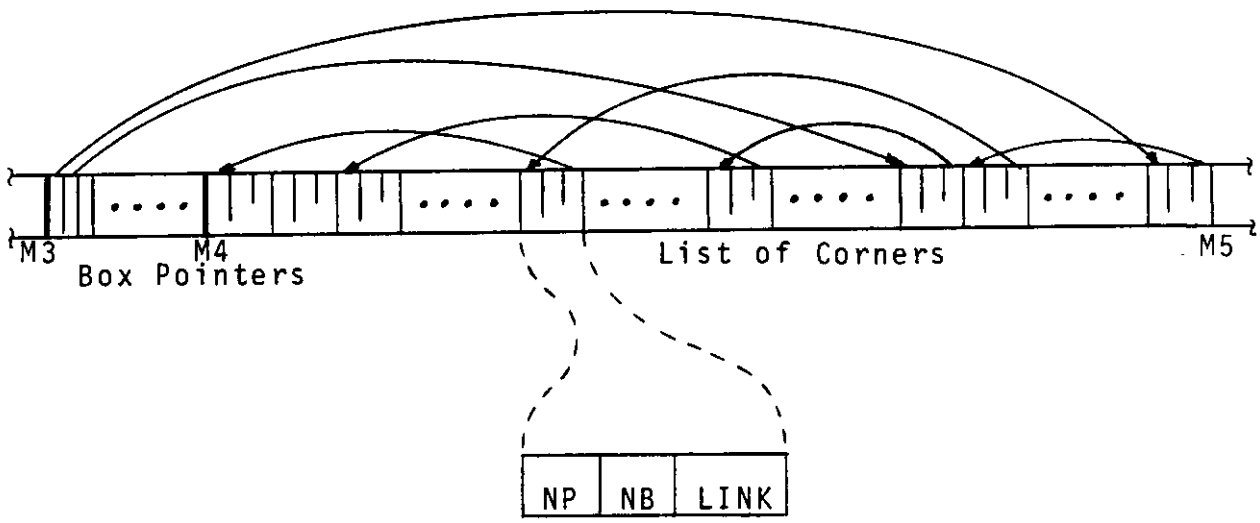
VX = velocity of block in x direction
VY = velocity of block in y direction
TDOT = angular velocity of block
FX = force on block in x direction
FY = force on block in y direction
MOM = moment on block
TAUA11, TAUA12, TAUA21, TAUA22 = applied stress on block
TAUI11, TAUI12, TAUI21, TAUI22 = internal stresses in block
EDOT11, EDOT12, EDOT21, EDOT22 = strain rates in block

NOTE: All these vectors are of length, NBLOKM.

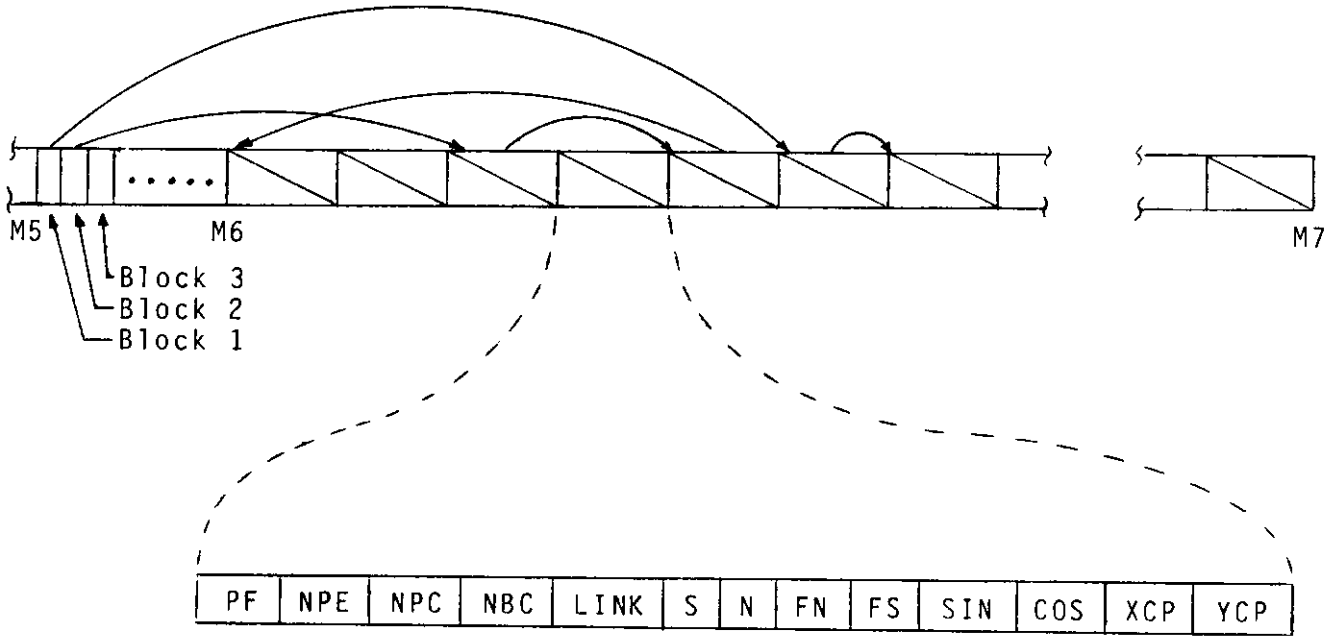


5 words per boundary condition, if no pressure boundary conditions are present, M3E = M3.

NB = block on which pressure is applied
NSIDE = side number on block
P1 = pressure multiplier on first node on side
P2 = pressure multiplier on last node on side
TARRIV = pressure arrival time



NP corner number
NB block number
LINK address of next corner



PF	Not used
NPE	Edge number of this block
NPC	Corner number of block contacting this block
NBC	Block number of block contacting this block
LINK	Address number of next contact data for this block
S	Shear displacement
N	Normal displacement
FN	Normal force
FS	Shear force
SIN	Sine of edge NPE
COS	Cosine of edge NPE
XCP	X coordinate of contact point
YCP	Y coordinate of contact point

Appendix E User's Manual for DBLOCK

DBLOCK is a plotting program that generates plots of the configuration of the blocks at various times.

All input cards for DBLOCK have the following format:

```
CODE WORD, NUM1, NUM2, .... NUM7
```

The code words read by DBLOCK are listed in Table E.1. These cards may be input in any order provided that the ENDDATA card be placed last.

The option to produce color plots is available in DBLOCK. This option allows the user to color code the different materials in the geometrical region. The color capability is initiated simply by including data cards that define the colors of the separate materials in the problem. These cards are listed in Table E.2.

Code Word	NUM	Description
BLOCKS	--	Plot original configuration of blocks only.
ZOOM	NUM1 - R_{min} NUM2 - R_{max} NUM3 - Z_{min} NUM4 - Z_{max}	Plot only the R-Z indicated area.
ENDDATA		Ends input string.
FRAMESIZE	NUM1	NUM1 = 1 Plot as large as possible (default) NUM1 = 2 Medium size plot for reports NUM1 = 3 Movie size plot (default with 'MOVIE')
PRINT	--	Print nodal coordinates in IX array.
NTIMES	NUM1	NUM1 evenly spaced plots from TMIN to TMAX. Default NUM1 = 10.
ALLTIMES	--	Plot all times from TMIN to TMAX.
TMIN	NUM1 = T_{min}	Minimum time at which plots are requested. Default = mimimum time on Tape 10.
TMAX	NUM1 = t_{max}	Maximum time at which plots are requested. Default = maximum time on Tape 10.
DELTIME	NUM1 = Δt	Plots at every Δt increment between t_{min} and t_{max}.
TIMES	NUM1,NUM2,....	Array of times at which plots are requested. Any number of TIMES CARDS may be present.
MOVIE		Sets plot size to keep frames from overlapping.
FLIP		Flips the plot about the z axis. Note that if the zoom option is being used, the bounds must include the flipped blocks.

TABLE E.1

Code Word	NUM	Description
CLEARC	NUM	Material number NUM will be drawn in white.
BLUE	NUM	Material number NUM will be drawn in blue.
CYAN	NUM	Material number NUM will be drawn in cyan, a light blue green.
GREEN	NUM	Material number NUM will be drawn in green.
MGENTA	NUM	Material number NUM will be drawn in magenta, a reddish blue.
RED	NUM	Material number NUM will be drawn in red.
YELLOW	NUM	Material number NUM will be drawn in yellow.
ZUNI	NUM	Material number NUM will be drawn in zuni, Sandia light blue.

TABLE E.2 - COLORS AND CODE WORDS AVAILABLE IN DBLOCK

Appendix F Using TPLOT2

TPLOT2 is a post-processing program for generating time history plots. The program is used primarily for plotting the results of finite element analysis. The post-processing tape written by BLOCKS must be translated before TPLOT2 can be run. The program which performs this task is called NEWTEN.

A user's guide for TPLOT2 by Beisinger and Stone [8] is available. The list of ALPHA names generated by NEWTEN is given in TABLE F.1.

ALPHA NAMES FROM DATA TAPES

	ELEMENT	NODAL	GLOBAL
1	VELX	X	
2	VELY	Y	
3	TOUT		
4	FORCEX		
5	FORCEY		
6	MOMENT		
7	SIGXX		
8	SIGXY		
9	SIGYX		
19	SIGYY		

TABLE F.1

Appendix G Tape Files

G.1 Post-Processing tape

The post processing tape for use with the DBLOCK and TPLOT2 plotting codes is written on unit 10. The initial data, written only once is:

```
NN = NBLOCKS * (NCMAX+1)
WRITE(10)  (HED(I),I=1,10) NBLOKM,NPAX,NBLOCKS,NCTOT,NCMAX
WRITE(10  (X(I),I=1,NCTOT), (Y(I),I=1,NCTOT), (JCON(I),I=1,NN)
```

The time dependent data, written at the interval of time defined on the PLOTS card is:

```
WRITE(10)  TIME,NBLOKS,NCTOT,NCMAX
WRITE(10)  (X(I),I=1,NCTOT), (Y(I), I=1,NCTOT),(JCON(I),I=1,NN)
WRITE(10)  (VX(I),I=1,NBLOKS),(VY(I), I=1,NBLOKS),(TDOT(I),
           I=1,NBLOKS),(FX(I),I=1,NBLOKS),(FY(I),I=1,NBLOKS),
           (TMOM(I),I=1,NBLOKS)

WRITE(10)  ((SIGI(I,J),J=1,NBLOKS),I=1,4)
```

G.2 Restart Tape

The restart tape is written on unit 12 as:

```
WRITE(12)
NBLOKM,NCMAX,NBOXES,M3A,M3B,M3C,M3E,M3,M4
M5,M6,M7,NBLOKS,NCTOT,NCYC,MCYCLE,NEMPT,IBOXES,JBOXES,
NBR,NPR,NUPDAT,IBR,MAXMAT,IBCFLG,NLIST,(ILIST(I),I=1,16),
NPLOT,MIA,MIB,MIC,MID,MIE,MIF,MIG,MIH,NBCP,NTIMEP,IUFLG,
IUDKNT,IRIGID
WRITE(12)
(HED(I),I=1,10),TFRAC,TDEL,BSIZEX,BSIXEY,UDMAX,TIME,CTOL,
UMOST,STIFN,STIFS,FRIC,BETA,GRAVX,GRAVY,XL,YL,XU,YU,
DTIME,DTOUT,DTPLOT,TEND,TOUT,TPLOT,((PT(I,J),I=1,50),J=1,2)
ANGM,ANGP
WRITE(12) ((PROP(I,J),J=1,8),I=1,10)
WRITE(12) (A(I),I=1,NEMPT)
```

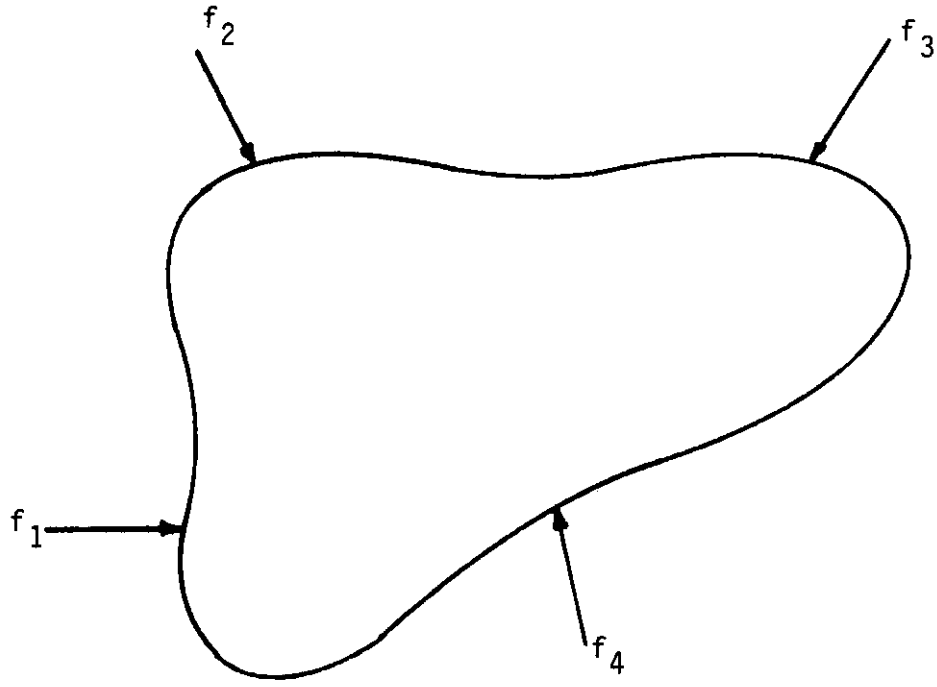


Figure 1. Applied Forces on a Body

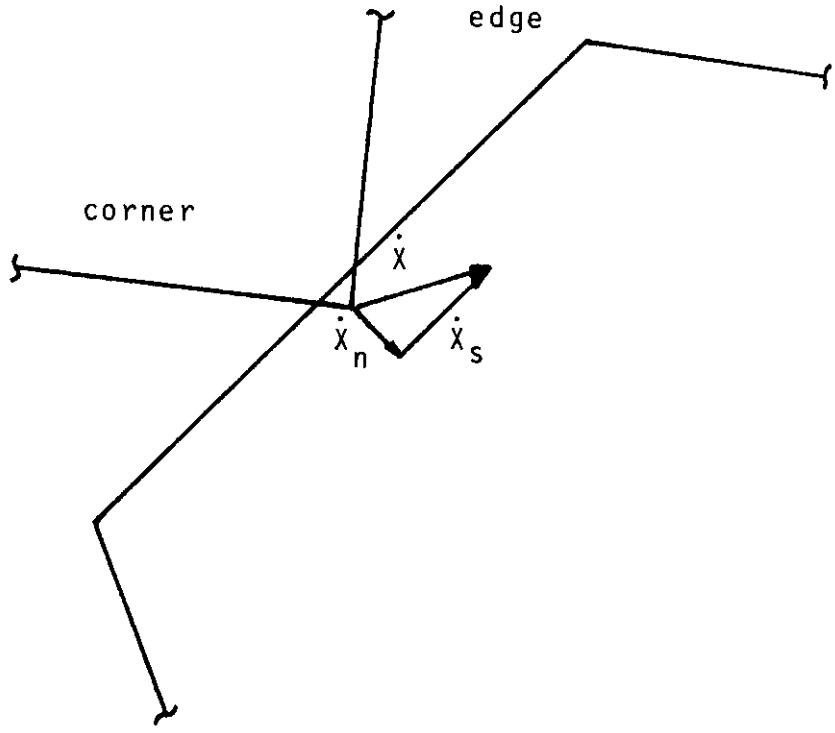


Figure 2. Corner to Edge Contact

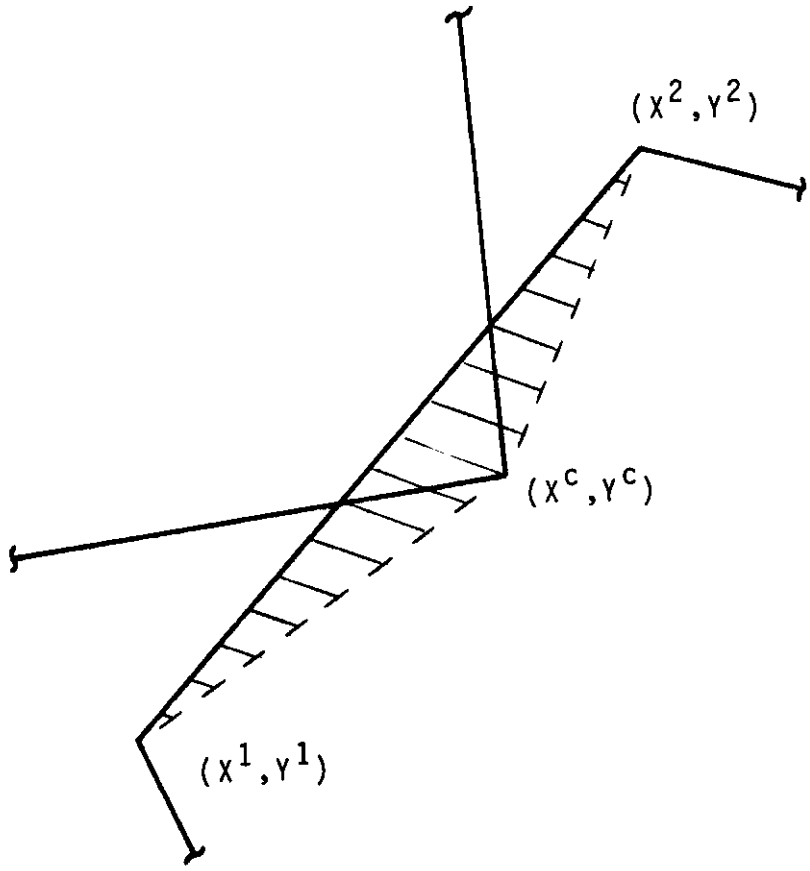


Figure 3. Penetration of an Edge by a Corner

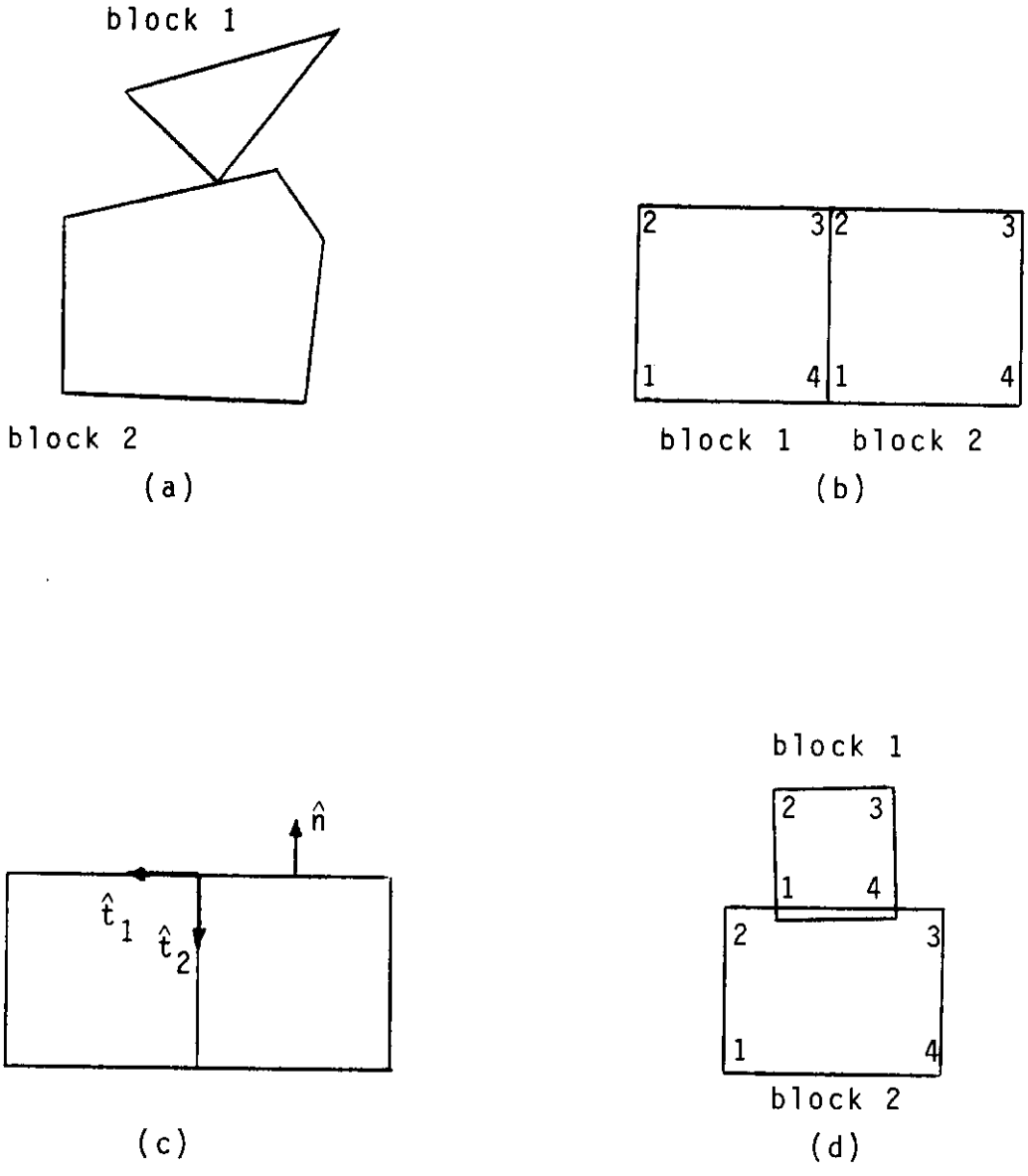


Figure 4. Contact Checks Associated with Dense Packing

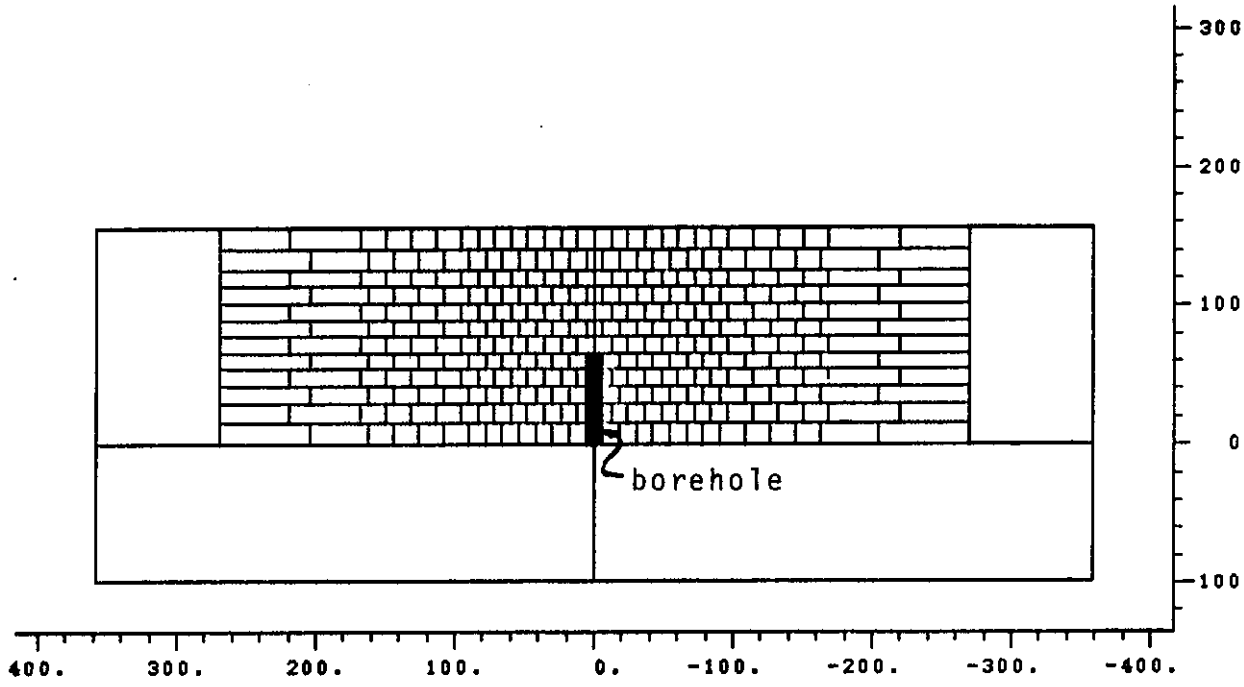


Figure 5. Initial Blocking and Geometry for Crater Problem

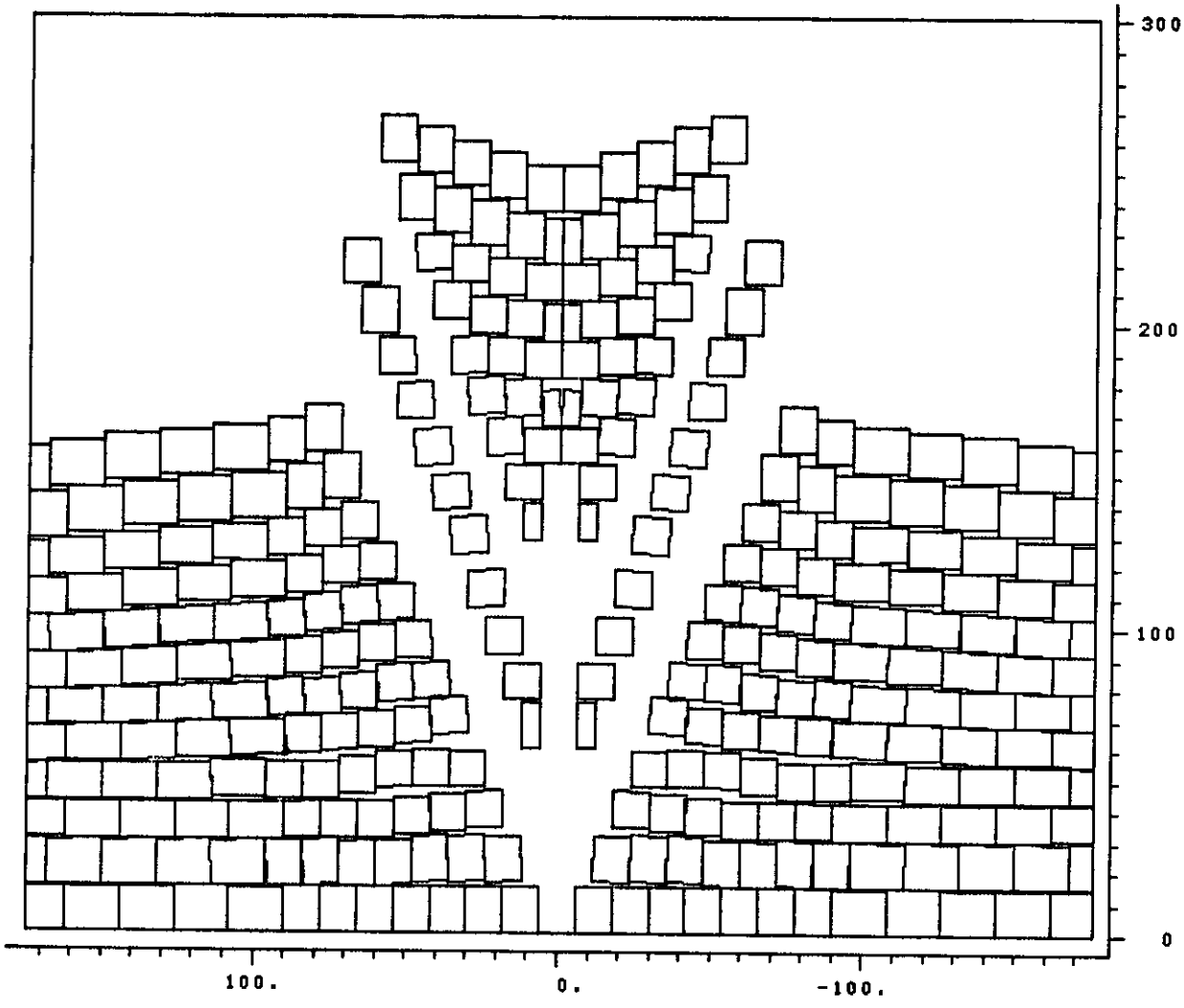
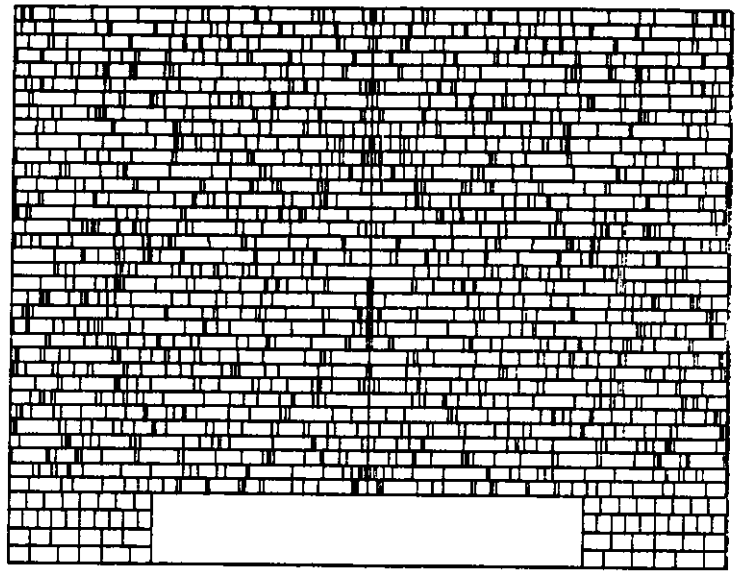
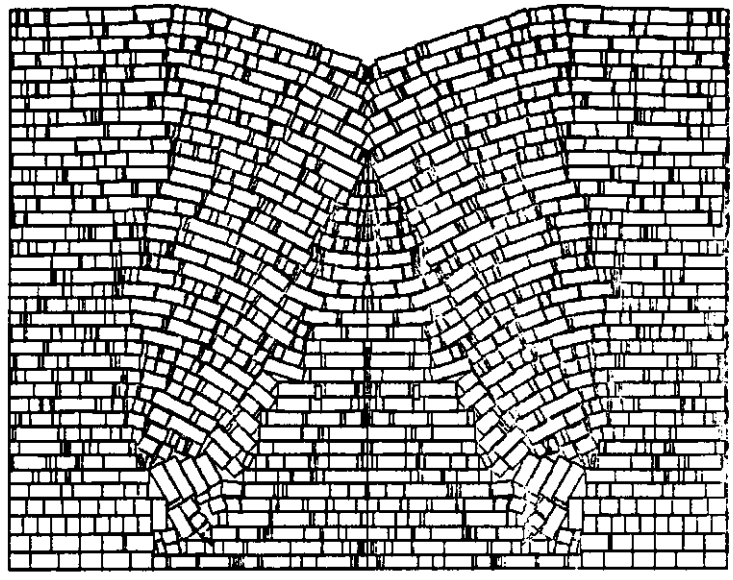


Figure 6. Crater Formation after 500 msec



(a)



(b)

Figure 7. Block Model for an Idealized Mine Structure.

DISTRIBUTION:

U. S. Bureau of Mines
Pittsburgh Research Center
P.O. Box 18070
Pittsburgh, PA 15236
Attn: C. A. Goode
J. C. Edwards (5)
V. A. Adamek
N. Moebs

U. S. Department of Energy
Division of Coal Mining
Mail Stop FE-25
Washington, DC 20545
Attn: R. Avellanet
M. Harthill

U. S. Bureau of Mines
Division of Mining Research
2401 E Street NW
Washington, DC 20241
Attn: B. Johnson
R. Bloomfield

U. S. Bureau of Mines
Denver Research Center
Denver Federal Center, Bldg. 20
Denver, CO 80225
Attn: F. Allgaier
T. Hackette

U. S. Bureau of Mines
Twin Cities Research Center
5629 Minnehaha Ave. S
Minneapolis, MN 55412
Attn: L. Powell
S. Tandanand

U. S. Bureau of Mines
Spokane Mining Research Center
E. 315 Montgomery Avenue
Spokane, WA 99207
Attn: P. Williams
B. Stewart
D. Kiley

Dept. of Energy and Natural
Resources Illinois State
Geological Survey
1615 E. Peabody
Champaign, IL 61820
Attn: R. A. Bauer
P. DuMontelle

Phil Halleck
Los Alamos National Laboratories
Mail Stop 329 G-6
Los Alamos, NM 87545

Prof. S. Benzley
Dept. of Civil Engineering
Brigham Young University
Provo, Utah 84601

Dept. of Material Science
and Materials Engineering
University of California
Berkeley, CA 94720
Attn: Prof. N.G.W. Cook
M. Hood

J. Daemen
Dept. of Mining and Geological
Engineering
University of Arizona
Mines 27
Tucson, AZ 85711

Prof. S. H. Advandi
Dept. of Engr. Mechanics
Ohio State University
Columbus, OH 43201

Prof. Z. T. Bieniawski
Dept. of Mining Engineering
Pennsylvania State University
104 Mineral Science Bldg.
University Park, PA 16802

Prof. Hon-Kim Ko
Dept. of Coal and Environ. Engr.
University of Colorado
Boulder, CO 80202

T. W. Thompson
Dept. of Petroleum Engr.
University of Texas at Austin
Austin, TX 78712

Dept. of Engineering and
Technology
S. Illinois Univ. at Edwardsville
Edwardsville, IL 62026
Attn: R. S. Pocreva
R. Yarbrough

R. C. Spec
GAI Consultants
570 Beatty Road
Pittsburgh,
Monroeville, PA 15146

V. A. Scovazzo
D'Appolonia
100 Duff Road
Pittsburgh, PA 15235

Peter J. Conroy
Dames and Moore
1550 Northwest Highway
Park Ridge, IL 60068

J. E. O'Rourke
Woodward-Clyde Consultants
Three Embarcadero Center,
Suite 700
San Francisco, CA 94111

G. Harloff
Gulf R D
P.O. Drawer 2038
Pittsburgh, PA 15230

A. L. Stevens
Occidental Reserach Corp.
2100 SE Main St.
Irvine, CA 92714

R. T. Weiss
Exxon Production Research Co.
P.O. Box 2189
Houston, TX 77001

D. S. Choi
Conoco R D
P.O. Box 1267
Ponca City, OK 74603

G. E. Holt
Exploration and Mine Planning
Group
Australian Coal Industry
Research Laboratories, Ltd.
P.O. Box 83
North Ryde, N.S.W. 2113
Australia

W. Kapp
c/o General Superintendent
Teel Division Collieries
Barclay Centre
26 Atchison Street
N.S.W. 2500 Australia

Jan Mares
Assistant Secretary for
Fossil Energy
Mail Stop B-126
Washington, DC 20545

Office of Deputy Assistant (5)
Secretary for Oil, Gas and
Shale Technology
U.S. Department of Energy
Mail Stop D-107
Washington, DC 20545
Attn: D. Uthus
A. M. Hartstein
J. W. Ramsey
H. E. Thomas
R. W. Markley

Laramie Energy Technology (3)
Center
P.O. Box 3395
University Station
Laramie, WY 82071
Attn: R. J. Jiacoletti
J. H. Weber
W. E. Little

Los Alamos National Laboratory (3)
Los Alamos, NM 87545
Attn: T. F. Adams, G-6, MS 665

Lawrence Livermore National
Laboratory
P.O. Box 808
Livermore, CA 94550
Attn: A. E. Lewis

Science Applications, Inc. (1)
1546 Cole Blvd.
Suite 210
Golden, CO 80401
Attn: R. L. Wise

Keith Britton
Blastmaster, Inc.
325 Beverly Avenue
San Leandro, CA 94577

Science Applications, Inc.
P.O. Box 1173
Steamboat Springs, CO 80477
Attn: C. Young

Cathedral Bluffs Shale Oil Co.
751 Horizon Drive
P.O. Box 2687
Grand Junction, CO 81501
Attn: A. Stevens

Geokinetics, Inc.
391 Chipeta Way D-2
University Research Park
Salt Lake City, UT 84108
Attn: M. A. Lekas

Bechtel, Inc.
2705-A Larkin Street
San Francisco, CA 94109
Attn: D. Zerga

Carbondale Mining Technology Center
Department of Energy
P.O. Box 2587
Carbondale, IL 62901
Attn: T. Ziegler

Maurice B. Dusseault
Associate Professor of Mining Engineering
Dept. of Mineral Engineering
The University of Alberta
Edmonton, Alberta T6G 2G6

J.F.T. Agapito and Assoc.
Suite 34
715 Horizon Dr.
Grand Junction, CO 81501
Attn: S. J. Mitchell

Rod Boade
Research and Development
Phillips Petroleum Co.
GB 105
Bartlesville, OK 74004

Stephen R. Winzer
Martin Marietta Laboratories
1450 South Rolling Rd.
Baltimore, MD 21227

J. Drake
Waterways Expr. Station
P.O. Box 631
Vicksburg, MS 39180

1500 W. Herrmann
1510 D. B. Hayes
1520 T. B. Lane
1521 R. D. Krieg
1521 L. M. Taylor (15)
1522 T. G. Priddy
1523 R. C. Reuter
1524 W. N. Sullivan
1530 L. W. Davison
1540 W. C. Luth
9740 R. K. Traeger
9741 J. R. Kelsey
9746 B. Granoff
9747 P. J. Hommert
9747 R. L. Parrish
9747 C. A. Romero
9747 J. T. Schamaun (5)
9750 V. L. Dugan
9752 H. M. Dodd
9752 H. J. Sutherland (5)
9753 D. A. Northrop
9755 B. W. Marshall
9756 D. Engi
3141 L. J. Erickson (5)
3151 W. L. Garner (3)
3154-3 C. Dalin (25)
(For DOE/TIC Unlimited Release
8214 M. A. Pound