AD-A061 658 DAMES AND MOORE LOS ANGELES CA F/G 8/7
COMPUTER MODELING OF JOINTED ROCK MASSES.(U)
AUG 78 T MAINI, P CUNDALL, J MARTI DACA39-77-C-0004
UNCLASSIFIED WES-TR-N-78-4 NL

1 OF 6
AD A061 658































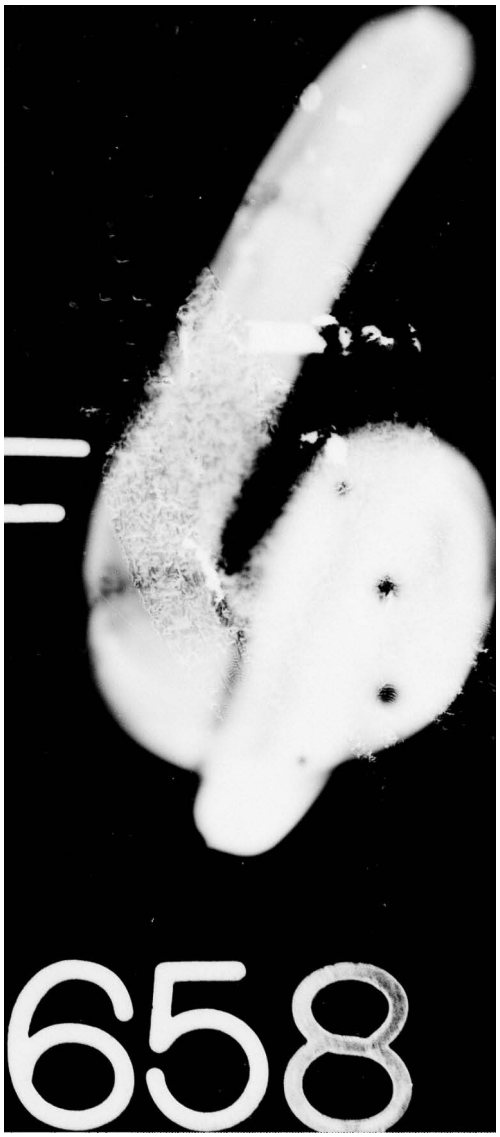










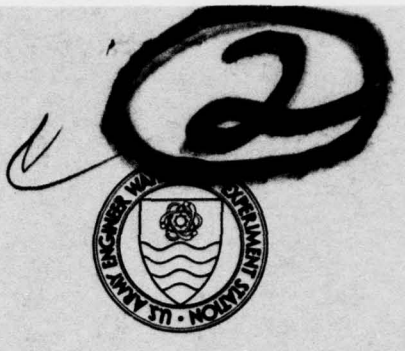


658

AD A061658

DDC FILE COPY

LEVEL II





TECHNICAL REPORT N-78-4

COMPUTER MODELLING OF JOINTED ROCK MASSES

by

Tidu Maini, Peter Cundall, Joaquin Marti
Peter Beresford, Nigel Last, Margaret Asgian

Dames and Moore
Suite 1000, 1100 Glendon Avenue
Los Angeles, Calif. 90024

August 1978

Final Report

Approved For Public Release; Distribution Unlimited



Prepared for Defense Nuclear Agency
Washington, D. C. 20305
and Office, Chief of Engineers, U. S. Army
Washington, D. C. 20314
Under Contract No. DACA39-77-C-0004
Monitored by Weapons Effects Laboratory
U. S. Army Engineer Waterways Experiment Station
P. O. Box 631, Vicksburg, Miss. 39180

DDC
RECEIVED
NOV 29 1978
D

78 11 20 109

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

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: Technical Report N-78-4

2. GOVT ACCESSION NO.

3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle): COMPUTER MODELLING OF JOINTED ROCK MASSES

5. TYPE OF REPORT & PERIOD COVERED: Final report

6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s): Tidu Maini, Peter Beresford, Peter Cundall, Nigel Last, Joaquin Marti, ~~Magaret Aegian~~

8. CONTRACT OR GRANT NUMBER(s): Contract DACA39-77-C0004

9. PERFORMING ORGANIZATION NAME AND ADDRESS
Dames and Moore
Suite 1000, 1100 Glendon Avenue
Los Angeles, Calif. 90024

393 643

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: -C-

11. CONTROLLING OFFICE NAME AND ADDRESS
Defense Nuclear Agency, Washington, D. C. 20305
and Office, Chief of Engineers, U. S. Army
Washington, D. C. 20314

12. REPORT DATE: August 1978

13. NUMBER OF PAGES: 399

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)
U. S. Army Engineer Waterways Experiment Station
Weapons Effects Laboratory
P. O. Box 631, Vicksburg, Miss. 39180

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

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

12 404 p.

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

18 WES 19 TR-N-78-4

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Computer programs
Computerized models
Dynamic loads
Jointed rock
Rock deformation
Rock masses

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

This report describes work that forms part of an effort to explain and predict the phenomenon of "block motion" which can occur in jointed rock when exposed to dynamic loading. The objectives of the study were to: (a) evaluate a general method for modelling jointed rock; (b) translate the original Deformable Element Technique (DET) written in machine language into standard FORTRAN; (c) develop a method for allowing blocks to crack and break into separate elements;

(Continued) next page

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

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

S/O 393643

20. ABSTRACT (Continued).

and (d) conduct a review of the behavior of rock joints and develop an improved constitutive law for rock block interactions. The major objectives of the study were achieved.

A numerical scheme for treating a fully deformable block was demonstrated to give accurate results. It was shown that very little error was introduced by the calculation of siding rock joints by the various rezoning schemes used. Although the new formulation is not likely to be more efficient than existing lagrange, finite difference codes, it was shown to have two major advantages; namely, it is completely general and can completely model any arbitrary jointing pattern and the joints are modeled accurately with no interpolation necessary at interface.

Other goals were accomplished. The original rigid block program was translated into FORTRAN Code, RBM.

A new idea for treating simple block deformability was developed. Each block was given three degrees of freedom to deform internally, with general constitutive laws given for the intact material. The method differs fundamentally from finite elements and finite differences in that it relies upon the stiffnesses of joints to link neighboring elements or zones. The new program, SDEM, is only slightly slower than the rigid block program and is useful in cases where the intact deformation of rock blocks is important but not large.

A modified version of the rigid block program, RBMC, was written which allow blocks to crack and divide into separate blocks in response to the loads acting on them. A simple cracking criterion was used which was based on empirical point-load tests on irregular blocks.

An extensive literature survey was made on the properties and behavior of rock joints. Based on these findings, a constitutive law was proposed for rock joints and coded into the subroutine JOINT.

Listings for all programs developed under this study are provided in appendices.

PREFACE

This report was prepared by Dames and Moore, Los Angeles, Calif., under Contract No. DACA39-77-C-0004 with the U. S. Army Engineer Waterways Experiment Station, Vicksburg, Miss. The work was jointly sponsored by the Defense Nuclear Agency and the Office, Chief of Engineers, U. S. Army, and was monitored by the Weapons Effects Laboratory (WEL), WES.

The studies were conducted by the Dames and Moore, Advanced Technology Group, and involved the following personnel: Dr. Peter Cundall, Principal Investigator; Dr. Joaquin Marti, Project Manager; Mr. Peter Beresford, Project Engineer; Mr. Nigel Last, Project Assistant; Ms. Margaret Asqian, Project Assistant; and Dr. Tidu Maini, Principal-in-Charge.

Contract Monitor for WES was Mr. James L. Drake of WEL. Dr. Eugene Sevin and MAJ Dave Spangler of DNA actively followed the progress of the contract and offered general guidance. Mr. Jerry S. Huie of the Soils and Pavements Laboratory, WES, managed the OCE program which supported part of this contract.

During the period of the contract, Director of WES was COL John L. Cannon, CE. Technical Director was Mr. F. R. Brown. Mr. William J. Flathau was Chief of WEL.

LEVEL II

ACCESSION for
NTIS White Section
DDC Buff Section
UNANNOUNCED
JUSTIFICATION
BY
DISTRIBUTION/AVAILABILITY CODES
Dist. AVAIL. and/or SPECIAL
A

DDC
RECEIVED
NOV 29 1978
D

78 11 20 109

TABLE OF CONTENTS

		page
	Chapter 1: OVERVIEW	1
1.1	Introduction	2
1.2	Historical Perspective	2
1.3	Present Work	4
1.4	Layout of Report	8
1.5	Conclusion	8
	Chapter 2: BASELINE PROGRAM, RBM	9
2.1	Introduction	10
2.2	Fortran Implementation	11
2.3	Departures from Original Logic	14
2.4	Use of Program	17
2.5	Examples and Validations	27
2.6	Conclusions and Suggestions	39
	Chapter 3: SIMPLY-DEFORMABLE PROGRAM, SDEM	53
3.1	Introduction	54
3.2	Simplified Explanation of Deformable-Block Approach	55
3.3	Sequence of Operations	62
3.4	Mathematical treatment	63
3.5	Changes to Program RBM	68
3.6	Examples and Validations	70
3.7	Conclusions and Recommendations	77
3.8	Acknowledgement	78
	Chapter 4: INTRODUCTION OF CRACKING	85
4.1	Introduction	86
4.2	Changes to Data Structures	88
4.3	Introduction of a Crack into a Block	92
4.4	Use of Program	97
4.5	Example Runs	98
4.6	Conclusions	100
	Chapter 5: BEHAVIOUR OF ROCK JOINTS	109
5.1	Introduction	110
5.2	Literature Survey	111
5.3	Constitutive Laws	115
5.4	Conclusions	127

page

Chapter 6: FULLY-DEFORMABLE BLOCKS 150

6.1 Introduction 151
6.2 Basic Equations and Parameters 155
6.3 Mesh Generation and Rezoning 165
6.4 Use of Program DBLOCK 178
6.5 Examples and Validations 181
6.6 Conclusions 191

Chapter 7: OVERALL CONCLUSIONS 208

7.1 Achievements of the Study 209
7.2 Applicability and use of the Computer Programs 212
7.3 Suggestions for Future Work 213

Appendix I: REFERENCES 215

Appendix II: BIBLIOGRAPHY ON JOINT PROPERTIES 216

Appendix III: INPUT COMMANDS FOR RBM AND EXAMPLE RUN 224

Appendix IV: SUBROUTINE GUIDE TO RBM AND PROGRAM CHANGES 229

Appendix V: DERIVATION OF EFFECTIVE MASSES - SDEM 233

Appendix VI: STRESS ROTATION CORRECTION TERMS 237

Appendix VII: SUBROUTINE GUIDE, PROGRAM RBMC 241

Appendix VIII: LIST OF SYMBOLS USED IN CHAPTER 6 243

Appendix IX: PROPERTIES OF TRIANGLES USED IN DBLOCK 244

Appendix X: INPUT CARDS FOR PROGRAM DBLOCK 246

Appendix XI: SUBROUTINE GUIDE - PROGRAM DBLOCK 247

Appendix XII: LIST OF VARIABLES AND PROGRAM LISTING FOR RBM 253

Appendix XIII: LISTING OF PROGRAM SDEM 280

Appendix XIV: LISTING OF PROGRAM RBMC 312

Appendix XV: LISTING OF PROGRAM JOINT 354

Appendix XVI: LISTING OF PROGRAM DBLOCK AND LIST OF FORTRAN NAMES 358

CHAPTER 1: OVERVIEW

1.0 The work described in this report is fitted into the context of the present state-of-the-art and the problems that need to be solved. A guide is given for the remainder of the report.

1.1 INTRODUCTION

This report describes work that forms part of an effort by the Defense Nuclear Agency to explain and predict the phenomenon of "block motion" that can take place in underground structures exposed to strong dynamic loading. Large displacements across rock joints are known to have occurred during such events. These displacements at once pose great dangers to the structures, and present difficulties as far as analysis is concerned. Standard calculation methods that treat the rock as a continuum are clearly inappropriate, but it is not always easy to incorporate discontinuities in a realistic way, and to prevent the computer programs from becoming large and unwieldy.

1.2 HISTORICAL PERSPECTIVE

Lagrangian, finite-difference programs are widely used in the Defense Community, and have evolved from pure hydrodynamics codes by utilizing a non-zero shear modulus. Yield and failure can be modelled, but the formulation remains essentially that of a continuum. Discontinuities may be incorporated by means of slide-lines, a technique that was described by Wilkins (1969); however the procedure becomes difficult and expensive when multiple joint sets must be represented. Part of the reason for this is that the formulation has evolved historically from a continuum approach that has been successively modified to treat greater and greater degrees of discontinuity.

Perhaps it would be better to work the other way: to introduce continuum behaviour into a method that is designed to model only discontinuities. It may turn out, of course, that the two approaches will lead to essentially the same final formulation, in which case nothing would have been gained. However it was thought to be worth a try.

The starting point was the distinct element method (DEM) that was originally proposed in a restricted form by Cundall (1971) and generalised later on (Cundall, 1974). It was developed for low-stress rock situations, where displacements due to joints far exceed those of the intact rock blocks. The simplifying assumption was made that the intact rock was rigid, and that only the joints could deform. The method works explicitly in time, and permits large block movements and rotations. A novel housekeeping scheme keeps track of all blocks in an efficient manner, and allows any block to interact with any other block. The notion of treating blocks as rigid was also pursued by several other workers, using the finite element method (e.g. Burman (1971), Chappel (1972, 1974)). However these formulations were restricted to small displacements, and used an implicit solution method, which brings with it a whole host of difficulties when modelling problems that have strong geometrical or material non-linearities.

1.3 PRESENT WORK

1.3.1 DEFORMABLE-BLOCK PROGRAM - DBLOCK

As its main topic, this report presents an evaluation of a general method for modelling jointed rock. In contrast to the updated hydrodynamics codes referred to above, the new work regards the joints and discontinuities as the most important components, and throws the burden of approximation onto the continuum formulation. As blocks slide over one another and make and break contact, continuum zones are created, deleted and re-zoned. The errors associated with this process are evaluated in this report. To summarize, the two approaches may be compared as follows:

New deformable-block program:

Continuum Formulation	Joint Formulation
• Subject to error due to re-zoning - triangular zones created, deleted and modified due to contact changes.	• More or less exact - the force/displacement law is used directly between grid-points; • Arbitrary block complexity is treated as standard.

Traditional Lagrangian code - e.g. HEMP:

Continuum Formulation	Joint Formulation
• No significant errors introduced by slide-lines.	• Subject to error due to interpolation necessary on slide-lines; • Only simple interaction geometry possible.

The deformable-block program, DBLOCK developed here is simply a "test-bed" code to prove and demonstrate the proposed formulation. It is not intended to be used to run complex problems, since the housekeeping logic used by Cundall (1974) has not yet been incorporated in the program. This will be the subject of a further contract, if the method proves successful. The point will be taken up again in the final chapter.

1.3.2 RIGID-BLOCK PROGRAM - RBM

Since the original program written by Cundall (1974) was partially written in machine language, part of the present contract was concerned with translating the machine language program into Fortran. The resulting program, RBM, is to be regarded as a "base-line" program that can be used by interested parties as a base on which more complex codes can be built. For this reason the program has been kept deliberately free of embellishments and complexities that would tend to obscure the workings of the program.

1.3.3 SIMPLY-DEFORMABLE BLOCK PROGRAM - SDEM

The general deformable block program (Section 1.3.1) is likely to be complex and possibly as expensive to run as regular HEMP-type codes, although it is anticipated that the results for jointed media will be superior. Such an approach is clearly necessary when each rock block deforms internally in a complex manner. Complex deformation implies many

degrees of freedom, which can only be represented by internal discretization or multi-noded zones (as in higher order finite elements). Multi-noded zones are not desirable in an explicit scheme as each zone must be treated implicitly. We appear to be stuck with internal discretization if we really need to represent complex modes of intact deformation.

There may by an argument for trying a simpler approach for those problems where internal block deformation is of secondary importance, but still has to be accounted for. This line of development has led to program SDEM, which is basically an enhanced version of the rigid-block program, RBM. SDEM gives each block three degrees of freedom to deform internally, in addition to the three rigid-body modes already associated with each block in RBM. The formulation is for large strains, and can use any form of constitutive law, such as plasticity.

The program gives the user a good deal more realism for intact behaviour, without too much degradation in efficiency compared to RBM.

1.3.4 CRACK FORMULATION

Although it is extremely complex computationally to model the complete mechanics of crack propagation accurately in rock, it was considered worthwhile to modify program RBM so that any block could split into two as a function of the loads applied to it and utilize semi-empirical laws to determine when the block would crack. There is quite a large body of

experimental data that enables crude estimates to be made of the point-contact loads that will cause rock blocks of given sizes and shapes to crack. In this way the program would be given added realism without sacrificing efficiency by too much. In many situations it is expected that the progressive movements in a rock mass are insensitive to the exact fracture loads, so long as the fractures do occur. Without any possibility of cracking, the behaviour can be most unrealistic, with corners "locking-up" even though the overlap is very small. When blocks are allowed to crack, the criterion for cracking may be varied in parameter studies to determine what effect it has on overall behaviour.

Chapter 4 describes the modifications necessary to RBM in order to allow cracking to occur. Suggestions are made concerning the choice of crack criterion.

1.3.5 JOINT FORMULATION

Whichever of the approaches described above is selected to model real situations, a constitutive law is needed to represent rock joints. The formulation should be general enough to accomodate the whole spectrum of observed behaviour. To this end, a literature survey has been made of field and laboratory tests on jointed rock. A summary of the findings is given in Chapter 5, together with a suggested framework for a constitutive law. A few tests have been made using this law in conjunction with program DBLOCK, but many more tests are anticipated when simulations of real events are made in the next phase of the work.

1.4 LAYOUT OF REPORT

Each of the topics discussed above is treated in detail in its own chapter. The figures for each chapter are collected together at the end of the chapter, but references are "global" and appear in Appendix I. Each chapter is more or less self-contained, with an introduction and conclusion, but global conclusions are presented in Chapter 7.

1.5 CONCLUSION

The work described in this report should be regarded as open-ended. Several approaches to the problem of modelling block motion have been investigated, with a view to using one or all of them in the next phase, which will be to simulate actual events, and compare predictions with measurements. The discussion of future possibilities will be left for Chapter 7.

CHAPTER 2: BASELINE PROGRAM, RBM

2.0 *The original distinct element program developed by Cundall (1974) was written in machine language. This chapter describes the translation of that program into Fortran. Examples, validations and a users' guide are given.*

2.1 INTRODUCTION

The original, general implementation of the distinct element method was a program written in machine language, and described in a report by Cundall (1974). It is assumed that the reader is familiar with that report, since it has been followed almost exactly in the Fortran version of the program described in this section. Few departures have been made from the original logic even though many potential improvements could be envisaged. This was in line with the philosophy that the Fortran program should be a "baseline program": that is, a simple, easily-understood program to form a base from which other, more flexible codes could evolve Section 2.6 contains several suggestions for improvements. Some of these have already been incorporated into the program SDEM, described in Chapter 3. Almost all of the improvements are built into another program, BALL, which models the mechanics of assemblies of circular discs. This program is described by Cundall (1978).

The baseline program, BBM (described in this section) models assemblies of arbitrary, angular blocks. There is no restriction, apart from memory limits, on the geometry, displacements and rotations of the rock blocks. Any block may touch any other block. The response to imposed forces and constraints is calculated using an explicit integration scheme, thereby allowing large displacements and general force/displacement laws to be handled in a straightforward manner.

The original program was written in assembly language for a computer with an interactive graphics display. This note describes an implementation of the calculation section of the program in a high level language; the program was not written for an interactive display, but it contains simple routines for drawing blocks on a standard plotter (e.g. Calcomp). The approach taken in writing the program was that it should be as machine-independent as possible. Consequently it is somewhat inefficient in its use of memory: for example whole (32-bit) words have been used for flags, which were stored as single bits in the original program.

The following sections have some notes on the Fortran implementation, the departures from the original logic, notes on the use of the program and some examples and validations. A list of input commands is given in Appendix III, a subroutine guide in Appendix IV and a program listing in Appendix XII.

2.2 FORTRAN IMPLEMENTATION

The original program was written in assembly language for a small minicomputer with no floating point processor. A Fortran implementation on that computer would have been very slow to run, and would have allowed only a small number of blocks to be modelled. In order to make the program more portable, it has been rewritten in Fortran, and it should be possible to implement it on most computer systems with little difficulty.

The original program was written in three phases, the first two for describing the model, and the third for allowing the blocks to respond to imposed forces and constraints. This Fortran version is concerned with the third (calculation) phase, and the input has deliberately been kept rudimentary.

In order to make the program as machine independent as possible, the following conventions have been observed:

(i) Word lengths are identical (i.e. integer and real variables are the same size).

(ii) Only four characters are stored in a word for alphanumeric representation.

(iii) Input and output is standard sequential access, for both formatted and unformatted files, and output formats use Hollerith strings.

The allocation of memory is performed in a manner similar to the original program, with all the linked list arrays stored in a single memory partition, the allocation of this memory depending upon the particular problem being analysed. In this way no storage is wasted. A map of this memory partition is shown in Figure 2.1.

The data arrays for the blocks, the box array and corner list, and the contact array and list are shown in Figures 2.2, 2.3 and 2.4 respectively. As noted earlier, each member of these arrays uses a single storage location. Substantial storage economies can be effected on installations allowing variable length words, but this practice was deliberately avoided here.

The original program was written for use with a CRT (cathode ray tube) display, and depended heavily upon this for its input and output (I/O). The present program was written specifically for a system without this capability, the primary purpose being to implement the data structures and physical algorithms in a high level language, and to leave the I/O to the user. In any case, sophisticated I/O is governed by the machine being used, not to mention the preferences of the programmer. The present program requires the user to describe his problem in card image format (the data input description is given in Appendix III) and the primary form of output is the pen plot (a snapshot of the problem at a given time). An echo of the data input stream is produced on the line printer, as well as the capability of printing the data lists and other information.

It is very easy to write an inefficient FORTRAN program, especially when it is essentially iterative in nature. The algorithm employed in the present program involves many thousands of timesteps,

and the computing time is directly proportional to the number of timesteps. Any savings in computer time effected in the iteration cycle would be very beneficial. To this end, the number of arguments in subroutine calls has been kept to a minimum. As a consequence of this practice, and of the petty restriction in ANSI FORTRAN that variables cannot be equivalenced to subroutine arguments, the program is a little less easy to read than it might be. However, reading the listing in conjunction with the data structure diagrams eliminates most of the obfuscation.

2.3 DEPARTURES FROM ORIGINAL LOGIC

This section describes the differences between the logic of the original program and the Fortran version.

2.3.1 NEW ENTRIES TO LISTS

New entries to lists are appended rather than inserted at the beginning. This was done purely for convenience, since the variables associated with the end of the list are available when extension is required.

2.3.2 UPDAT

Routine UPDAT is no longer called at regular intervals, but an algorithm has been implemented that automatically requests the updating of the contact lists when necessary. At each timestep, the maximum magnitude

of all block velocities is determined and integrated to give a fictitious increment in displacement. These displacement increments are summed, producing a maximum possible displacement since the last update. This displacement is usually fictitious. When it exceeds the specified tolerance, the update routine is called. The advantage of this algorithm is that UPDAT is called frequently at times of rapid motion, but is seldom called when the model is approaching equilibrium.

2.3.3 EMPTY LIST

The empty list, containing all unused contacts is initially created to the end of the available memory, so that the concept of additional storage, after this list has been exhausted, is no longer valid.

2.3.4 FIX FLAG

When a fixed block has been plotted, its "fix flag" is incremented by 2 so it will not be plotted again. This modification was made because of the relatively slow speed of pen plotting compared with CRT.

2.3.5 DRIFT CORRECTION

Due to the higher precision of the arithmetic in the Fortran version, the drift correction was assumed to be unnecessary, and was omitted. However

if a corner penetrates more than one unit into an edge, a warning message is printed out. This usually means that the program is being mis-used.

2.3.6 DAMPING

Both mass-proportional and stiffness-proportional damping are available, and are discussed in detail in Section 2.4.6.

2.4 USE OF PROGRAM

2.4.1 FILES

The following logical units are used by the program:

1. Restart file
(3) Plot file (only used in PDP 11/45 version)
5. Input file
6. Output file

The Restart file is written by the program whenever it stops normally, and is used when the program is started again to restart from the same point.

The Plot file is written whenever a PLOT command is given.

Logical Unit 5 expects a sequence of input commands; these are given in Appendix III.

The printer output appears on Logical Unit 6. All input commands are echoed on output, so that Unit 6 serves as a logging device.

2.4.2 CHOICE OF COORDINATES

The present version of RBM does not allow the user to choose an arbitrary coordinate space, for reasons that will be outlined below. This means that a problem posed in an unacceptable coordinate system must be transformed before using RBM, and the results transformed back again after the run. Suitable transformation logic could easily be built into RBM if desired, but has not been done so in order to keep the program as simple as possible.

Both reboxing and contact searching are triggered by blocks crossing integer boundaries. That is to say, when the integer part of either the x- or the y- coordinate of a block centroid changes, the routine REBOX is called for that block. REBOX determines whether the box entries for the block are correct, and if they are not, it creates and deletes entries as necessary. UPDAT, which is also triggered by an integer boundary being crossed, creates new contacts if they should exist but do not. The utilization of the integer parts of the coordinates to initiate reboxing and updating clearly limits the coordinate ranges that can be used, but a good deal of memory is thereby saved; otherwise old coordinates would have to be stored in order that cumulative movements could be calculated.

The following restrictions apply to the use of RBM:

i) The coordinate origin forms the lower, left-hand point of the problem space (box area).

ii) Block dimensions should be in the order of 10 to 100 coordinate units. Dimensions outside these limits can be used, but the user should know what he is doing.

iii) The number of boxes should be between 1 and 1/10 of the number of blocks in the problem, with a minimum of, say 16.

Before running a problem, the box area must be chosen. This area should be slightly larger than the greatest anticipated area that the system of blocks will require. At present, the program automatically fixes a block if it tries to move outside the box area. A suitable number of boxes to cover the box area should then be chosen, bearing in mind the considerations given above. Blocks may then be created.

Although the choice of coordinates should have no effect on the physics, it will influence the efficiency of the program, which will depend on the sizes of the blocks expressed as integers. The following effects may be noted:

Large numerical values of block dimensions will give:

1. minimum memory requirements, since contact space will only be allocated for particles that are very close;

2. increased computation time, since reboxing and updating will be done very frequently.

Small numerical values of block dimensions will give:

1. high memory requirements, since contact space will be retained when blocks separate to large distances;

2. low computation time, since reboxing and updating will be triggered infrequently.

For a problem that is critical in time or storage, experiments can be made to determine an optimum coordinate range.

The number of boxes also influences speed and storage. Many boxes will increase the storage requirements, but decrease search time; few boxes will decrease storage requirements, but increase search time.

2.4.3 SEQUENCE OF OPERATIONS

The commands given to the program (described in Appendix III) are divided into two sets. The first command must be either START or RESTART. If it is RESTART, the program reads an existing restart file and jumps to data set number two. A START command, on the other hand, must

be followed by a specification of the number of blocks in the problem, the number of boxes and their sizes and the fraction of critical time-step to be used. Data set two may then be given to the program.

The program RBM is similar to the original machine language version in that the set of blocks, once created, cannot be added to. In practice this means that no further CREATE commands may be given after a CYCLE command has been used. All other commands in set two may be given at any time.

2.4.4 PRINTOUT

Each input line is echoed on output, followed in some cases by an informative or error message. The CREATE command is followed by the coordinates of the block that have been read, together with the computed centroid coordinates, the mass and the moment of inertia.

The DUMP command produces, initially, a summary of the program pointers and other internal variables. On request, it will follow this with a list of box entries, block data and contact data. These printouts are explained below:

2.4.4.1 Box entries: These are simply given as a list of block corners that map into each box.

2.4.4.2 Block data: The headings are self-explanatory, except perhaps for the following:

NB . . .	block number
IF . . .	"fix" flag
NC . . .	number of corners
THETA . .	this is the angle that the block has moved through since THETA was last set to zero. THETA is set to zero when its magnitude exceeds 0.01 radians. The actual angle can be deduced from the COS and SIN printouts.

2.4.4.3 Contact data: The symbols are as follows:

NBE . . .	block number corresponding to the edge of the edge/corner pair
PRE . . .	preserve flag
NPE . . .	number of the edge involved in the contact
NPC . . .	number of the corner involved in the contact
NBC . . .	block number corresponding to the corner of the edge/corner pair
LINK . . .	pointer to next contact
S, N . . .	not used at present, but intended for storing comulative shear and normal displacements, if required for a non-linear law.
FN, FS . . .	normal and shear forces
SIN, COS . . .	sine and cosine of the angle of edge (of the edge/corner pair) to the global x-axis of the run
XCP, YCP . . .	coordinates of contact point.

2.4.5 TIME-STEP

With any explicit program, the critical time-step is determined by the highest eigenvalue (highest natural frequency) in the system. If the time-step is made larger than critical, numerical instability results. This form of instability is usually obvious when it occurs, but may be masked if mass-proportional damping is used or energy is being dissipated by sliding.

If there is any doubt about the results, the run should be repeated with lower time-step and the results compared. The program calculates a critical time-step, but one that is based only on the oscillation of a single degree-of-freedom system with the lowest mass and highest stiffness in the problem. The user must reduce this time-step still further by specifying a fraction (FRAC) by which the computed time-step is multiplied. This is necessary because the apparent stiffness acting on a block increases as it becomes surrounded by other blocks. A value of FRAC of 0.1 is probably safe for most problems, but 0.2 to 0.5 may be used with caution for loosely-packed assemblies. If stiffness-proportional damping is used, a lower time-step is needed for stability, but mass-proportional damping does not influence the numerical stability.

2.4.6 DAMPING

Two forms of viscous damping are incorporated into the formulation embodied in RBM. Physically these correspond to:

a) dashpots from block centroids to "ground";

b) dashpots across contacts.

The dashpots across the contacts operate both in the shear and normal directions; the shear dashpot is "switched off" during sliding. In finite-element notation, the damping is described as follows:

$$[c] = \alpha[m] + \beta[k]$$

where $[c]$ is the damping matrix

$[m]$ is the mass matrix

$[k]$ is the stiffness matrix

α and β are constants, given by

$$\alpha = \lambda_{min}\ \omega_{min} \text{ and}$$

$$\beta = \frac{\lambda_{min}}{\omega_{min}}$$

where $f_{min} = \frac{\omega_{min}}{2\pi}$

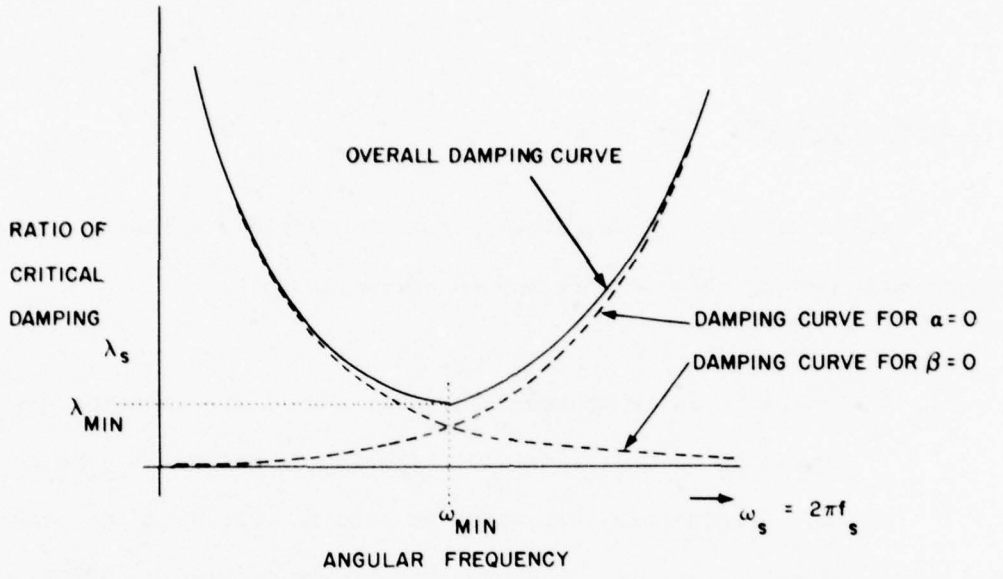
For an elastic continuous system (i.e. one in which there is no slip or breaking and making of contacts), the damping scheme described above is termed Rayleigh damping (e.g. see Seed et al., 1970). It damps the natural modes of oscillation of the system as follows:

$$\lambda_s = \frac{1}{2}\left(\frac{\omega_s}{\omega_{min}} + \frac{\omega_{min}}{\omega_s}\right)\lambda_{min}$$

where λ_s is the fraction of critical damping for mode s with angular frequency ω_s.

ω_{min}, λ_{min} are input parameters described on the plot below.

The equation given above shows that the damping varies with frequency, and in particular there is a minimum value for damping. This is shown in the following plot:



As remarked earlier, the theory reported above applies to an elastic, continuous system, which has natural modes of oscillation. For a discontinuous system that dissipates energy in slip the theory does not apply, but damping still occurs, and can be understood in terms of the physical effects of each type of dashpot. The mass-proportional damping has an effect similar to that of immersing the block assembly in a viscous fluid; i.e. absolute motion relative to the frame of reference is damped. Stiffness-proportional damping on the other hand is equivalent to dashpots connected across the contact stiffnesses (springs); in this case block-to-block motion is damped, but absolute motion is not.

Either type of damping can be used separately or together. Mass-proportional damping is effective in reducing low frequency motion where the whole block assembly "sloshes" from side to side. Stiffness-proportional damping is more effective against the high-frequency noise of individual blocks "rattling" against their neighbours.

2.4.7 KNOWN PROBLEMS

Errors may occur if a block is rotating rapidly without translational motion; these errors may be of two forms:

i) Reboxing is triggered only after cumulative rotation has built up to 0.01 radians. Reboxing of corners may be needed more frequently than this for long blocks, with the result that, either an error message will be printed by REBOX, or UPDAT will fail to find a contact, and allow blocks to interpenetrate.

ii) Updating of contacts is triggered only on cumulative translational motion, so that new contacts may be missed for blocks that are only rotating. Again, the effect will be for blocks to interpenetrate.

The errors described above are unlikely to occur for normal runs. If they do, it will be immediately obvious - either blocks will interpenetrate, or the program will halt with an error message. The problem may be overcome by storing global coordinates for corners, instead of local coordinates. Updating and reboxing could then be triggered directly from true corner movements, regardless of whether they were caused by translation or rotation. Such a scheme is used in program SDEM, described in Chapter 3.

2.5 EXAMPLES AND VALIDATIONS

This section documents some of the validation runs that have been made with RBM to verify that it works as intended. During the course of these tests some bugs were found, and some changes made to the pre-release version of the program that was made available in September 1977. For completeness these changes are documented in Appendix IV; the current version of RBM is listed in Appendix XII.

2.5.1 VALIDATION 1 - Block on plane

This qualitative example illustrates a block sliding down a plane. The coefficient of friction is firstly set so that the block accelerates under gravity, secondly to bring the block to rest, then thirdly to allow it to accelerate again and topple off the edge of the plane. This final event is illustrated in Figure 2.5.

2.5.2 VALIDATION 2 - Impact of two blocks

This example was intended to check for symmetry during impact, when two identical blocks are allowed to collide. It is worth noting that the contact forces are transmitted at the block corners only. Energy and momentum are conserved, and there is no angular motion after the impact, even in the extreme case of point-to-point contact in the second run. Figure 2.6 shows the results.

2.5.3 VALIDATION 3 - Conservation of energy and momentum

Figure 2.7 shows the effect of varying the timestep on the energy and momentum after the eccentric impact of two blocks. The kinetic energy and total momentum after impact are compared with the initial values, and the kinetic energy is plotted against fraction of critical timestep. Linear momentum is conserved exactly in all examples. It is observed that total momentum is conserved, and that kinetic energy is conserved as the timestep tends to zero.

The reason kinetic energy is not conserved exactly for higher values of timestep is that the program does not compute exactly the time of making or breaking of the contact. When the two blocks come into contact, the resulting mass/spring system starts to execute part of a cycle of simple harmonic motion. Theoretically, the release of the contact should occur when the force between the two blocks passes through zero. However this will not be the case if an integral number of time-steps does not fit into half a cycle of oscillation, since the release time is only taken to the nearest time-step, Δt. It is quite simple to overcome this inaccuracy by employing logic that determines the exact release time, and interpolates the force accordingly. This was not done, for several reasons:

a) to save computing time;

b) Δt would be small anyway, due to the constraints imposed by a dense packing of blocks;

c) the program would not generally be used to model impact problems.

2.5.4 VALIDATION 4 - Blocks falling into a receptacle

The object of this test was to check the reboxing and updating logic for multiple blocks and complex interactions. The example illustrates a number of blocks being allowed to fall under gravity into a fixed receptacle, and coming to rest where they have fallen (Figure 2.8). Inspection of the contact and box lists using the DUMP command showed that the logic was working as intended.

2.5.5 VALIDATION 5 - Block falling onto plane

The object of this test was to verify that the accelerations and decelerations both in free fall and in sliding were correct.

The initial conditions (time t = 0) for this validation are presented in Figure 2.9. The coefficient of friction (μ) was chosen to be slightly greater than tan α so that the free block would be retarded after it started sliding on the plane. The fraction of critical time-step was taken to be 0.1, in view of the considerations presented in the previous section.

Since the problem was basically one of a dynamic nature, the important damping effects were those due to stiffness damping and so the mass damping term of the Rayleigh equation was suppressed. The fraction of

critical damping at the minimum frequency f was taken as 0.5, and f was calculated as the frequency of the single degree-of-freedom system comprising the mass of the block and a single contact stiffness.

Validation 5 (case A) was run from time t = 0 with the position of the free block being plotted and numerical data being obtained after two periods of 500 cycles. The block was then just above the inclined plane; after this it was necessary to reduce the periods to 100 cycles in order to observe the details of the motion. Graphical output is presented in Figure 2.10(a).

Figure 2.10(b) shows the variation of the velocities in the x and y directions with time, from which it is evident that after 2,500 cycles the accelerations have become constant. The falling block has been through three distinct stages:

i) constant acceleration due to gravity;

ii) an impact resulting in the block toppling over onto one side;

iii) constant retardation due to the effect of friction between the block and the plane.

Stages i) and iii) can be examined analytically since they represent simple motion in a straight line under the effects of constant forces.

Stage i)

For a free falling body with zero initial velocity the following equations of motion can be applied:

velocity, $v = gt$

distance, $s = \frac{1}{2}gt^2$

For a time t represented by 1,000 cycles those equations predict $v = -62.04$ and $S = -196.20$. The program gives values of -62.0 and -196.0 respectively.

Stage iii)

For a body sliding down a rough inclined plane under the action of a constant force P

$$P + \mu mg \cos\alpha - mg\sin\alpha = 0$$

and since $p = ma$

$$a = g(\sin\alpha - \mu\cos\alpha).$$

For $\mu = 0.30$,

$$a = -0.476$$

And in the coordinate directions

$$a_x = 0.461$$

$$a_y = 0.115$$

The program indicates values of 0.459 and 0.114 respectively.

The run was repeated with a higher coefficient of friction (case B) and the results are shown in Figures 2.11(a) and 2.11(b). With this slightly higher value the block comes to rest while still on the plane. For this case the theoretical predictions are:

$$a_x = 0.922$$

$$a_y = 0.231$$

The program indicates values of 0.919 and 0.231 respectively.

An examination of the contact forces reveals the expected results. The condition FS = μFN is maintained almost exactly as the block slides down the plane and comes to rest. The forces on the stationary free block are those predicted from static theory.

The conclusion to be drawn from this validation is that the friction formulation and the MOTION subroutine appear to be functioning correctly. Any small discrepancies in the numerical data probably result from the fact that the free block undergoes small oscillations about its centroid as it travels down the plane.

2.5.6 VALIDATION 6 : TOPPLING BLOCKS

Goodman and Bray (1976) present an analysis of a system of toppling blocks using limiting equilibrium methods. They analyse the system of blocks shown in Figure 2.12(a). For various values of the coefficient of friction, the analysis gives the magnitude of the toe force, T, necessary for the system to be just on the point of collapse. The mode of failure is also predicted: in the case illustrated, the three upper blocks remain stable, the lower four blocks slide on the base, and the remaining nine blocks rotate about their lower corners.

Figure 2.12(b) shows the system of blocks as set up for the run with RBM. The dimensions of the blocks were ten times those used by Goodman and Bray (G&B) in order that contacts would be detected within a small fraction of the block dimensions. Consequently block weights and forces were one hundred times those used by G&B. Two runs were made: the first was to determine the coefficient of friction required for the blocks to be stable with T=0 i.e. zero toe force . The second run was for a coefficient of friction of 0.65; the object was to determine the magnitude of the force, T, for limiting equilibrium. For interest, Figure 2.12(c) shows a plot from RBM with the coefficient of friction equal to 0.65, but with zero toe force: the blocks are in the process of failing.

2.5.6.1 Run 1, Validation 6

For this run the load, T, was set to zero and the coefficient of friction varied, in order to bracket the value required for stability. Initially, however, the system of blocks was "compacted" by allowing gravity to act for some time along the long dimension of the blocks. Gravity was then rotated to the correct direction for the stability test. For the initial compaction, the mass damping (see DAMPING command in Appendix III) was set to 0.5,0.7,0,1. The damping was then reduced to 0.05, 0.7,0,1 for the subsequent test of stability. Two different friction values were used for the stability test, which consisted of two restart runs, both starting from the same restart file created at the end of the compaction stage. The predicted coefficient of friction for limiting equilibrium was 0.7855. For a value of 0.80 the system of blocks moved slightly and then stabilised, as judged by the velocities reducing to low values and the out-of-balance forces and moments on block centroids tending to zero. However for a friction coefficient of 0.77, the blocks continued moving, with the velocities increasing. It was noted that the mode of failure was slightly different from that predicted by Goodman and Bray, in that only three of the lower blocks were sliding rather than toppling, compared to four blocks predicted by Goodman and Bray.

2.5.6.2 Run 2, Validation 6

For this run the friction coefficient was held constant at 0.65, and the load, T, varied in order to determine the value required for stability. As for Run 1, the various tests were made from a common starting-point, using the restart file created at the end of the compaction phase. The damping parameters for the stability tests were:

$$\begin{cases} \lambda_{min} = 0.25 \\ f_{min} = 0.7, \text{ with the stiffness term set to zero.} \end{cases}$$

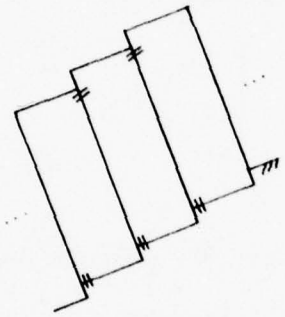
Goodman and Bray predict the critical value of T to be 2013 units, but the system was found to stabilise with T=2000 units. At T = 1900 units the blocks were definitely unstable, and collapse occurred.

2.5.6.3 Conclusions from Validation 6

Run 1 bracketted the coefficient of friction with a range that includes the value predicted by Goodman and Bray. However Run 2 showed that the system of blocks was stable for a lower value of toe force than predicted. Although the difference was quite small, it is worth trying to understand what caused it.

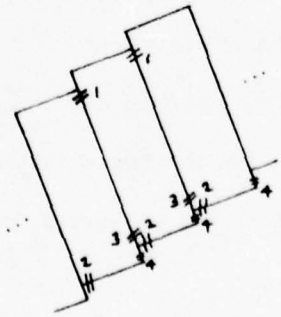
For T = 2000, RBM found a valid system of forces that satisfied equilibrium. Since this corresponds to the requirements for a lower bound solution, the observed force, T, should have been higher than the "true" value for limiting equilibrium, rather than lower. A check was made by

hand that the forces acting on individual blocks did indeed sum to zero. The reason for the discrepancy was traced to the fact that the system of contacts assumed in Goodman and Bray's analysis differed from that developed during the computer run with RBM. For the toppling blocks, Goodman and Bray assumed the contacts to be located as shown below:



// = contact between two blocks.

The contacts observed to have formed at equilibrium in Run 2 were as follows:



The magnitudes of the forces were:

1 and 2 ... high

3 ... quite high

4 ... very low

It was conjectured that the different location of the forces was responsible for the greater apparent stability of the system of blocks. In particular the effect of the two forces acting on the long edge of each block would be equivalent to a single resultant force acting some way down the edge, and not at the top, as assumed by Goodman and Bray. In order to test the conjecture, a program was written that carried out a Goodman and Bray limit equilibrium analysis, but with a variable contact location on the long edges, rather than fixed at the top. A variable, C, which could take values between 0 and 1, specified the location of the contact point as a fraction of the block height. All blocks were affected by the change, with a value of 1.0 for C giving the standard Goodman and Bray solution. However, moving the contact points down gave the following reductions in the force, T, necessary for stability:

C	T	
1.0	2013	(standard result)
0.75	1995	
0.50	1902	

Both the direction and magnitude of the change in T correspond well with the experience with RBM (Run 2) reported earlier. Although this result lends support to the conjecture that changes in contact locations are responsible for the changes in stability, it is not the whole story, since the contact locations can also vary on the lower edges and they can also be different for different blocks.

It is not clear whether the assumptions made by Goodman and Bray are any more realistic than those implicit in the distinct element simulation. Finite rotations must occur before the contact points migrate to the top corners of the toppling blocks, but these cannot occur until the system starts to fail. It would appear that the force necessary to prevent any movement is smaller than the force necessary to restore equilibrium, once a very small movement has occurred.

2.6 CONCLUSIONS AND SUGGESTIONS

Almost a one-for-one translation of the original assembly-language distinct element program into Fortran has been made. The temptation to "improve" the program has been resisted, because the program is intended to serve as a base-line version from which more exotic versions can be developed. Its use of memory could certainly be improved, and perhaps its speed as well; these points are addressed below. However it has been established by several validations that the program works as intended.

It should perhaps be recorded that certain constraints in the original assembly language version are not present in the Fortran version, RBM:

1) Block velocities can assume any value; there is no limiting velocity in RBM.

2) Inertial masses have the correct physical values (recall that all blocks had the same mass in the original program).

3) The moments of inertia are also correct.

4) The timestep in RBM corresponds to real, physical time (in the original program the apparent time-scale was different for different-sized blocks).

The constraints of the original program were due to the fact that variables had to be represented as integers, which limited the range of permissible numbers. Of course RBM does not suffer in this way.

2.6.1 SUGGESTIONS FOR FUTURE MODIFICATION AND DEVELOPMENT

Many of the suggested developments have already been made to the following programs, all of which work on similar principles:

1) SDEM, the program described in Chapter 3;

2) BALL, the program developed by Cundall (1978) for research into the mechanics of granular media;

3) a new program under development to allow the discrete blocks of SDEM to crack.

The suggestions are as follows:

1) Free-format input (as in SDEM).

2) Coordinate transformations built-in, so that any set of problem coordinates could be used.

3) Allow blocks to be created at any time. (see BALL)

4) Use global coordinates for block corners instead of local; this will allow reboxing to be more efficient and foolproof (see SDEM).

5) Store box entries all along block edge, rather than just for corners (see BALL). This will allow local (single block) updates, rather than the present global updates.

6) Eliminate storage of non-essential variables - e.g. edge lengths, block angle, COS , SIN of block, force-sums on block.

7) Allow different properties for different joints and blocks.

8) Incorporate edge-to-edge contact formulation for almost-parallel edges.

9) Incorporate cracking, joint water pressure and dynamic loading.

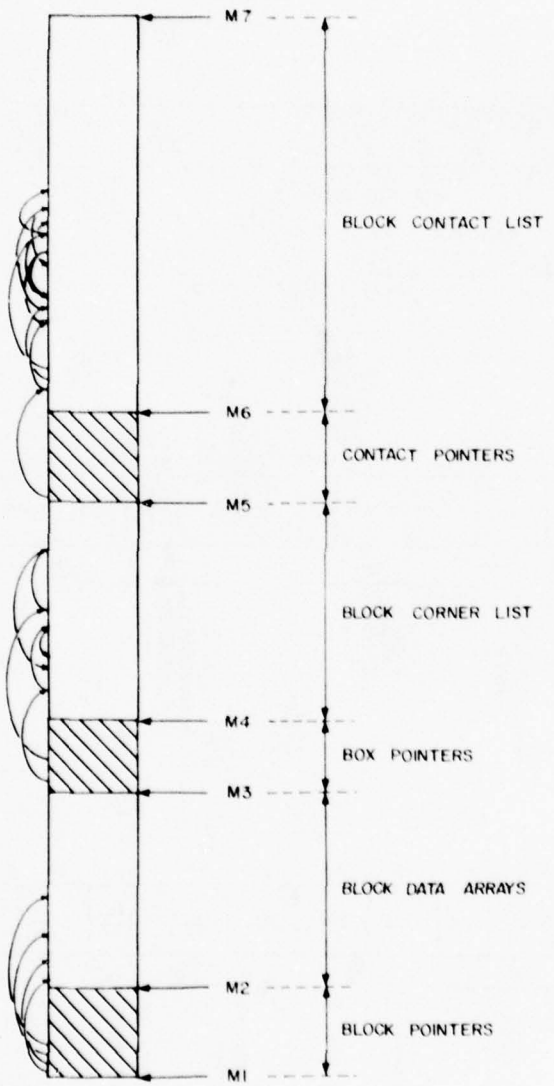


FIGURE 2.1 MEMORY ALLOCATION FOR DATA LISTS

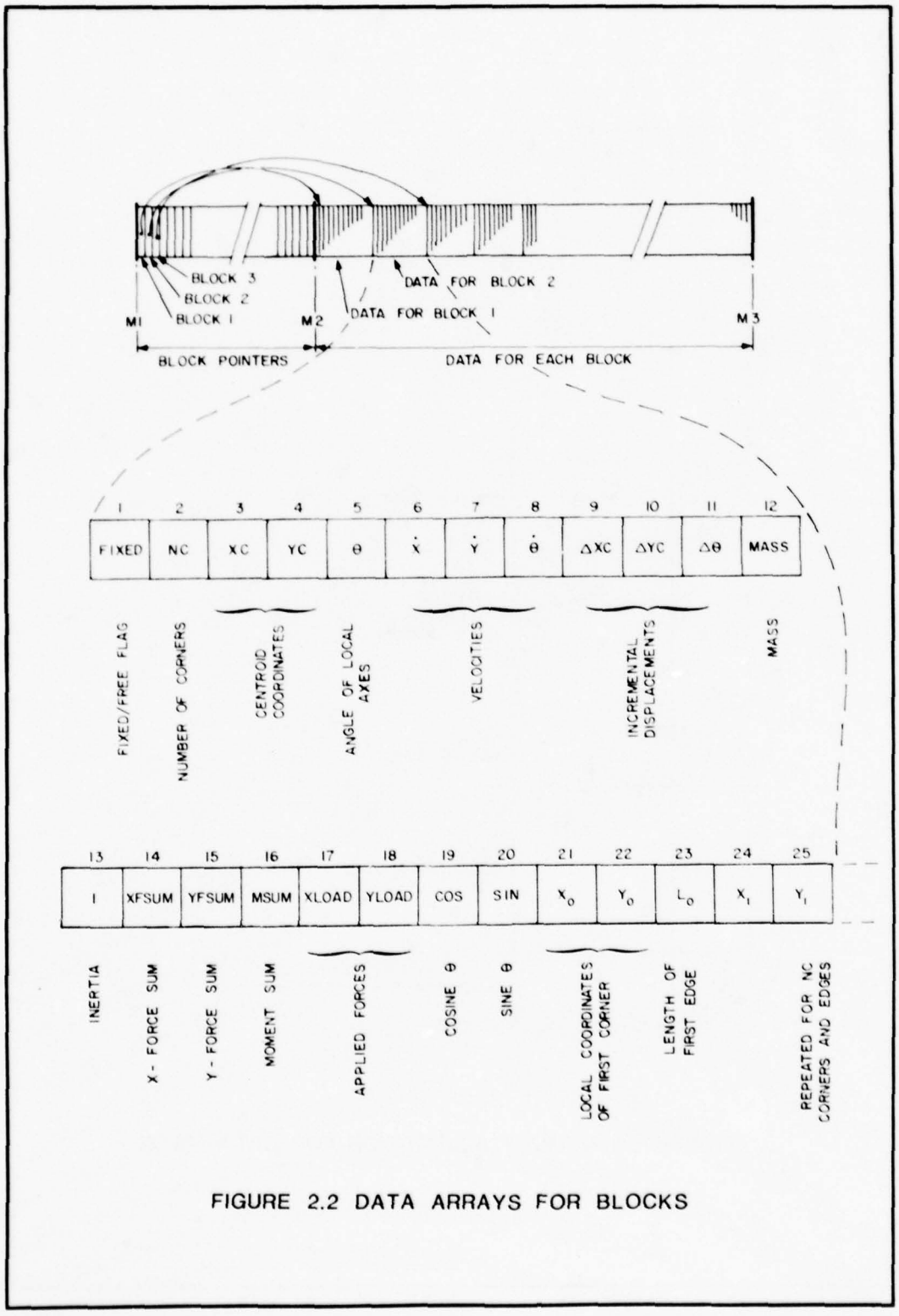


FIGURE 2.2 DATA ARRAYS FOR BLOCKS

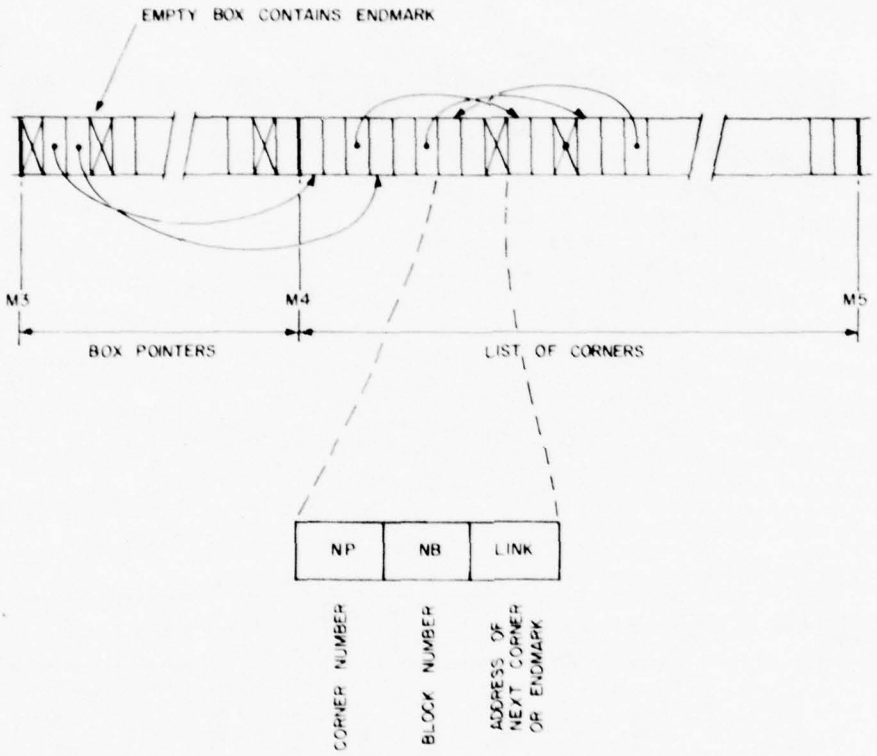


FIGURE 2.3 LINKED LIST OF BLOCK CORNERS BY BOX

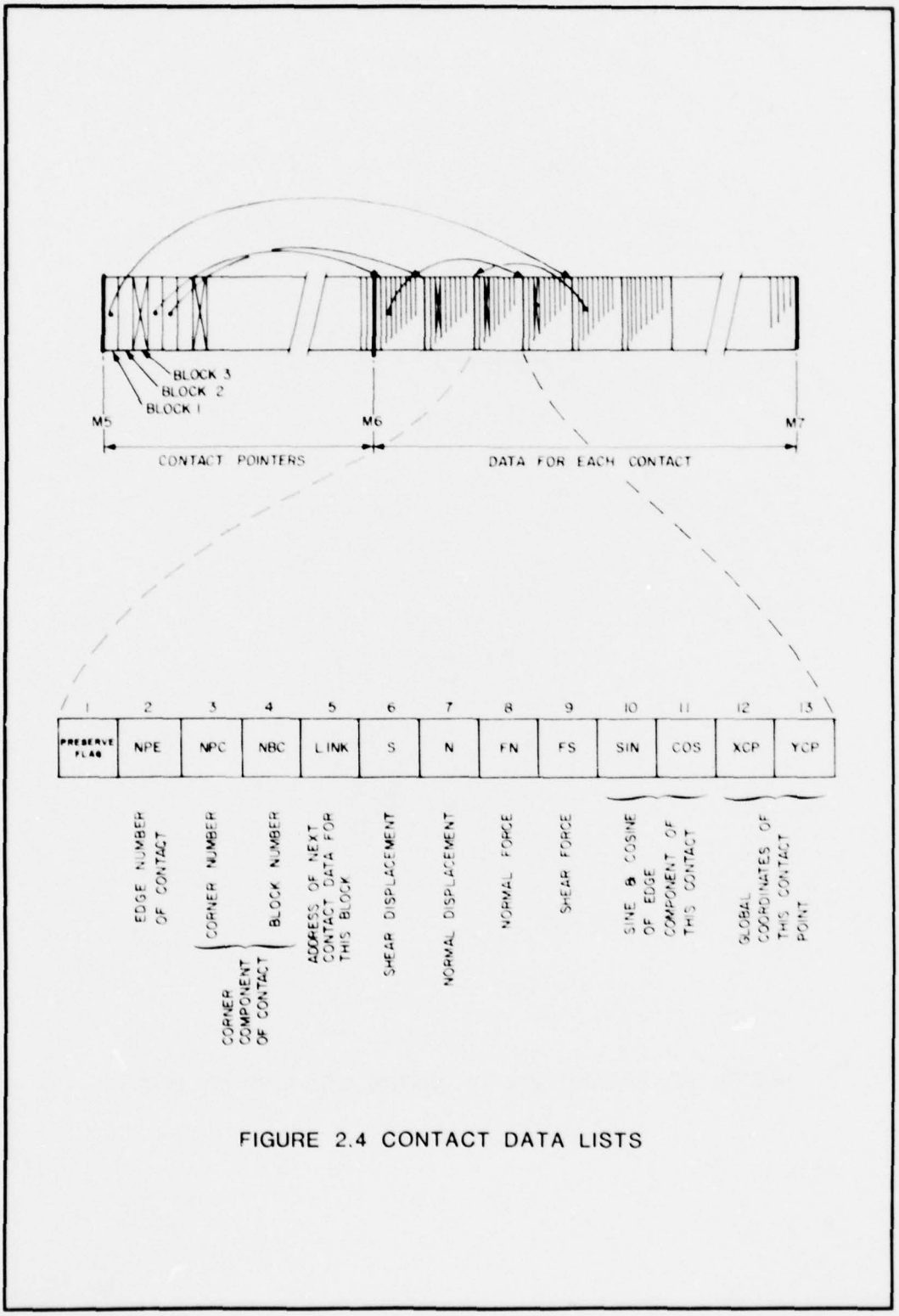


FIGURE 2.4 CONTACT DATA LISTS

The program indicates values of 0.459 and 0.114 respectively.

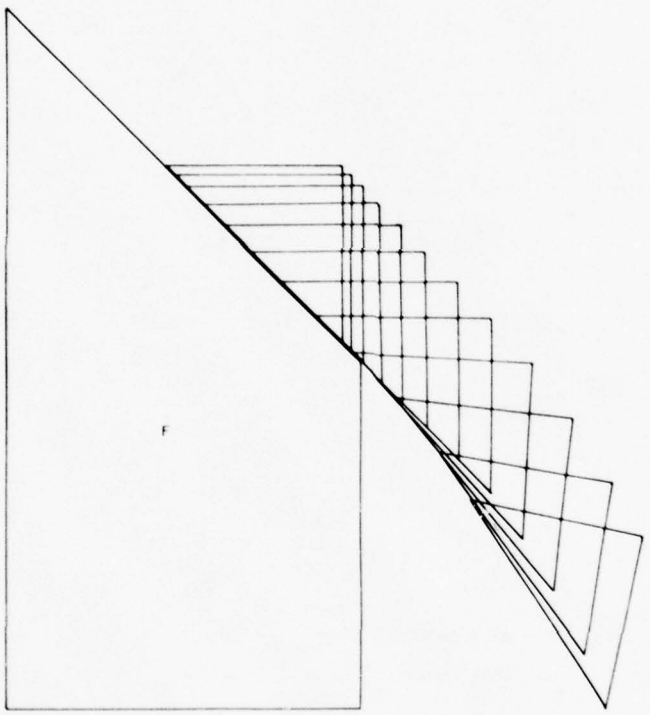
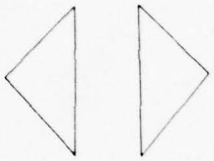


FIG. 2.5 VALIDATION 1 - BLOCK SLIDING AND TOPPLING

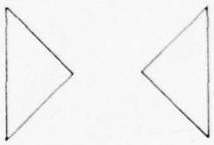
FIRST CASE



INITIAL VELOCITY 1.0→ 0.0

FINAL VELOCITY 0.0 1.0→

SECOND CASE

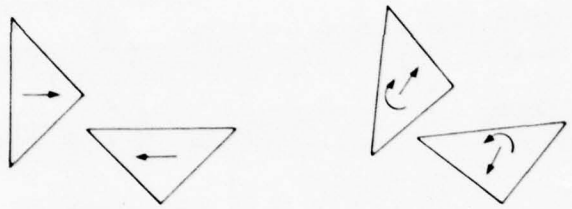


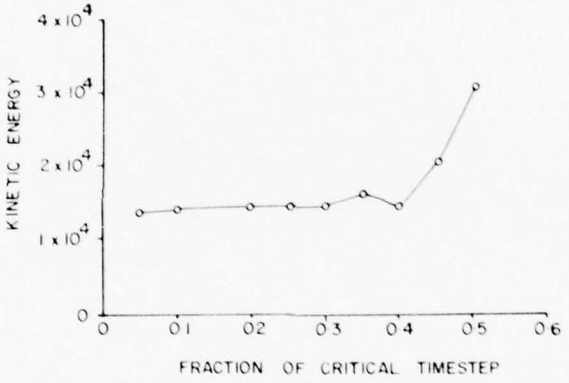
INITIAL VELOCITY 1.0→ 0.0

FINAL VELOCITY 0.0 1.0→

FIG. 2.6 VALIDATION 2 - IMPACT OF TWO BLOCKS

DAMES & MOORE





FRACTION OF CRITICAL TIMESTEP	TOTAL KINETIC ENERGY x 10^4	TOTAL MOMENTUM x10^5
0 50	3 13	9 645
0 45	2 11	9 645
0 40	1 47	9 645
0 35	1 62	9 645
0 30	1 52	9 645
0 25	1 49	9 645
0 20	1 44	9 645
0 10	1 40	9 645
0 05	1 39	9 645
BEFORE IMPACT	1 39	9 645

FIG. 2.7 VALIDATION 3 - EFFECT OF TIMESTEP ON ENERGY AND MOMENTUM CONSERVATION

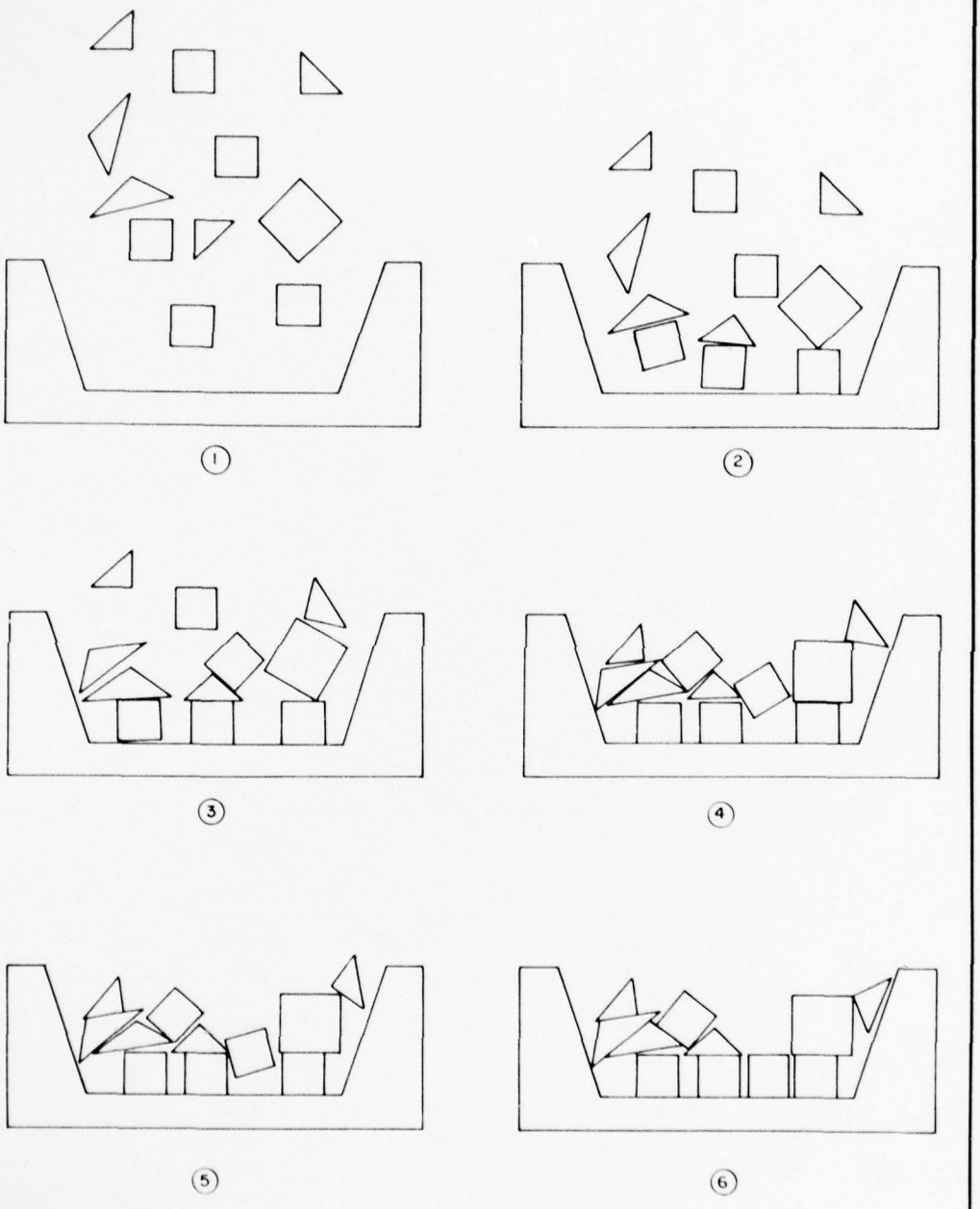
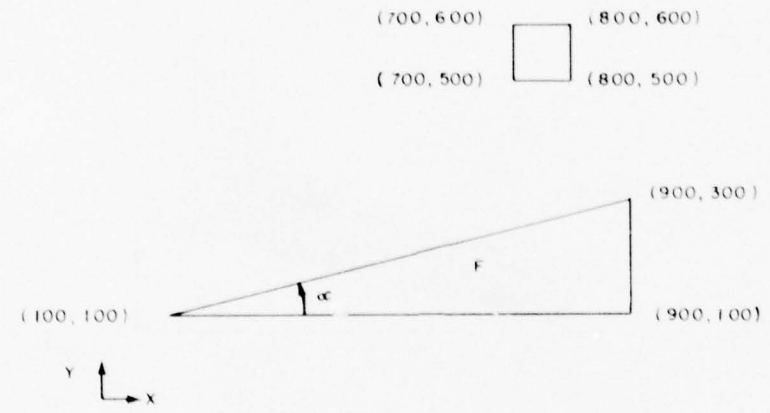


FIG. 2.8 VALIDATION 4 - BLOCKS FALLING INTO A RECEPTACLE



INPUT DATA

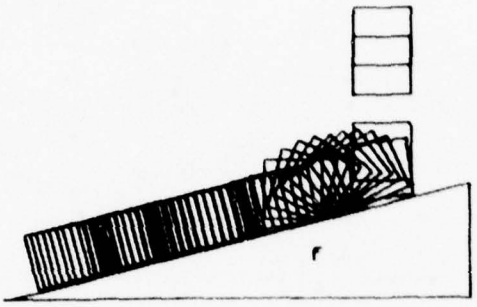
μ	= 0.30	FRICTION
α	= ARCTAN (0.25)	ANGLE OF PLANE
g_y	= -9.81	GRAVITY
ρ	= 1.00	DENSITY
$k_n = k_s$	= 10^7	SHEAR AND NORMAL STIFFNESS

FRACTION = 0.1
λ = 0.5 } RAYLEIGH DAMPING PARAMETERS
f = 5.0 }

CALCULATED TIME STEP $\Delta T = 6.3246 \times 10^{-3}$

FIG. 2.9 VALIDATION 5 - INITIAL GEOMETRY

2.10 (a)



2.10 (b)

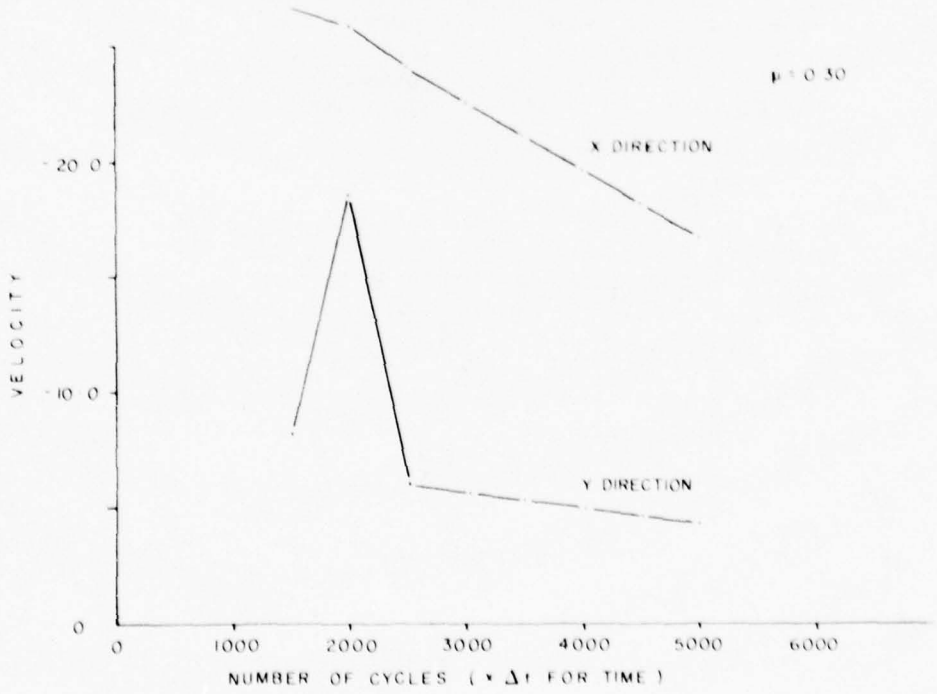
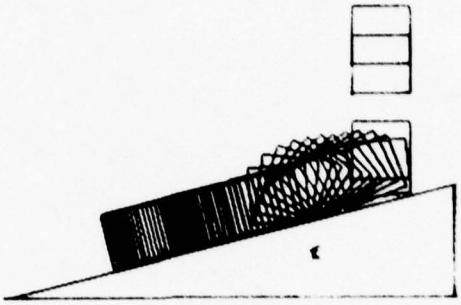


FIG. 2.10 VALIDATION 5 - CASE A

DAMES & MOORE



2-11(a)

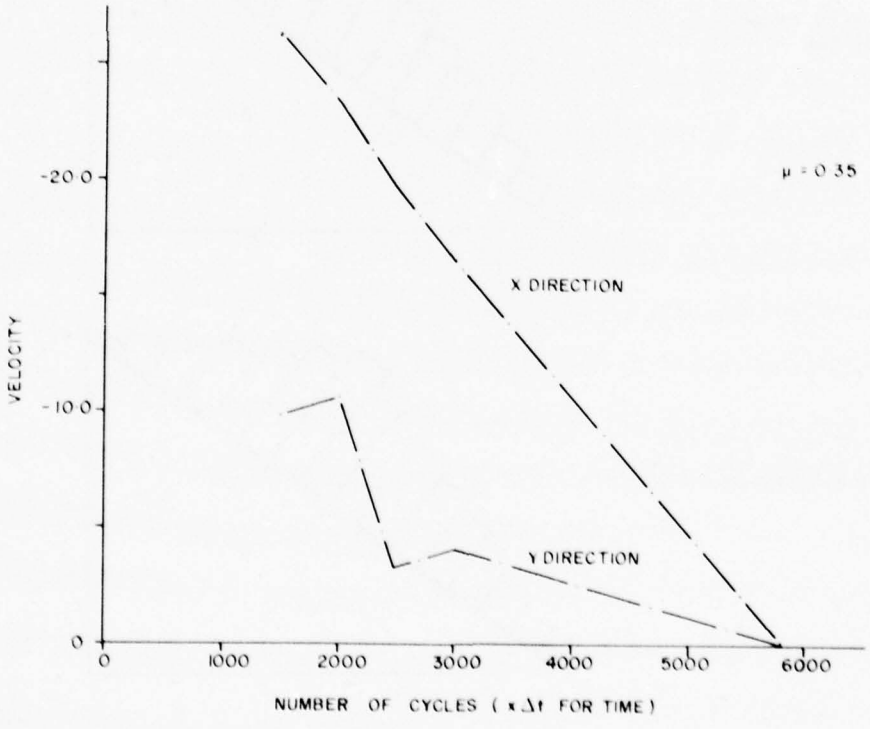
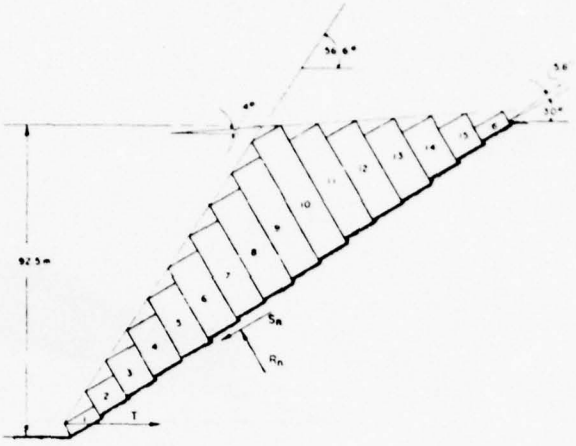


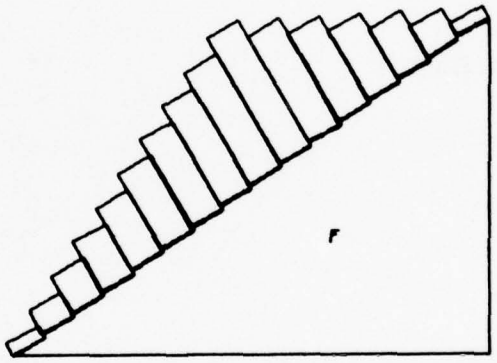
FIG. 2-11 VALIDATION 5 - CASE B

DAMES & MOORE

a) SLOPE CONFIGURATION FROM GOODMAN AND BRAY'S PAPER (1976)



b) INITIAL PLOT FROM PROGRAM RBM



c) "SNAPSHOT" PLOT FROM RBM SHOWING FAILURE MODE FOR A FRICTION COEFFICIENT OF 0.65

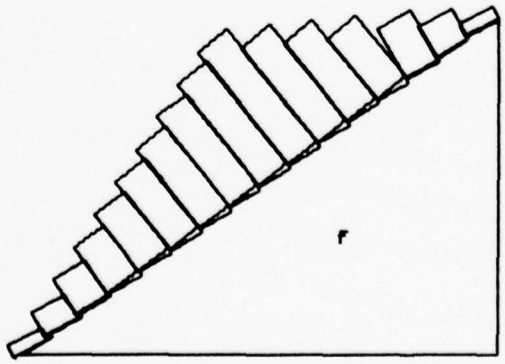


FIG. 2.12 VALIDATION 6 - TOPPLING FAILURE OF A ROCK SLOPE

DAMES & MOORE

CHAPTER 3: SIMPLY-DEFORMABLE PROGRAM, SDEM

3.0 This Chapter describes the introduction of limited deformability into the blocks of the distinct element method, as a simpler alternative to the general approach of Chapter 6. Examples and validations are given.

3.1 INTRODUCTION

The scheme described in this section is an extension of the distinct element method (DEM) in which limited deformability is given to each element. It is an attempt to preserve the simplicity and efficiency of the DEM while extending its usefulness in modelling high stress applications. The new program, SDEM, is intended for representing rock systems where the intact behaviour of the rock, although affecting significantly the mechanics of the system, does not participate strongly enough for the more complex deformation modes to contribute much to the overall deformation.

However, for situations involving blocks of rock with complex shapes, or with complex loading, and which also undergo large deformations of the intact material, there is no alternative but to discretize the block itself in order to provide the many degrees of freedom that can permit the block to deform in an adequately complex manner. This approach is taken with program DBLOCK, which is described in Section 6.

The simplifications embodied in SDEM are concerned only with the number of degrees of freedom associated with block deformations; the constitutive laws can be completely general, and include plasticity and arbitrary non-linearity. It should be appreciated that the approach described in this section has not yet been fully explored, and that evolution will take place as experience is accumulated: the formulation that has been presented should not be taken in any sense as the optimum.

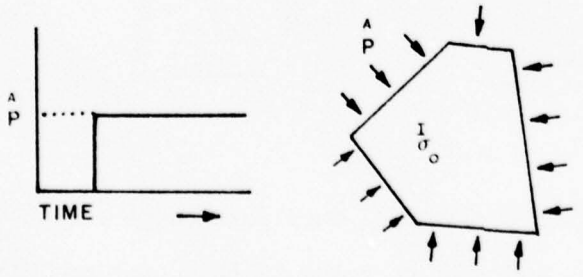
A listing of SDEM is given in Appendix XI. The input commands are almost identical to those of RBM, except for some changes noted in Section 3.5

3.2 SIMPLIFIED EXPLANATION OF DEFORMABLE-BLOCK APPROACH

In order to give the reader a physical picture of what is involved in the new formulation, a descriptive account of the behaviour of a block deforming volumetrically is presented below. The general formulation is given later.

3.2.1 VOLUMETRIC DEFORMATION

Consider an arbitrary block suddenly exposed to a uniform pressure, $\overset{A}{p}$:

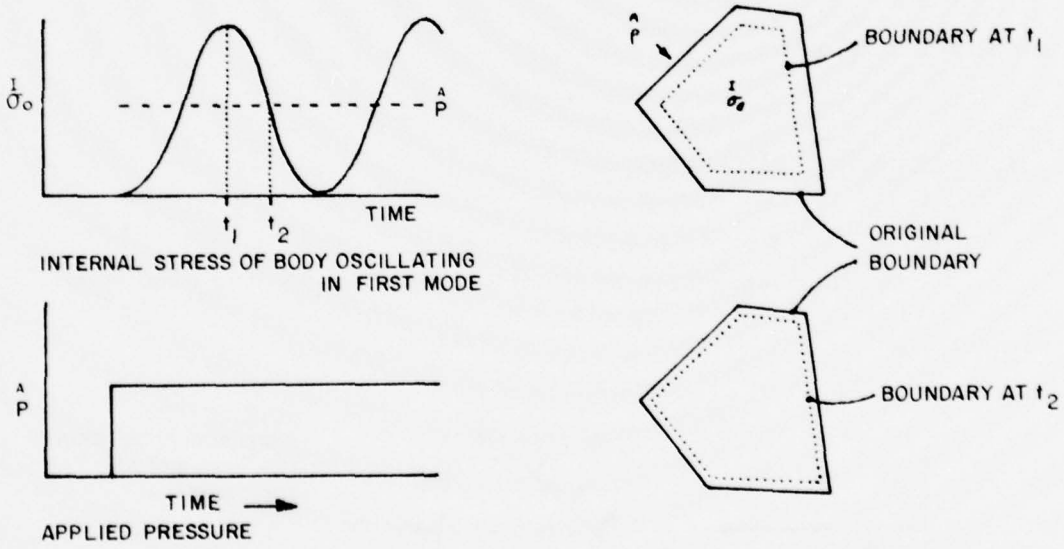


NOTE:
SUPERSCRIPT "A" STANDS FOR "APPLIED" AND "I" FOR "INTERNAL"

Instantaneously, nothing will happen owing to the finite wave speed in the solid. In particular, the isotropic stress in the interior of the solid, $\overset{I}{\sigma}_o$, will be zero at the instant that the pressure is applied, and will build up gradually as the pressure wave reaches the interior of

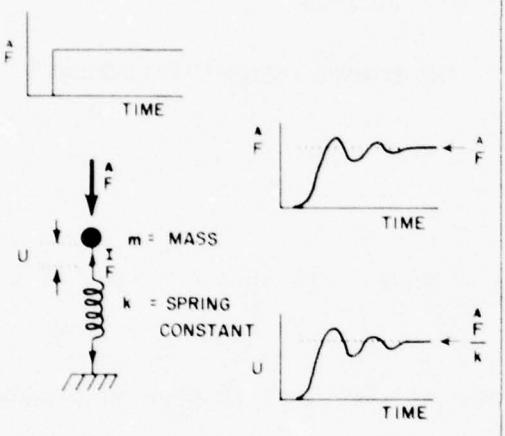
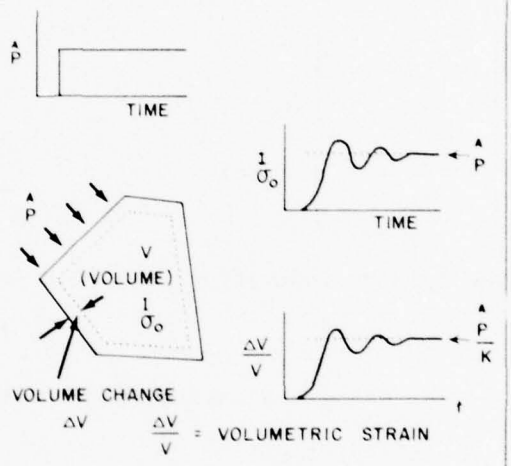
th block. In addition, the boundaries of the block will move inwards towards the centre due to the volumetric compression of the material.

Over a larger time span the internal stress and the boundary displacements will be oscillatory, and will consist of a summation of the many natural modes of oscillation of the body. The predominant mode, however, will be the fundamental, which corresponds to all points in the body moving either inwards or outwards in phase:



The internal isotropic stress will oscillate about a mean value equal to the applied pressure, and will converge to this pressure after internal damping has dissipated all dynamic energy.

The behaviour described above - that of a solid body oscillating in its first volumetric mode - is simply that of a single degree-of-freedom oscillator, such a mass-spring system, being excited by a step force. The analogy between the block and the mass-spring system is as follows:

MASS-SPRING SYSTEM	BLOCK WITH APPLIED PRESSURE
	
$\overset{A}{F}$ APPLIED FORCE	$\overset{A}{p}$ APPLIED PRESSURE
$\overset{I}{F}$ SPRING FORCE	$\overset{I}{\sigma_o}$ INTERNAL ISOTROPIC STRESS
U DISPLACEMENT	$\frac{\Delta V}{V} = \epsilon_v$ VOLUMETRIC STRAIN
$\dot{U}$ VELOCITY	$\dot{\epsilon}_v$ VOLUMETRIC STRAIN-RATE
m MASS	m^e EFFECTIVE MASS
k SPRING CONSTANT	K BULK MODULUS
$U^{EQUIL} = \frac{\overset{A}{F}}{K}$ EQUILIBRIUM DISPLACEMENT	$\epsilon_v^{EQUIL} = \frac{\overset{A}{p}}{K}$ EQUILIBRIUM VOLUMETRIC STRAIN
$\omega_o = \sqrt{\frac{k}{m}}$ NATURAL ANGULAR FREQUENCY	$\omega_o = \sqrt{\frac{K}{m^e}}$ NATURAL ANGULAR FREQUENCY

NOTE: HERE, m^e IS THE "MASS" NECESSARY FOR THE NATURAL FREQUENCY ω_o TO BE EQUAL TO THAT OF THE FIRST VOLUMETRIC MODE. ITS EVALUATION WILL BE TREATED IN APPENDIX V.

The central-difference algorithm for computing the response of the mass-spring system is as follows:

$$\dot{u} := \dot{u} + \frac{(\overset{A}{F} - \overset{I}{F})\,\Delta t}{m}$$

$$\overset{I}{F} := \overset{I}{F} + k\dot{u}\Delta t$$

Δt = TIME STEP

THE SYMBOL := MEANS "REPLACED BY"

Here, F is an independent variable and is prescribed as a function of time.

Using our analogy, the response of the block to applied pressure takes the same form:

$$\dot{\epsilon}_v := \dot{\epsilon}_v + \frac{(\overset{A}{P} - \overset{I}{\sigma}_o)\,\Delta t}{m^e}$$

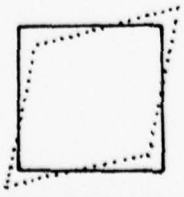
(LAW OF MOTION FOR BLOCK STRAIN)

$$\overset{I}{\sigma}_o := \overset{I}{\sigma}_o + K\dot{\epsilon}_v\,\Delta t$$

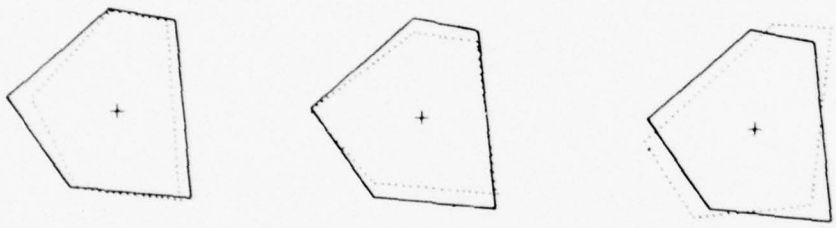
(CONSTITUTIVE LAW FOR BLOCK)

3.2.2 Other Modes of Deformation

Analogies similar to that described above can be made for other simple modes of deformation, such as a shear mode:



In general, any number of independent modes of deformation (oscillation) may be taken. However, in the program three modes are used, corresponding to the three strains in two dimensions i.e. ε_{11}, ε_{22}, $\varepsilon_{12} + \varepsilon_{21}$.* These modes are visualised as follows:



ε_{11} ε_{22} $\varepsilon_{12} + \varepsilon_{21}$

In addition to these modes there remain of course the three rigid-body modes considered by the original block program (two translational and one rotational degrees of freedom), making six degrees of freedom per block in total.

3.2.3 Coupling of Blocks

The degrees of freedom described above are associated with each block individually. Blocks are coupled together via the joint stiffnesses, whose formulation is unchanged from that of the original rigid block program. However, there now are two additional stages in the calculation:

*In this section, ε_{ij} is taken as the partial derivative of the i displacement with the j co-ordinate.

3.2.3.1 Applied Stress Derived from Boundary Forces

The average stress in a closed volume can be derived from the forces acting on the boundary of the volume, using Gauss' divergence theorem. In this way the "applied stress" on the block is evaluated from the set of forces developed by the contacts around the block. Thus:

$$\overset{A}{\sigma}_{ij} = \frac{\sum^{c} \overset{c}{F}_i \overset{c}{X}_j}{V}$$

in tensor notation, where

$\sum^{c}$ represents the sum over all contacts, and

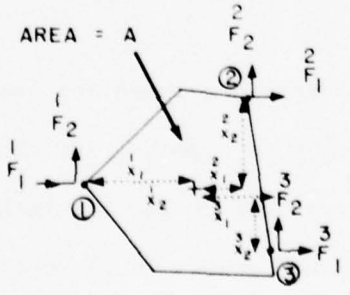
V is the block volume (= area in 2 dimensions)

$\overset{c}{F}_i$ are the forces at contact c

$\overset{c}{X}_j$ are the co-ordinates of contact c relative to the centroid

For example, the applied pressure $\overset{A}{p}$ used in 3.2.1 is given by:

$$\overset{A}{p} = \frac{1}{2}\left(\overset{A}{\sigma}_{11} + \overset{A}{\sigma}_{22}\right) = \frac{\sum^{c} \overset{c}{F}_1 \overset{c}{X}_1 + \sum^{c} \overset{c}{F}_2 \overset{c}{X}_2}{2A}$$

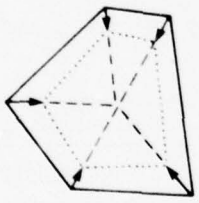


$$\overset{A}{p} = \frac{\overset{1}{F}_1 \overset{1}{X}_1 + \overset{2}{F}_1 \overset{2}{X}_1 + \overset{3}{F}_1 \overset{3}{X}_1 + \overset{1}{F}_2 \overset{1}{X}_2 + \overset{2}{F}_2 \overset{2}{X}_2 + \overset{3}{F}_2 \overset{3}{X}_2}{2A}$$

3.2.3.2 Boundary Displacements Derived from Strains

In the original block program, the displacement increments at block boundaries were evaluated in order to give normal and shear increments of displacement at contacts, which would then allow contact forces to be found. The boundary displacements were derived from the known centroid displacements and rotation.

For deformable blocks, the displacements at a block boundary are also functions of the strains in the block. For example, in the case of a volumetric compression, at each boundary point the displacement vector is directed inwards towards the centroid, with a magnitude proportional to the distance from the centroid:



BOUNDARY DISPLACEMENTS DUE TO VOLUMETRIC COMPRESSION

In general, for strain increments referred to orthogonal axes, the expression for boundary displacement increments is:

$$\Delta u_i^c = (\Delta \epsilon_{ij} + \Delta R_{ij}) \, x_j^c + \Delta u_i$$

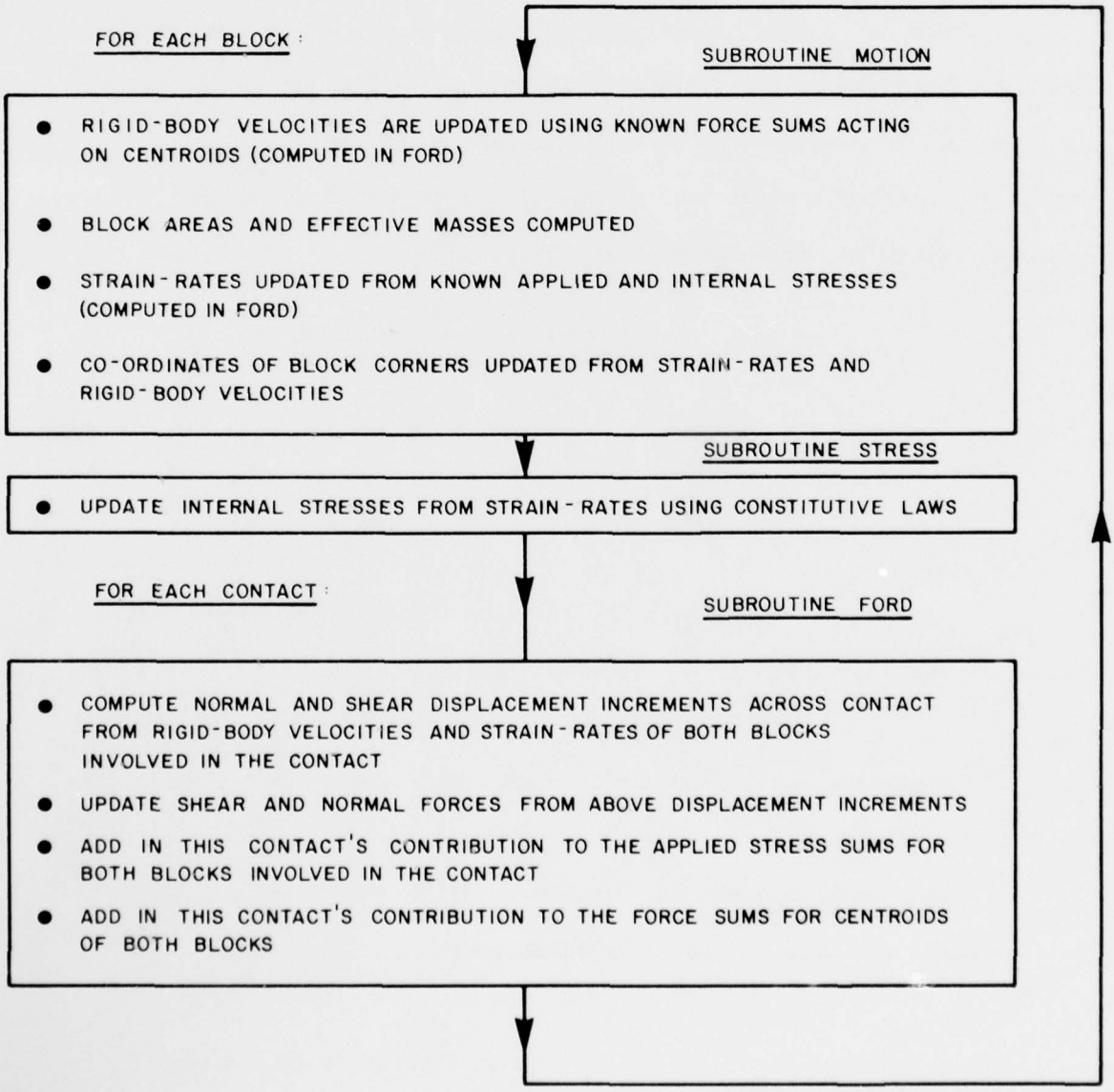
Here, the superscript "c" stands for "contact" or "corner".

The terms ΔR_{ij} and Δu_i are the rotation and rigid-body translations that were used in the rigid-block program.

x_j^c are the co-ordinates of contact c relative to the centroid.

3.3 SEQUENCE OF OPERATIONS

Before giving the mathematical treatment of the calculation method, it may be useful to present a flow diagram in order to get an overall picture of what happens in the program:



3.4 MATHEMATICAL TREATMENT

The equations corresponding to the flow diagram given in Section 3.3 are presented below. Tensor notation is used for subscripts i,j, and k only, where repeated indices imply summation over the index.

Indices:

- i, j, k — These have values 1,2 and refer to orthogonal coordinates.
- c — Contact or corner number.
- R — means "relative"
- P, E — These superscripts are intended to distinguish between the two blocks participating in a contact. "P" refers to the block contributing the corner, and "E" refers to the block contributing the edge to the edge/corner contact.

Symbols:

- x_i — Centroid coordinates of block.
- $\dot{x}_i$ — Centroid velocities of block.
- ΔX_i — Incremental shear and normal displacements at a contact.
- P_i — Shear and normal forces at a contact.
- Q_i — Global contact forces.
- $\dot{\theta}$ — Angular velocity of block (anticlockwise positive).
- $\dot{R}_{ij}$ — Skew-symmetric tensor equal to $\begin{bmatrix} 0 & \dot{\theta} \\ -\dot{\theta} & 0 \end{bmatrix}$
- ℓ_i — Maximum block widths.
- $\dot{\varepsilon}_{ij}$ — Strain rate for a block.
- $\overset{A}{\sigma}_{ij}$ — Applied stress on a block.
- $\overset{c}{x}_i$ — Global coordinates of contact point or block corner.
- $\overset{I}{\sigma}_{ij}$ — Internal stress in a block

$\dot{x}_i^c$ Velocities of contact point or block corner.

F_i Force sums on block centroid.

M Moment sum on block centroid.

$a = 1 - \frac{\alpha \Delta t}{2}$

$b = 1/\left(1 + \frac{\alpha \Delta t}{2} \right)$

$c = \beta / \Delta t$, where α, β are the Rayleigh damping parameters.

a^e, b^e Same as a and b, but with α defined for internal block damping.

g_i Acceleration due to gravity.

I Moment of inertia for block.

m Block mass.

A Block area.

$m^e_{(i)}$ Effective block mass for strain-rate calculations (see Appendix V).

K Bulk modulus for block material.

G Shear modulus for block material.

μ Friction coefficient for contacts.

k_{ij} Contact stiffnesses, equal to $\begin{bmatrix} k_s & 0 \\ 0 & k_n \end{bmatrix}$

where k_s = shear stiffness;

k_n = normal stiffness.

δ_{ij} Kronecker delta = $\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

e_{ij} A tensor

$$= \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$$

:= means "replaced by"

(i) Subscripts in parentheses indicate non-tensorial character.

The following calculations are done for all blocks (subroutines MOTION & STRESS):

Rigid-body velocities:

$$\dot{x}_i := (a\dot{x}_i + (\frac{F_i}{m} + g_i)\Delta t)b$$

$$\dot{\theta} := (a\dot{\theta} + \frac{M}{I}\Delta t)b$$

$$F_i = 0$$

$$M = 0$$

Block areas and effective masses:

$$A = \tfrac{1}{2}\sum_{c=1}^{c=n-1}\left\{(x_1^c + x_1^{c+1})(x_2^{c+1} - x_2^c)\right\} + \tfrac{1}{2}(x_1^n + x_1^1)(x_2^1 - x_2^n)$$

(n = number of corners for the block)

$$\ell_i = \max_{c=1,n}(x_i^c) - \min_{c=1,n}(x_i^c)$$

$$m^e_{(i)} = \frac{m(\ell_i)^2}{4A}$$

Strain-rates:

$$\dot{\varepsilon}_{ij} := (a^e\dot{\varepsilon}_{ij} + \frac{\overset{\wedge}{\sigma}_{ij} - \overset{I}{\sigma}_{ij}}{m^e_{(j)}}\Delta t)b^e$$

$$\overset{\wedge}{\sigma}_{ij} = 0$$

Corner displacements:

$$\dot{x}_i^c = \dot{x}_i + (\dot{\varepsilon}_{ij} + \dot{R}_{ij})(x_j^c - x_j)$$

$$x_i^c := x_i^c + \dot{x}_i^c\Delta t$$

Rigid-body displacements:

$$x_i := x_i + \dot{x}_i\Delta t$$

Internal stresses:

$$\overset{I}{\sigma}_{ij} := \overset{1}{\sigma}_{ij} + ((K-\tfrac{2}{3}G)\dot{\varepsilon}_{kk}\delta_{ij} + 2G\dot{\varepsilon}_{ij} + \dot{r}_{ij})\Delta t,$$

where $\dot{r}_{ij}$ are the stress rotation correction terms derived in Appendix VI, and are given by:

$$\dot{r}_{ij} = \Delta\sigma_{ij}/\Delta t = -(\sigma_{kj}\dot{R}_{ik} + \sigma_{ik}\dot{R}_{jk})$$

The following calculations are done for all contacts (subroutine FORD):

Velocities of corner relative to edge at a contact:

$$\dot{x}_i^R = \dot{x}_i^P - \dot{x}_i^E + (\dot{\varepsilon}_{ij}^P + \dot{R}_{ij}^P)(\overset{C}{x}_j - x_j^P) - (\dot{\varepsilon}_{ij}^E + \dot{R}_{ij}^E)(\overset{C}{x}_j - x_j^E)$$

Incremental shear and normal displacements:

$$\Delta X_i = J_{ij} \dot{x}_j^R \Delta t \quad ,$$

where $J_{ij} = \begin{bmatrix} \cos\gamma & \sin\gamma \\ -\sin\gamma & \cos\gamma \end{bmatrix}$

and γ = angle between x_1 and X_1 axes.

Shear and normal forces:

$$\Delta P_i = k_{ij} \Delta X_j$$

$$P_i := P_i + \Delta P_i$$

$$D_i = c \Delta P_i$$

If $P_2 > 0$ ⟶ $P_i = 0$

Otherwise $D_i = 0$

If $|P_1| > \mu |P_2|$ ⟶ $P_1 = \mu P_2 \text{ sign } (P_1)$

Otherwise $D_i = 0$

Global contact forces:

$$Q_i = H_{ij} (P_j + D_j) \quad , \text{ where } H_{ij} J_{jk} = \delta_{ik}$$

Contributions to centroid force sums:

$$F_i^P := F_i^P - Q_i$$

$$M^P := M^P - Q_j (\overset{C}{x}_i - x_i^P) e_{ij}$$

$$F_i^E := F_i^E + Q_i$$

$$M^E := M^E + Q_j (\overset{C}{x}_i - x_i^E) e_{ij}$$

Contributions to applied stresses in blocks:

$$\overset{A}{\sigma}{}_{ij}^P := \overset{A}{\sigma}{}_{ij}^P - Q_i (\overset{C}{x}_j - x_j^P) / A^P$$

$$\overset{A}{\sigma}{}_{ij}^E := \overset{A}{\sigma}{}_{ij}^E + Q_i (\overset{C}{x}_j - x_j^E) / A^E$$

Notes: 1) The computer program departs slightly from the formulation given above, in that:

a) the normal force, P_2, is stored with the opposite sign convention from that implied above - i.e. it is taken as positive in compression;

b) the applied stresses are stored in the block data arrays as $A\sigma_{ij}$, not σ_{ij}. The division by A is done once only in MOTION, and not for each contribution in FORD, as indicated above.

2) At present, two shear stresses (σ_{12} and σ_{21}) and two shear strain-rates ($\dot{\varepsilon}_{12}$ and ε_{21}) are calculated, as well as the moment and rotation for a block. In other words the program is operating as if there were seven degrees of freedom per block, whereas there are only six. Although this redundancy apparently causes no problems, it is unnecessary and should be removed in future versions.

3.5 CHANGES TO PROGRAM RBM

The following changes were made to the baseline program RBM (described in Section 2.0) in order to introduce simple deformability into the blocks:

1. The structure of the data lists for blocks was changed, but the system of pointers shown in Figure 2.1 was retained. The new data list is given in Figure 3.1. A new variable, NVARB, was introduced to facilitate future changes to the block data list. NVARB equals the number of words in the block data list, excluding the corner and length data. It is currently set at 24. Changes were made to all routines that contained references to the length of the block data list.

2. The corner co-ordinates stored in the block data lists were altered from local co-ordinates to global co-ordinates. Hence the corner co-ordinates of each block must be updated at each cycle. As a result, functions GLX and GLY were eliminated. Subroutines NEXT, MOTION, REBOX, UPDAT and BPLOT were modified to reflect the change from local to global co-ordinates.

3. Subroutine REBOX was modified to enable it to operate on individual corners rather than on whole blocks. This was

made possible because the global co-ordinates of each corner are known at each cycle, so that re-boxing can be triggered as each corner crosses an integer boundary. The calling sequence for REBOX was moved from CYCLE to MOTION to reflect the different philosophy.

4. Subroutines FORD and MOTION were changed to incorporate the deformable-block calculations described in Section 3.4. New listings of the complete program are given in Appendix XIII.

5. A new subroutine STRESS was added. This routine updates internal stresses from strain rates. The constitutive law is utilized in this routine, but at present elasticity only is used.

6. Several changes were made in NEXT to enable parameters pertaining to block deformability to be specified. The following commands were added:

ELASTIC	K,G	Defines bulk modulus, K and shear modulus, G for rock.
BDAMPING	DAMP, FREQ	Defines internal damping parameters for blocks.
LOCK ON OFF		LOCK ON forces joints to remain in contact once they have touched; i.e., infinite tensile strength is implied. However, sliding can still occur. LOCK OFF returns the behaviour to normal. This is the default setting.

The commands CHECK and LOAD are withdrawn temporarily. Other commands are identical to those of RBM (see Appendix III).

7. A free-format input routine, PARSE, was incorporated in the program. This enables command parameters to be entered in any columns, separated by blanks or commas. The command word can be any length greater than or equal to four characters. Subroutine NEXT was modified to call PARSE.

3.6 EXAMPLES AND VALIDATIONS

3.6.1 EXAMPLE

Figure 3.2 shows that the static deformations of a block resting on the edge of a table, with different values of various parameters. The block was elastic with small elastic moduli, in order to demonstrate the capability of the program to treat large deformations.

3.6.2 RIGID-BLOCK COMPARISON

Comparison runs were made of a block falling on a plane, both with the program RBM (original program with rigid blocks) and with the new program SDEM with very high elastic moduli. The results were almost identical, which is to be expected, since the formulations become identical

when the strain-rates tend to zero. Figure 3.3 shows the block falling on the plane for the two cases of high modulus and low modulus. It is interesting to note that the block slows down more in the low modulus case, presumably because more energy goes into internal vibration, which is then dissipated by the internal damping.

3.6.3 CONTINUUM VALIDATION

In the limit of very high joint stiffness an assemblage of blocks should resemble a continuum, both statically and dynamically. The column of blocks shown in Figure 3.4 was used to check this aspect.

In all runs, the "LOCK ON" option was used, in which all contacts, once created, were locked in the normal direction, but with slip possible in the shear direction when permitted by the value of friction. Three cases were treated: unconfined column and confined column in compression, and column in shear. The column was loaded by applying gravity either in the x or y directions. For the dynamic cases, the mass damping and internal damping were zero, with the joint stiffness-proportional damping as follows:

fraction of critical = 0.1

frequency = 10.0

For the static cases, the stiffness-proportional term was set to zero, and the mass damping was as follows:

Mass damping { fraction of critical = 1.0; frequency = 0.03

Internal damping { fraction of critical = 0.5; frequency = 0.25

The case of confined compression was modelled by setting a high value of joint friction, which prevents lateral deformation of the blocks. For unconfined compression zero friction was used. Other properties were as follows:

Joint stiffnesses: $k_n = 2.0 \times 10^7$

$k_s = 2.0 \times 10^7$

Material properties:

Bulk modulus: $K = 2.0 \times 10^4$ } for
Shear modulus: $G = 0.428572 \times 10^4$ } compression tests
(Poisson's ratio = 0.4)

Bulk modulus: $K = 1.0 \times 10^4$ } for
Shear modulus: $G = 1.0 \times 10^4$ } shear test

Density: $\rho = 1.0$

Applied gravity: $g_y = -1.0$, for compression tests
$g_x = 0.1$, for shear test

Column height: $L = 800$

Column width: $W = 100$

Number of blocks: $n = 8$

The moduli appropriate to the various modes of deformation were as follows:

confined compression	unconfined compression	shear
$K + \frac{4}{3} G$	$\frac{4G(\frac{1}{3}G + K)}{(K + \frac{4}{3} G)}$ (plane strain Young's modulus)	G
2.5714×10^4	1.4286×10^4	1.0×10^4

In the results that follow, the theoretical values for natural period of oscillation and static deflection were calculated as follows:

Natural period, $T = 4L\sqrt{\frac{\rho}{E^*}}$

Static displacement of top block centroid $\delta = \frac{- g\rho\ (L^2 - x^2)}{2E^*}$

where E^* is the appropriate modulus selected from the above table

x is the distance from the top of the column to the top block centroid (50 units, in this case)

RESULTS		mode: confined compression	unconfined compression	shear
DYNAMIC	theoretical	20.0	26.77	32.0
Period, T	measured	19.8	24.4	32.1
STATIC	theoretical	12.4	22.3	
displacement δ	measured	12.4	21.0	not performed

Notes: (1) The period T was taken over only one or two cycles of oscillation, and was thus subject to some error.

(2) The ratio of σ_{11} to σ_{22} in the confined static case was also measured, and conformed well with the theoretical value.

3.6.4 PLASTIC FLOW DEMONSTRATION

A run was made to illustrate the unique capabilities of program SDEM to model slip on the discontinuities of a complex joint pattern simultaneously with large-strain plastic flow in the intact material. This type of simulation would probably be quite difficult to do with existing finite element or finite difference programs. SDEM handles it quickly and efficiently.

For the purpose of the run, a simple Von Mises yield law and associated flow rule was inserted into subroutine STRESS. The formulation was similar to that presented by Wilkins (1969).

Figure 3.5, frame 1, shows the initial geometry of the rock blocks. The triangular blocks were 100 units on the side with density of 1.0 unit. The "missile" was much heavier, with a density of 10 units, and was given a y-velocity of -40 units. Joint stiffnesses were 10^7 units and bulk and shear moduli 10^7 units also. The yield stress was set at 10^2 units, and joint friction coefficient 0.2. Frames 2 to 10 show "snapshots" of the

system every 100 cycles after impact. It should be noted that the two block corners of initial impact were initially set very slightly apart, in order to allow entry for the missile.

From the sequence of frames, large plastic strains can be observed in some of the blocks, together with gross joint displacements. Although it may not be obvious from the pictures, the missile is being simultaneously deflected to the right, and slowed down.

Although only a simple plasticity law was used in the example, more complex formulations with non-associated flow rules can be inserted into the program, literally in a matter of minutes. Such formulations are currently being used extensively in continuum finite difference programs such as DAMSEL (Cundall, 1976).

For interest, Figure 3.6 shows an identical run as Figure 3.5, but performed with the rigid block program RBM. All physical constants are the same (except of course the intact elastic and plastic properties) and the frames are plotted at the same time intervals. It can be seen that the missile is slowed down almost instantaneously to a very low velocity. The momentum is transferred to the blocks, particularly the lower row, which is propelled downwards at high speed. The simulation was stopped after Frame 7, since one of the lower blocks had reached the limit of the computation field (box area), as evidenced by the "fix" flag that has been attached to it. It should be noted that line 51 of routine UPDAT (see Appendix XII)

was removed for the purposes of this run. This statement deletes the contact if the normal displacement exceeds 3 units, which is the case for the present simulation, where the transient forces are very high.

3.6.5 COMPARISON OF PROGRAM SPEEDS

It is difficult to compare the speed of the new program SDEM with the old, rigid-block program RBM, since the run times depend on how many contacts exist, how large the blocks are compared to the box area, and how many updates are done during the run. As a rough check, both programs were run on the Goodman and Bray validation problem described in Section 2.5, using 300 time steps. During this time RBM did 13 updates and SDEM did 21, due to the higher deformability and the different update logic. The ratio of executing times (excluding problem setup and printout) was 1 to 1.6.

3.7 CONCLUSIONS AND RECOMMENDATIONS

A means has been devised to give the blocks of the distinct element method the freedom to deform in simple ways. Although the tests performed on the new program have been quite limited so far, the results look promising. The formulation allows large displacements within blocks, so that large plastic flows of intact material could be modelled together with slip on complex joint sets. No change has been made to the scheme of edge-to-corner contacts for representing joints between blocks although the scheme described in Chapter 5 could be used. The new program SDEM was found to be slower than RBM by a ratio of 1.6 to 1.

The new program, SDEM, can be regarded as filling the gap between the original distinct element method and the general lagrangian scheme (with re-zoning of sliding contacts) that has been studied in Chapter 6.0. There are definite limitations to program SDEM that the general program will not have. For example, there seems to be no obvious way to make SDEM represent a true continuum without joints. It is true that, in the limit of very stiff joints between blocks, the behaviour tends to that of a continuum, but this is achieved only at the expense of considerable computer time, since the time step is forced to be very small due to the joints. Furthermore, the approximation to a continuum is only good for triangular blocks; in this case the formulation is similar to that of constant strain finite elements or finite difference zones. As the number of edges increases beyond three, the blocks look "stiffer" because they still only have three degrees of freedom to deform even though their large number of edges implies that they should have a correspondingly large number of degrees of freedom.

A useful future development could be to arrange for blocks to have the same number of degrees of freedom as the number of edges. In other words, a greater number of stress and strain terms would be stored for a block with many edges than for a block with few edges. The additional stress and strain terms could be derived from the differentials of stress and strain with respect to the spacial co-ordinates. Although this proposed formulation would treat quasi-static problems more realistically, it is not clear whether it would be good for problems where wave propagation was important.

This section presents only the initial attempt at representing block deformability. No doubt as the topic is explored further, better ideas will appear.

3.8 ACKNOWLEDGEMENT

The motivation for developing the new method came from the meeting of September 20, 1977 at DNA, where Ivan Sandler, Russell Duff and others expressed their concern that the simplicity and efficiency of the distinct element program would be lost with the general deformable block program, and that perhaps a middle line could be taken in which simple deformability would be added to the original procedure.

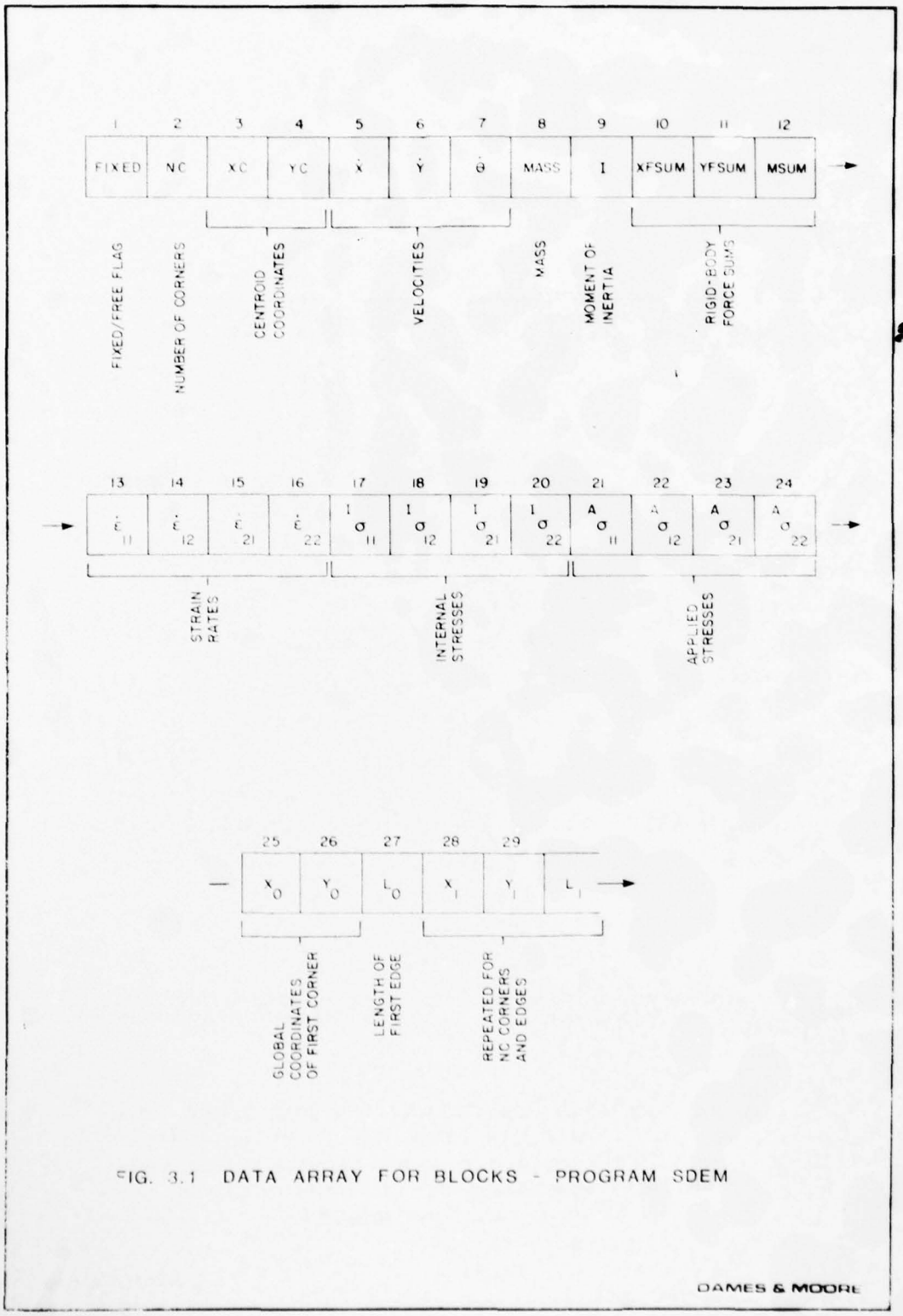
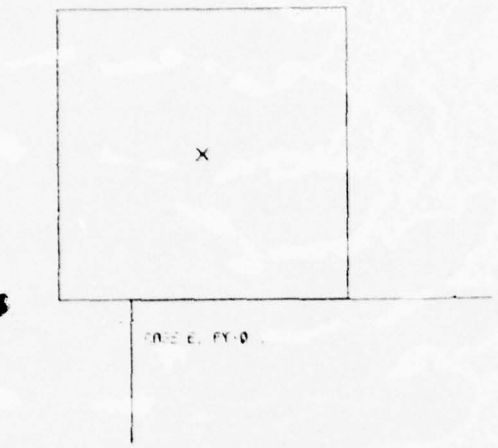
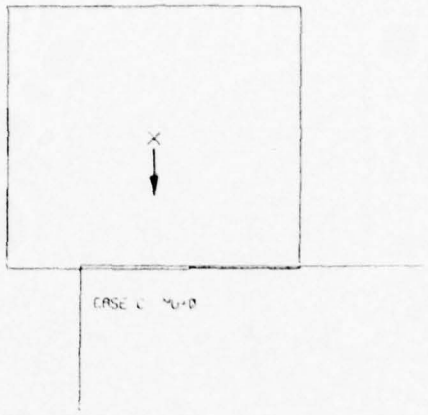


FIG. 3.1 DATA ARRAY FOR BLOCKS - PROGRAM SDEM

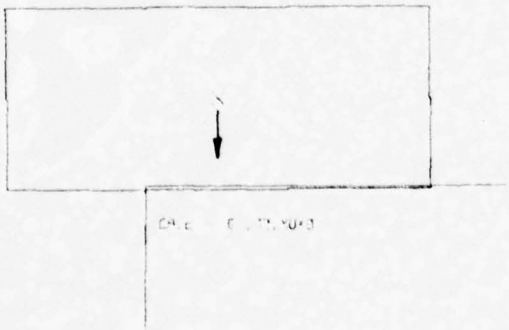
DAMES & MOORE



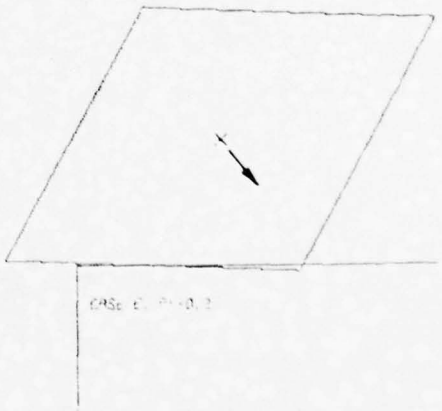
A INITIAL POSITION OF BLOCK - NO APPLIED FORCES



B VERTICAL FORCE ON CENTROID - ZERO POISSON'S RATIO



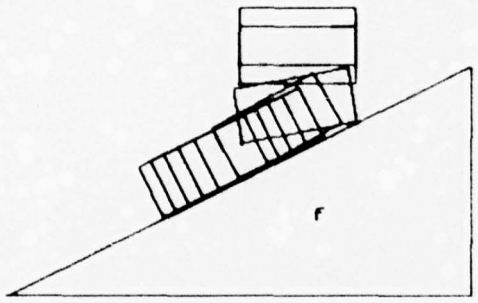
C VERTICAL FORCE ON CENTROID - LOW SHEAR MODULUS, ZERO JOINT FRICTION, HIGH POISSON'S RATIO



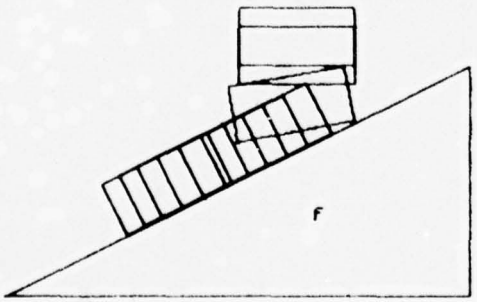
D INCLINED FORCE ON CENTROID - ZERO POISSON'S RATIO, HIGH JOINT FRICTION

FIG. 3.2 STATIC DEFORMATION OF A SINGLE BLOCK ON A RIGID TABLE. THE PLOTS SHOW THE STATIC EQUILIBRIUM POSITIONS OF THE BLOCK UNDER VARIOUS LOADS AND MATERIAL PROPERTIES. ALL DEFORMATIONS ARE REAL, AND NOT MAGNIFIED

DAMES & MOORE



A LOW ELASTIC MODULUS OF SOLID MATERIAL



B HIGH ELASTIC MODULUS

FIG. 3.3 BLOCK FALLING ON TO PLANE - FRICTION ANGLE IS SLIGHTLY HIGHER THAN BLOCK ANGLE

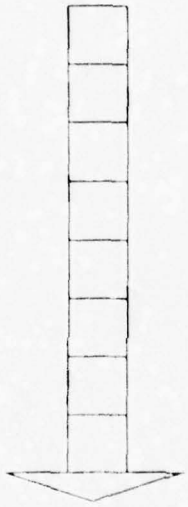


FIG. 3.4 CONFIGURATION OF BLOCKS USED FOR CONTINUUM VALIDATION

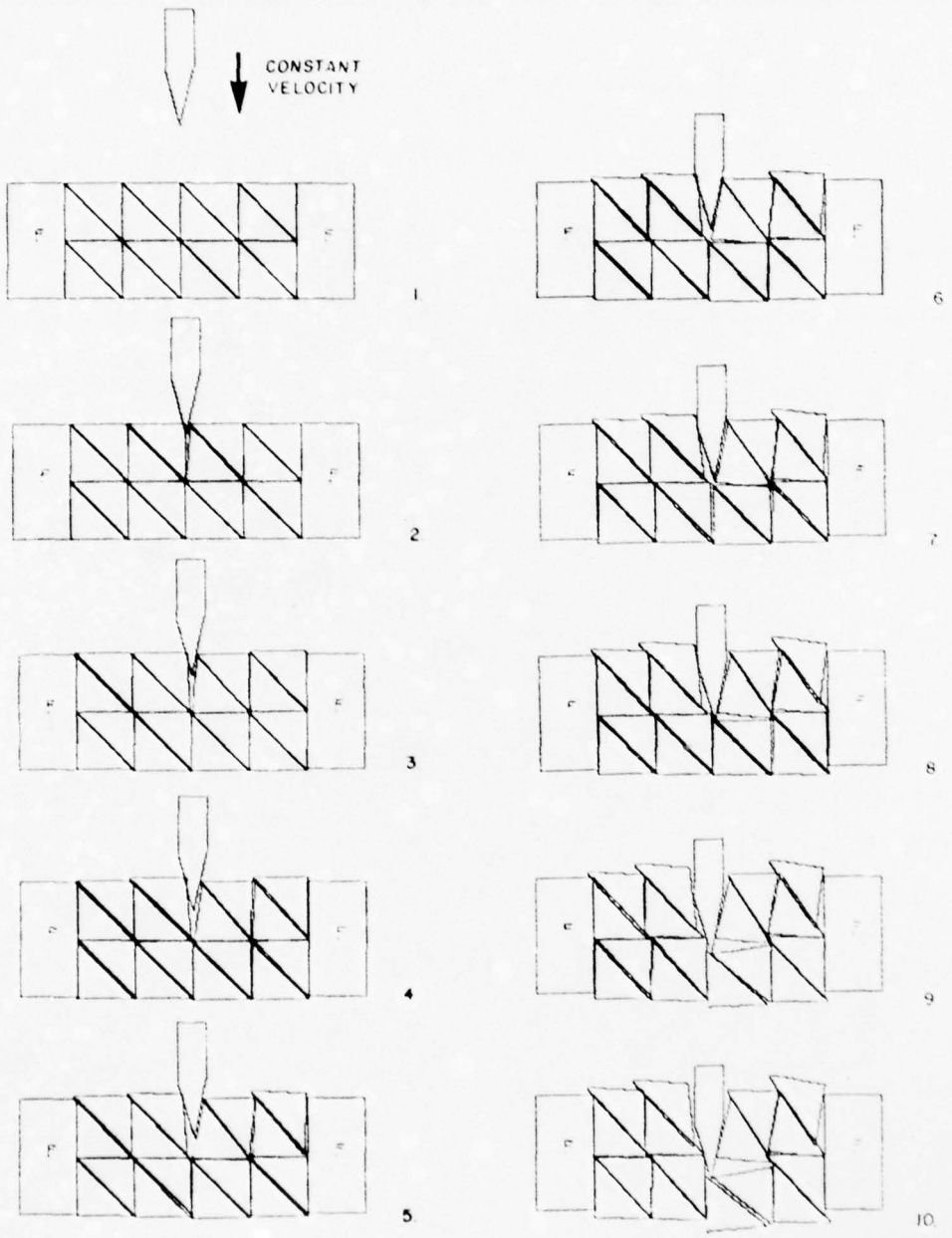


FIG. 3.5 PENETRATION TEST WITH PLASTIC FLOW AND JOINT SLIP - PROGRAM 'SDEM

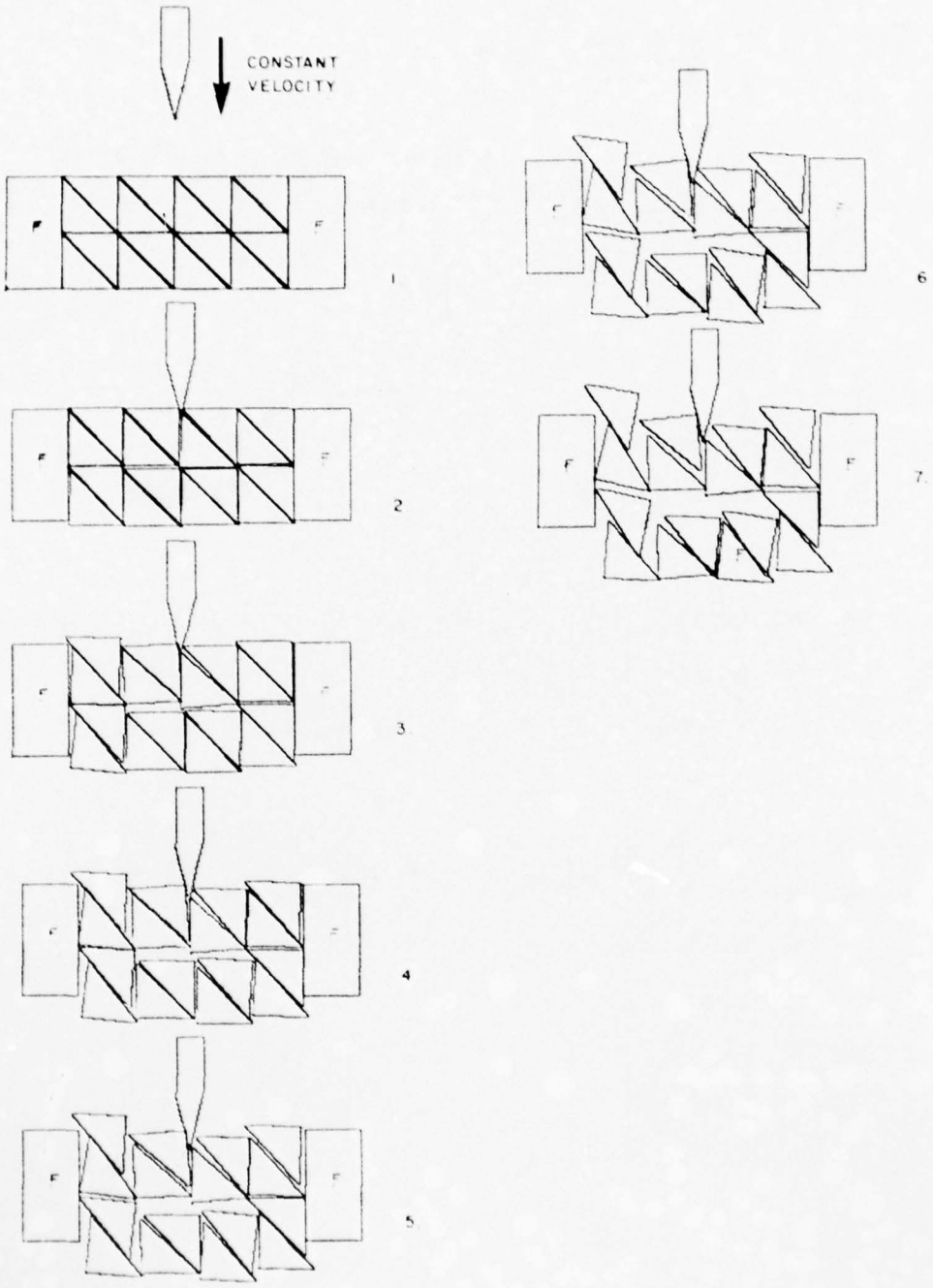


FIG. 3.6 AS FOR FIG. 3.5, BUT USING RIGID BLOCKS - PROGRAM RBM

DAMES & MOORE

CHAPTER 4: INTRODUCTION OF CRACKING INTO THE RIGID-BLOCK PROGRAM

4.0 A new version of RBM is described with modified data structures that allow blocks to be subdivided repeatedly in an efficient manner according to simple crack criteria. Several simple examples are given.

4.1 INTRODUCTION

It is often observed that cracking of intact blocks occurs during laboratory tests on jointed models. Presumably the same sort of thing happens in the field.

Two commonly-observed behaviour-patterns are: cracking of corners and cracking across a block in a manner similar to that of the Brazilian test. It would seem that the realism of the rigid-block program would be considerably enhanced if simple cracking modes could be allowed. For the two modes mentioned, even raw empirical data could be used as the criteria for cracking, although a number of algebraic relations have been proposed to represent the results from tests such as the point-load test (Franklin & Broch). The point here is that it is probably not necessary to go to sophisticated laws, requiring much computer time, in order to increase the realism of the rigid block program considerably. On the contrary, if cracks are not permitted, the modelling of many problems is rendered seriously unrealistic, since "locking-up" occurs when only small overlaps exist between block corners. In practice these corners would break off at relatively low loads, and allow the block motion to continue. Even a crude cracking law is better than no cracking law. In any event, once cracking is permitted, parameter studies may be made, so that the effect of crack criterion on overall behaviour can be determined.

The program RBM was modified to allow blocks to split under the action of applied loads, using user-defined criteria. The new program was

written in such a way that alternative crack criteria can be inserted easily into a skeleton subroutine provided for the purpose. In this subroutine the user is given a list of loads that are currently being applied to each block, together with their location. It is up to the user to determine whether a crack will form, and, if so, where it will form. The built-in logic then splits the block into two, and carries out the various housekeeping tasks, such re-numbering, re-boxing, etc.

4.2 CHANGES TO DATA STRUCTURES

It was necessary to make several changes to the data structures in order to render the program efficient when applying the crack criterion and during the creation of new blocks. The major changes in program logic were as follows:

- Both edges and corners are mapped into boxes (not just corners, as in RBM). This is to ensure that a newly-created block will be guaranteed of finding all possible nearby blocks by interrogating the boxes that it maps into.

- Each contact is pointed to by both blocks concerned (rather than from just one block as in RBM). The reason for this is that any crack criterion needs, for a given block, to have available to it all the contacts involving that block.

- Block corners are stored in a circular linked-list, pointed to by the data for the block concerned. The linked-list format was necessary because corners would have to be re-allocated to new blocks, as blocks cracked. Edge lengths are not stored now.

- Updating (i.e. searching for, and creating new contacts) is now done locally, for each corner that crosses an integer boundary. This was made possible by storing the global coordinates for corners, rather than local. Local updating should be more efficient for most problems.

The detailed changes will now be examined.

4.2.1 FIXED AREAS OF MEMORY

The new program, RBMC, now has only two fixed areas in memory (see Figure 4.1): the lower area contains one word for each potential block. These words point to the data array for each block, which can now be anywhere in the linked-list area. The other fixed area contains one word per box, with each word being a pointer to a list of blocks that map into that box.

4.2.2 BLOCK DATA STRUCTURE

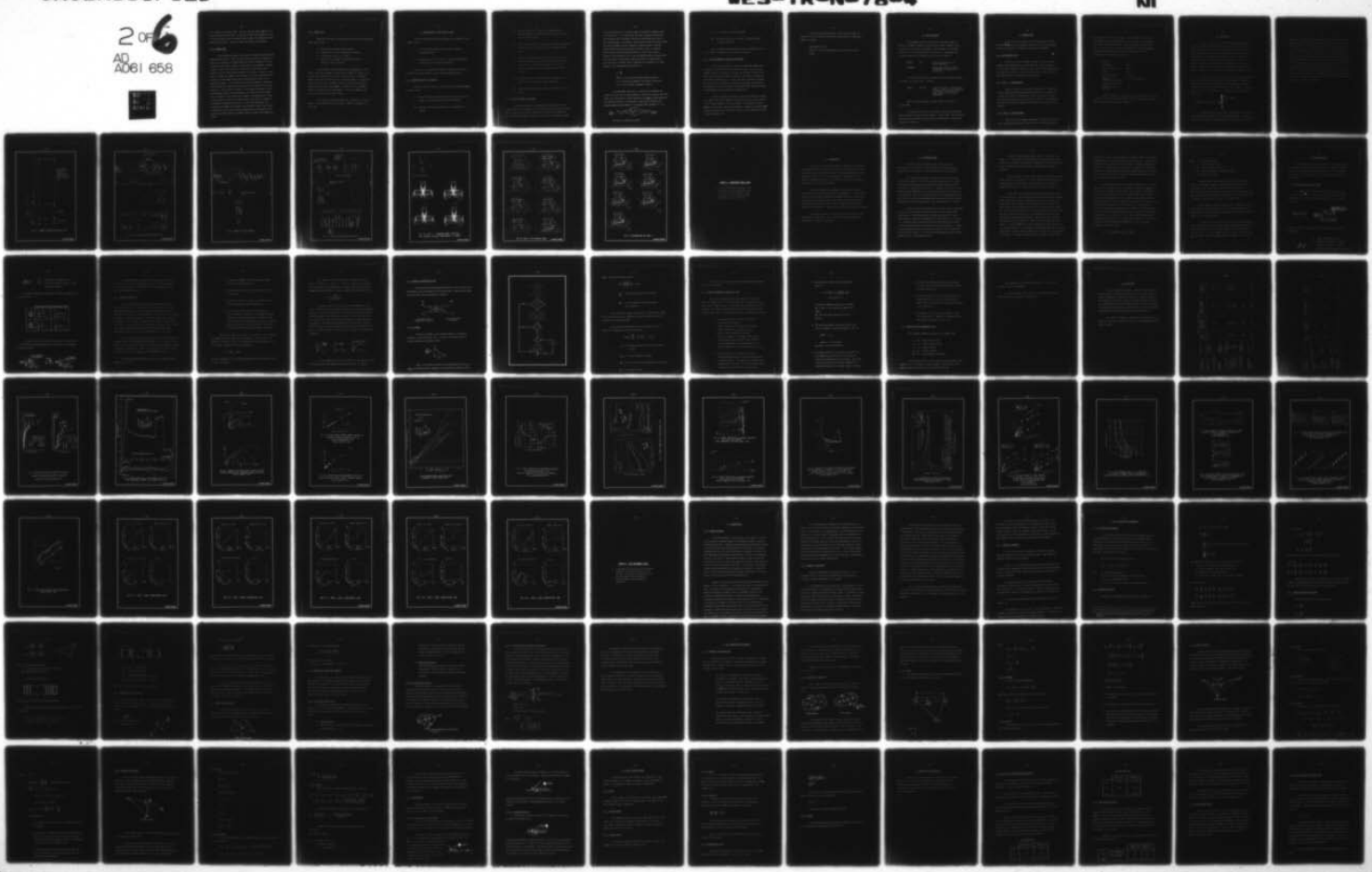
Each block is allocated a basic array of contiguous data as well as a variable-length linked-list containing the corner-coordinates. This arrangement is shown in Figure 4.2, which also gives the format for the two data structures. At the start of a new run the user must estimate the total number of blocks that are likely to be needed during the run. The area of memory between M1 and M2 is allocated on the basis of this estimate. Although the maximum number of blocks must be specified in advance, any number of corners can be accommodated, within the limits of the main memory area.

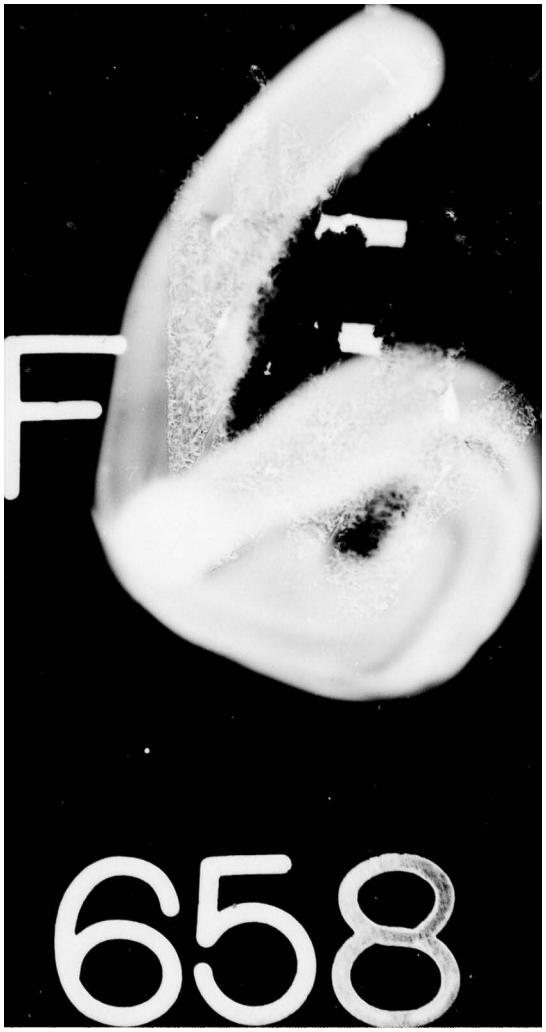
4.2.3 BOX ENTRIES

Each box (which corresponds to one word between M2 and M3) contains either zero, indicating that no blocks map into it, or a pointer

AD-A061 658 DAMES AND MOORE LOS ANGELES CA F/G 8/7
COMPUTER MODELING OF JOINTED ROCK MASSES.(U)
AUG 78 T MAINI, P CUNDALL, J MARTI DACA39-77-C-0004
UNCLASSIFIED WES-TR-N-78-4 NL

2 OF 6
AD
A061 658




658

to a linked-list of double words. The list gives the block numbers of all blocks that map into that box. In contrast to RBM, corner numbers are not stored, since both corners and edges cause entries to be deposited into the boxes that they cross. Figure 4.3 shows the scheme of box entries.

4.2.4 CONTACT DATA

As mentioned previously, each contact is pointed to by both blocks comprising that contact. Word 14 of the block data array (see Figure 4.2) points to a linked-list of double-words that serve as pointers to all the contacts for that block. Figure 4.4 shows such a list pointing to several contacts. Each of the contacts is also pointed to by similar lists emanating from the other blocks involved in the contacts. Figure 4.4 also gives the format of a typical contact array. It will be noted that word 4 contains the block number of the block contributing the corner to the edge/corner contact. It is necessary to store this number to ensure that during scanning for contacts, the forces due to each contact will only be taken once. This is done as follows: if all blocks are scanned, and the corresponding contact lists interrogated, each contact will be accessed twice. However, if the identification number of accessing block is equal to the contents of word 4 of the contact data, the contact is skipped for the purposes of updating that contact. On the other hand, when the accessing block number is not equal to the contents of word 4, the contact is processed, and the resulting forces added into the force sums for the two blocks concerned (which is possible because now both block numbers are known).

4.2.5 EMPTY LISTS

There are four types of data structures located in the linked-list area from M3 to M4:

1) Basic block data arrays, with 14 words;
2) Linked list of corners, consisting of triples;
3) Contact data arrays, of 12 words;
4) Linked lists of doubles, serving either as box entries or contact pointers.

Only two empty lists are maintained, corresponding to 3 and 4 above, since both the number of blocks and the number of corners (1 and 2 above) will always increase, so that there will never be redundant memory associated with these entities. However, numbers of contacts and doubles will fluctuate. Redundant contact arrays are linked together and pointed to by the variable NEMPTC. Similarly a list of redundant doubles is headed by the variable NEMPTD. The variable NEMPTG points to the first word in free (unused) memory.

The concept of a pre-formed empty list stretching to the end of memory, as used in RBM, has been discarded for RBMC; free memory is unstructured.

4.3 INTRODUCTION OF A CRACK INTO A BLOCK

The basic assumptions that have been made in the present version of RBMC are that:

i) a crack forms instantaneously, and splits the block concerned into two;

ii) the decision to create a crack in a block is based solely on the known forces acting on the block.

The process of cracking in the present context has two aspects, the numerical and the physical; each is treated separately below.

4.3.1 NUMERICAL ASPECTS OF CRACKING

When the decision is made to crack a block, the following sequence of events occurs:

(i) The two contacts (between which the crack will extend) are broken (i.e. the forces that were acting vanish).

(ii) The basic data array and pointer for one new block is created.

(iii) Four new corners are created, and, together with the existing corners, are linked up to form the two new corner lists for the two blocks.

(iv) Areas, masses, moments of inertia and centroids are computed for the two blocks, and written into the appropriate data arrays.

(v) Contacts (other than those that produced the crack) that existed around the single original block are transferred to one or other of the two resulting blocks. The contact forces are retained, but a re-numbering and re-linking is necessary.

(vi) Box entries for the original block are converted to box entries for the created block where appropriate.

(vii) A new problem time-step is computed for the whole array of blocks.

4.3.2 PHYSICAL ASPECTS OF CRACKING

Almost any criterion for cracking may be incorporated in the program with little difficulty; in fact several schemes may be used together. The description that follows corresponds to the particular scheme that is contained within RBMC at present. It is based on the tensile cracking

that occurs when more or less point loads are applied on opposite sides of a rock block. In its ideal form, this type of approach constitutes the "Brazilian test" (Fairhurst, 1964) and is performed with a disc of rock loaded between flat plattens. More generally, many "point load tests" have been performed in which irregular or regular blocks of rock are loaded between two indentors that may have small radii of curvature. Such tests have been described by Franklin & Broch (1971), Broch & Franklin (1972) and Brook (1977), and are characterised by a failure load that is related to the distance between the points of application of the loads, and a "strength" that is supposed to be constant for a given material. The relationship for fracture is:

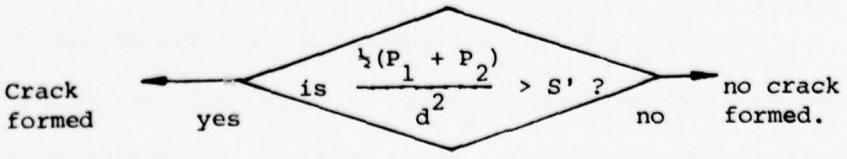
$$S = \frac{KP}{h^2}$$

Where h is the distance between applied loads, P;

K is a factor that depends on the shape of the block;

S is a rock strength, supposedly constant.

In program RBMC, each block is scanned for the maximum two forces, P_1 and P_2 that are being exerted on the block by corners from other blocks (the forces that are being applied by the edges of other blocks are not taken into account). The average is then taken, divided by the distance squared, and compared to a user-specified "strength", S': i.e.



The crack is rejected if either:

(i) d is too small (<1 unit at present)

(ii) the applied forces are too close to existing corners (<2 units at present).

Once a crack has been accepted, the logic described in Section 4.3.1 is used to introduce it into the block concerned.

4.3.3 USE OF ALTERNATIVE CRITERIA FOR CRACKING

The cracking criterion is applied in subroutine CRACK, which is called for each block at every time-step if there are two or more suitable contacts on the block. The user has available to him a list of those contacts that correspond to corners of other blocks touching the block under consideration. It is up to the user to decide if the block will crack, and if so, which two contacts are involved. He then passes the addresses of these two contacts to subroutine SPLIT, which deals with the housekeeping tasks necessary to form the crack.

The list of contacts referred to above is available to the user as a simple array in free memory, and takes the form of a set of pointers to the relevant contacts. The first pointer is contained in location NEMPTG in the array A, and the last pointer is contained in location NEMPT. Access may be made to any of the data stored for each contact (the format is given in Figure 4.4).

If it has been established that a crack will be formed, the pointers to the two contacts involved must then be passed to subroutine SPLIT, as follows:

CALL SPLIT (IC1,IC2),

Where IC1, IC2 are the addresses of the two contact data arrays.

4.4 USE OF PROGRAM

Program RBMC is used in the same way as RBM, with almost the same set of commands. However, density is now a global variable, and is specified when setting up a new problem. The "strength", S, described in Section 4.3.2 is also specified in the same way. Consequently the following commands are now added to those of data set number 1:

DENSITY	RHO	Defines the density for all blocks (A10,F10.0)
TSTRENGTH	S	Defines the cracking "Strength" S, described in Section 4.3.2 (A10,F10.0)

The create command in data set number 2 has been changed slightly to eliminate the specification of density:

CREATE	NC,IFIX	(A10,2I10) Creates a block with NC corners. IFIX $\neq$ 0 for a fixed block. This command is followed in the usual way by a list of corner coordinates.

Apart from these changes, the input stream is identical to that for RBM.

The output format is also very similar, except that the program prints a message informing the user whenever a crack occurs. The coordinates of the crack are printed, together with the new time-step if this differs from the old time-step.

4.5 EXAMPLE RUNS

It was not within the scope of the present study to perform extensive validations on the program; rather, the object of the example runs was to verify that the modified logic worked as intended.

4.5.1 NON-CRACKING TESTS

Validations 5 and 6 described in Chapter 2 (block on plane; Goodman & Bray toppling) were run using RBMC with a high strength in order to inhibit cracking. Identical results were obtained, showing that the translation from local to global coordinates was correct, and that the modified boxing and updating logic was working.

4.5.2 TEST 1: 3-BLOCK PROBLEM

Figure 4.5 shows a case in which a single crack is induced in a block loaded above and below by wedge-shaped blocks. The lower block is fixed, and the other two blocks are acted on by gravity, with the system starting from the state shown with zero velocity. The subsequent movements of the blocks are shown for four values of the coefficient of friction.

4.5.3 TEST 2: 7-BLOCK PROBLEM

This test is more complex and provides a searching test of the logic concerned with cracking, contact detection, re-linking lists and

re-boxing entries. The pile of blocks shown in frame 1 of Figure 4.6 are suddenly acted on by gravity from an initial state of zero velocity. The "rock" has very low strength, so that the subsequent frames (Figs. 4.6 & 4.7) show the ends of the blocks breaking off, and then secondary and tertiary fractures as these pieces strike other blocks. The properties for the run were as follows:

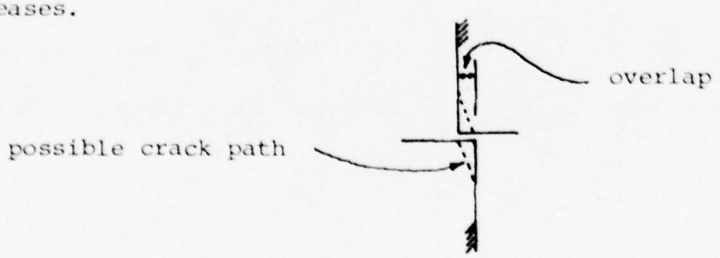
Density:	1.0
"Strength", S:	100.0
Mass of largest block	6000.0
Gravity	-10.0
Contact Stiffnesses	1.0×10^7
Damping (stiffness-proportional)	0.25 at frequency 30.0
Coefficient of friction	0.2
Fraction of critical timestep	0.1

Frames 1 through 10 are plotted at 400 cycle intervals, and frames 11 through 15 are at 800 cycle intervals, since the time-step was reduced when the tiny block was created in frame 10.

4.6 CONCLUSIONS

The data structures and internal housekeeping logic of RBM have been changed considerably to allow cracks to form across blocks. After a block has cracked, the two resulting parts become independent blocks, which can then interact with other blocks in the normal way, and subsequently crack again should conditions permit. Although only a simple criterion for crack formation has been tried so far, the program is written in a way that makes the incorporation of alternative criteria a simple matter. The type of cracking that is permitted at present corresponds to fractures of the type observed in point-load tests.

Corner-cracking, in which a single force near a corner causes the corner to break, has been discussed but not implemented. It appears that the present scheme may not be the best way to overcome the problem of "locking-up" caused by very slight overlap of block corners. The argument runs as follows: the "strength" of a corner will reduce as the overlap decreases.



Consequently even an infinitesimal overlap will cause a crack to form, since the force required will be vanishingly small. An initially sharp-cornered block will acquire several very small extra faces that will

increase the computation time considerably, but will hardly modify the subsequent physical performance of the block. The extra faces will not even guarantee that close encounters with future corners will be trouble-free, since additional tiny chips may break off, in view of the very small strength noted earlier. It may be better to consider other ways of overcoming the "locking-up" problem that are physically equivalent to corners breaking, but do not carry the high computational overhead involved in creating new faces. For example corner-to-corner contacts could be made especially deformable, so that two corners would literally "deform through" each other if the driving force were high enough. Even though such schemes would require more development work, the program RBMC will provide a good starting-point, since the data structures are arranged in a much more convenient way than RBM.

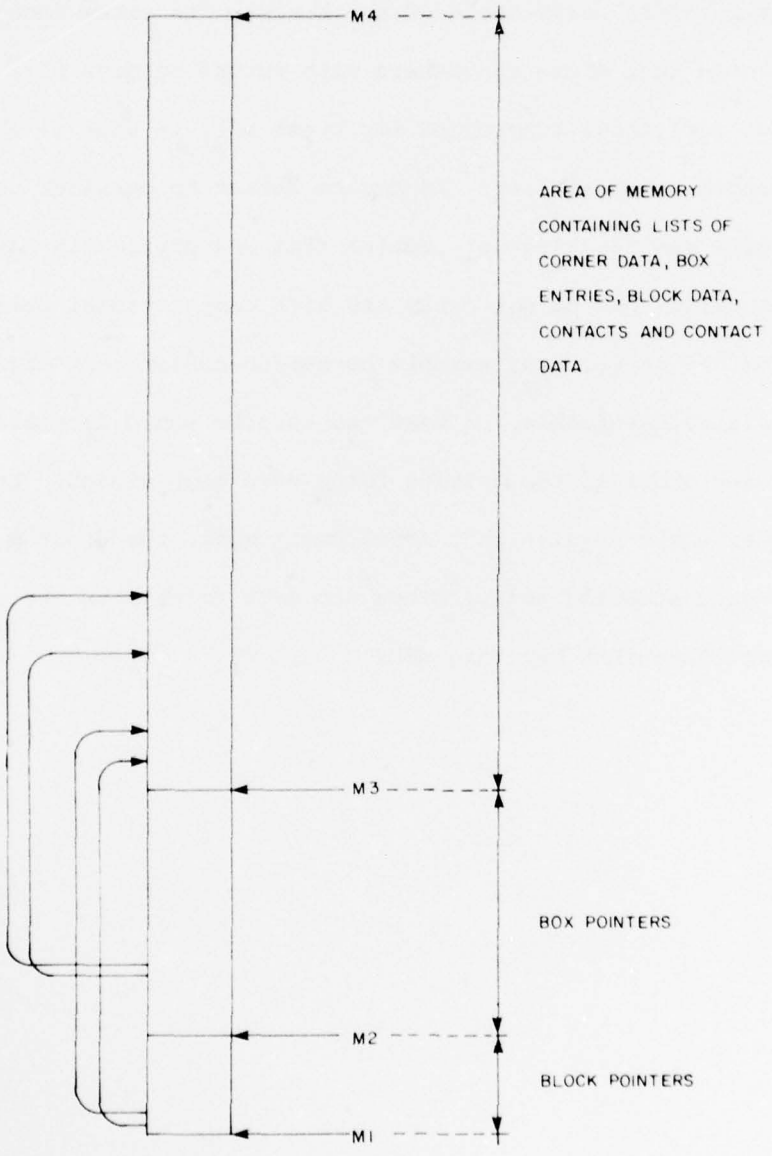


FIG. 4.1 MEMORY ALLOCATION FOR DATA LISTS

DAMES & MOORE

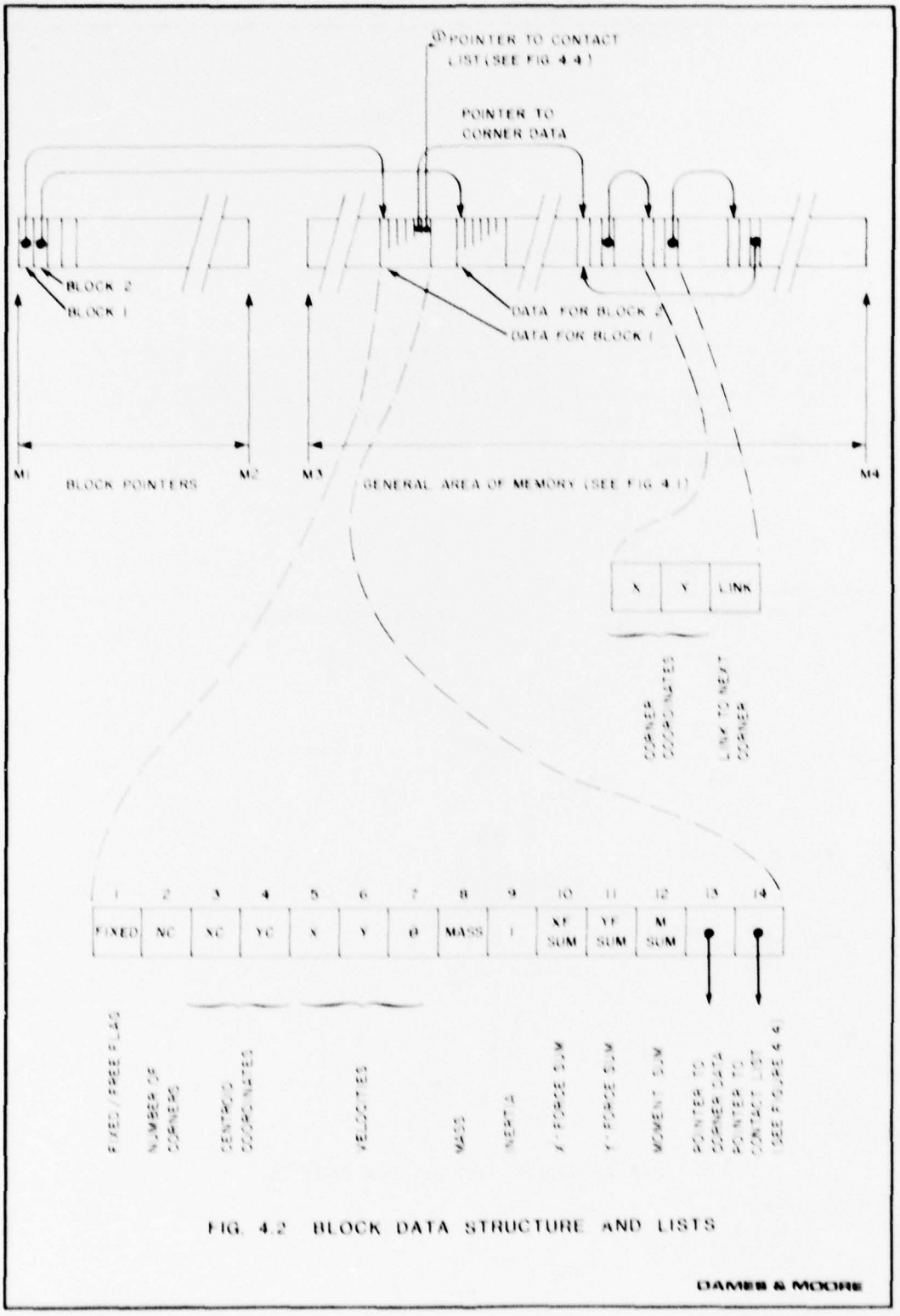


FIG. 4.2 BLOCK DATA STRUCTURE AND LISTS

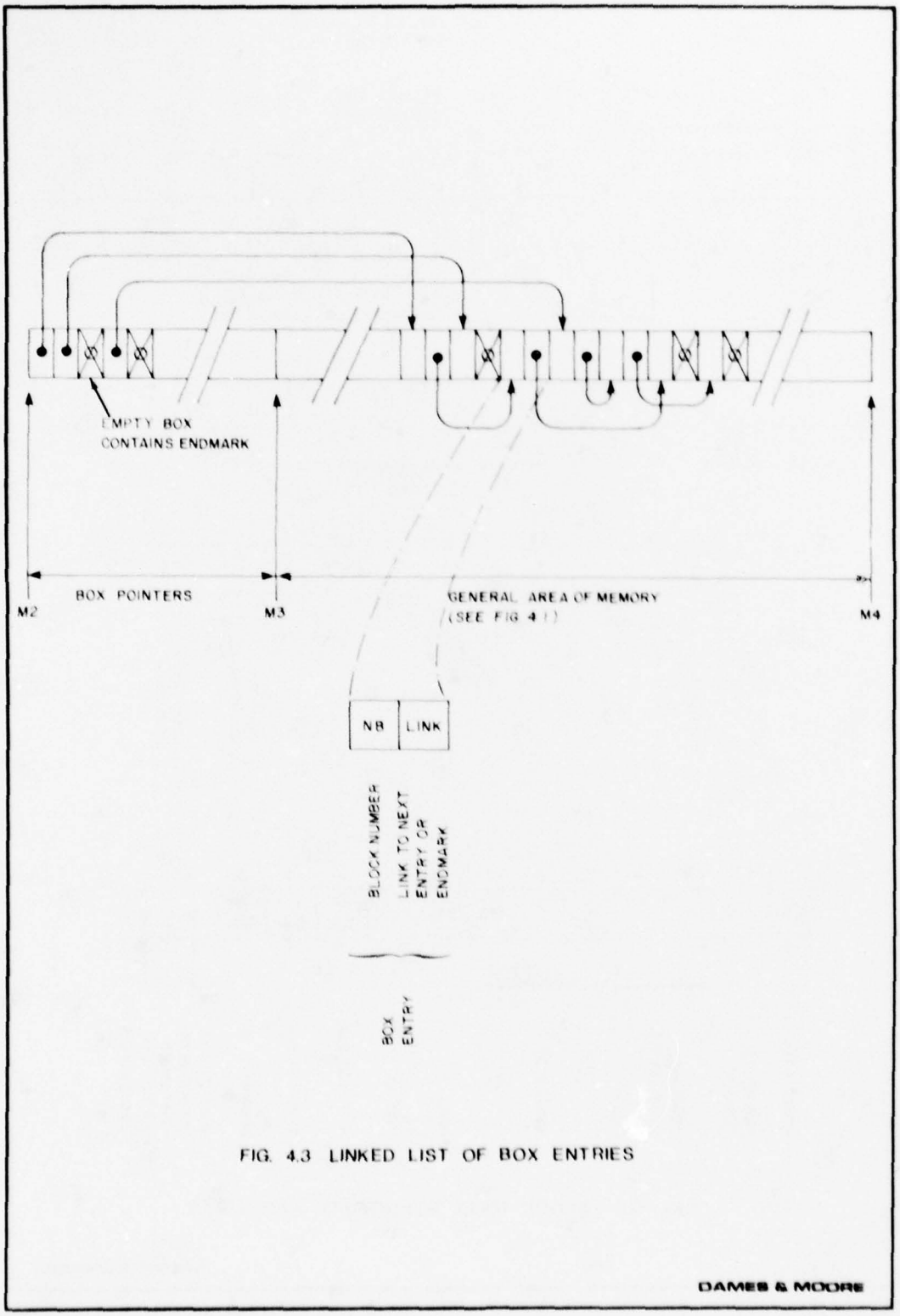


FIG. 4.3 LINKED LIST OF BOX ENTRIES

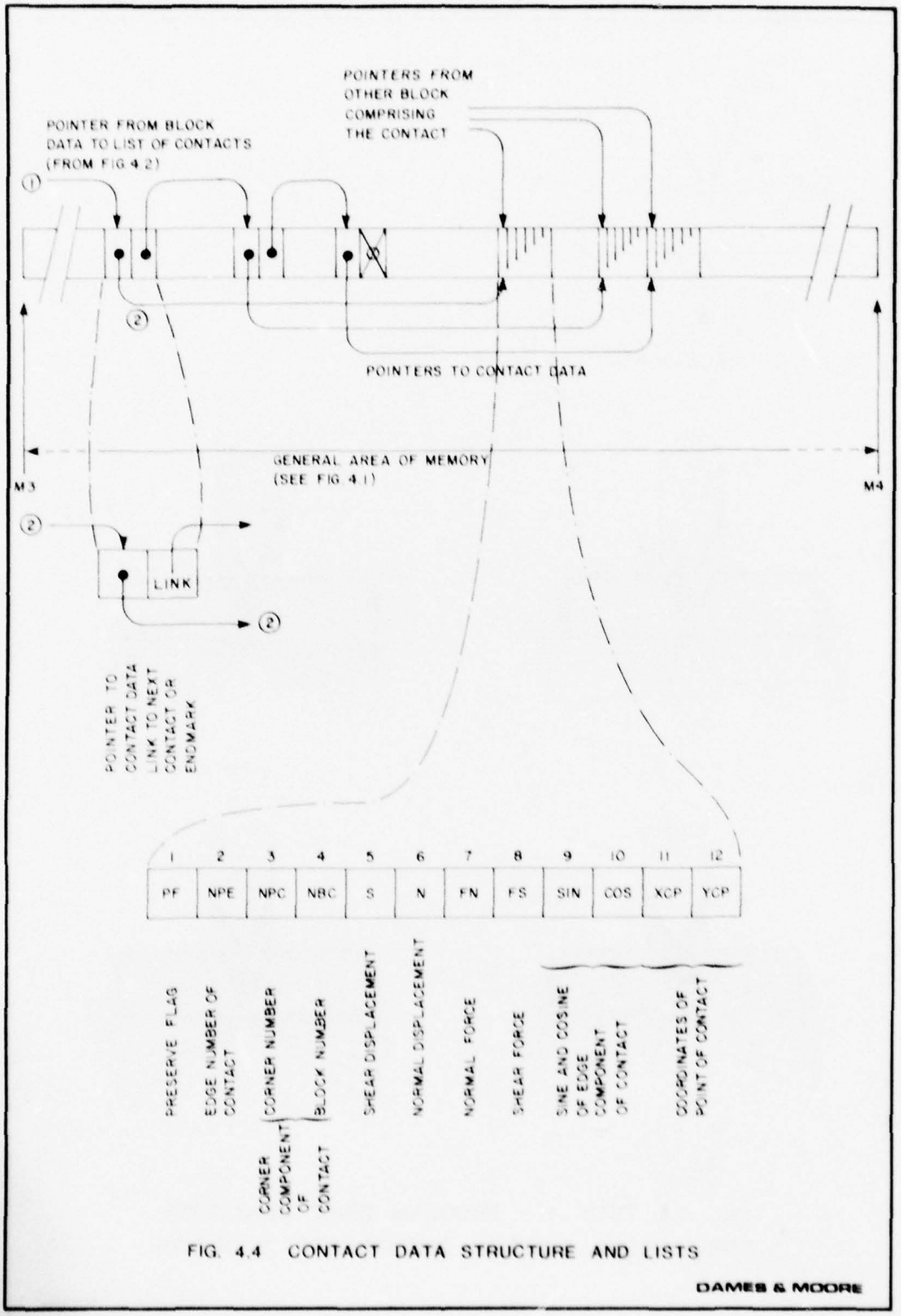
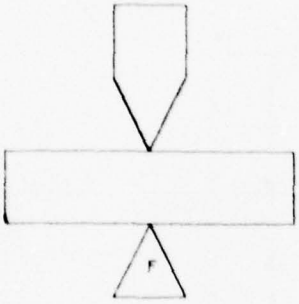
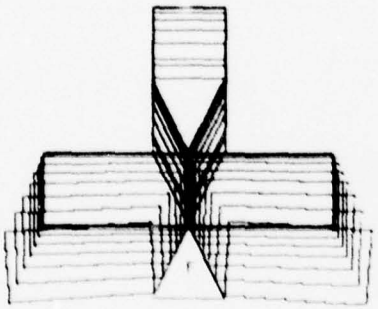


FIG. 4.4 CONTACT DATA STRUCTURE AND LISTS

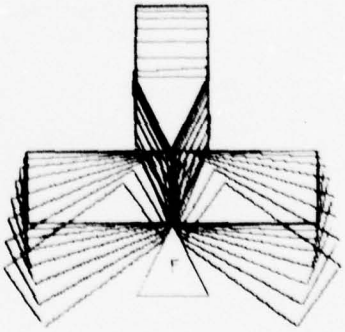
DAMES & MOORE



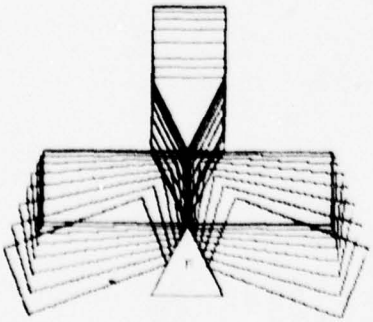
INITIAL CONFIGURATION



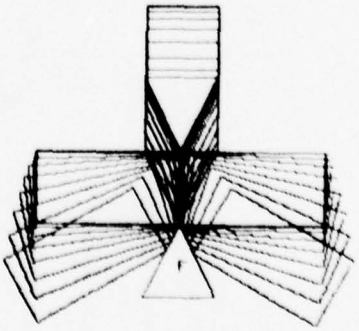
$\mu = 0.0$



$\mu = 0.8$



$\mu = 0.2$



$\mu = 0.4$

FIG. 4.5 TEST 1 - PROGRAM RBMC BEHAVIOUR FOR VARIOUS FRICTION COEFFICIENTS IS SHOWN.

DAMES & MOORE

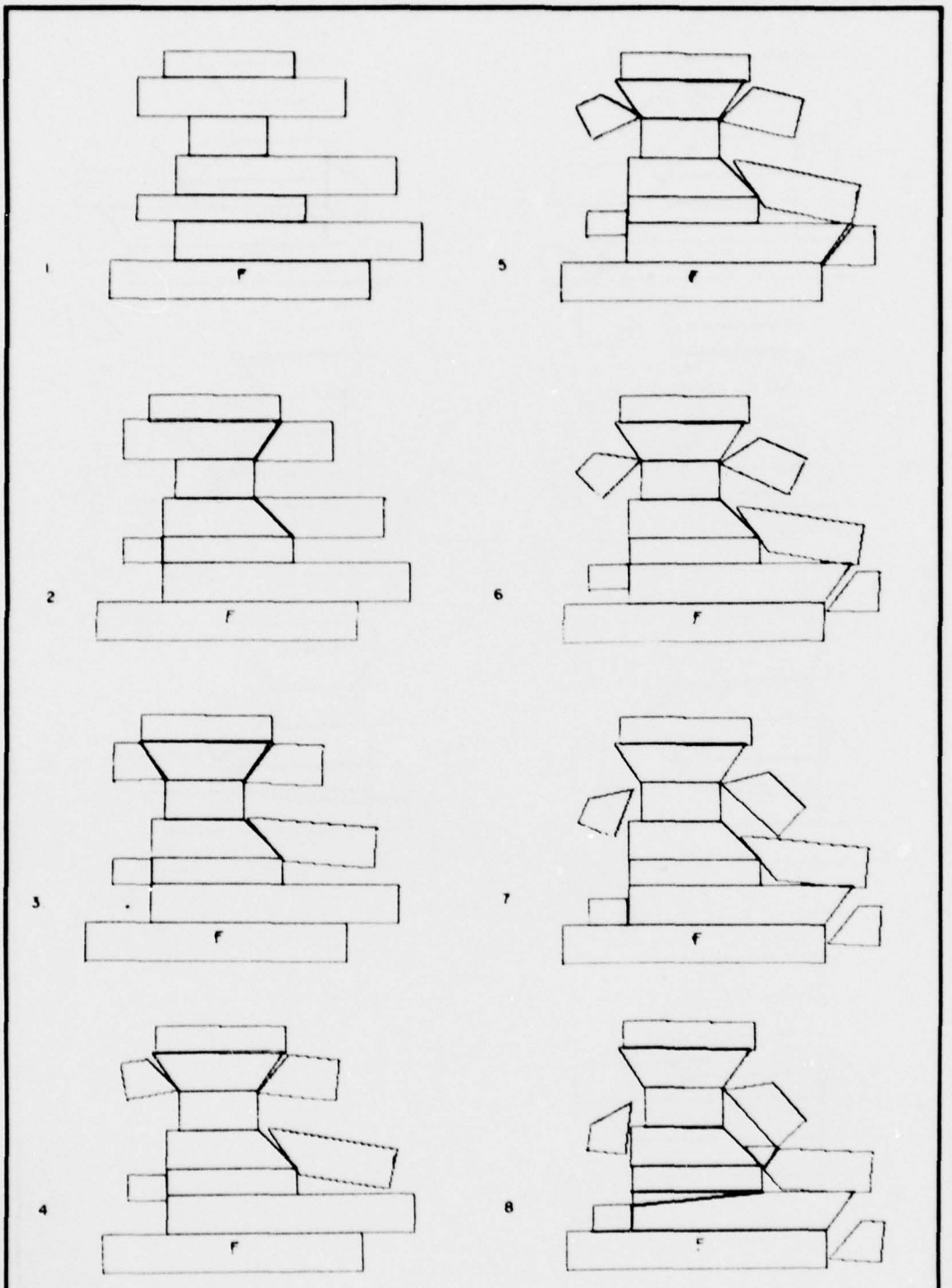


FIG. 4.6 TEST 2 OF PROGRAM RBMC

DAMES & MOORE

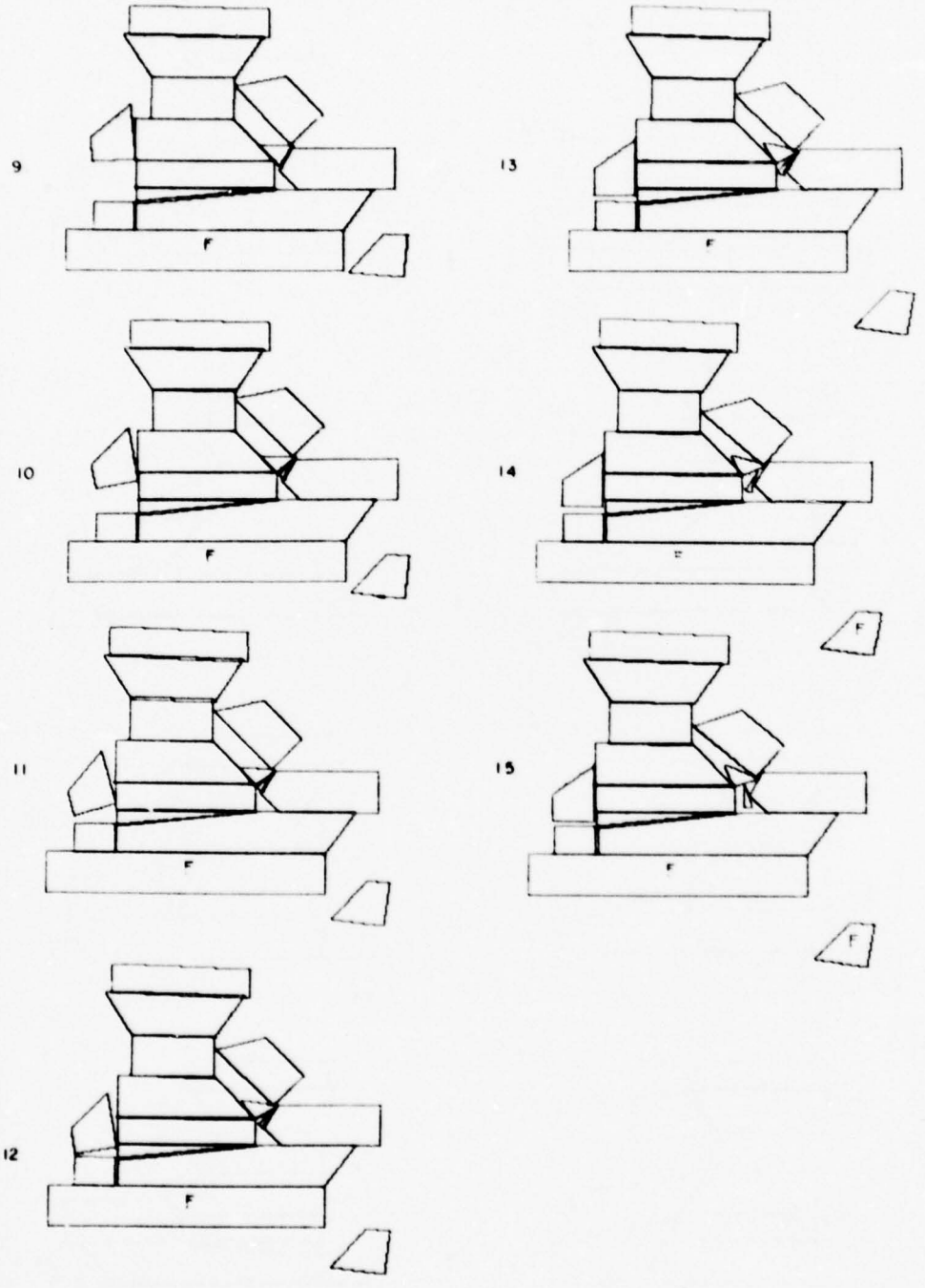


FIG. 4.7 CONTINUATION OF TEST 2

DAMES & MOORE

CHAPTER 5: BEHAVIOUR OF ROCK JOINTS

5.0 *A literature survey is presented on the results from laboratory and field tests on rock joints. The information from the survey is used to develop a simple constitutive law for rock joints.*

5.1 INTRODUCTION

Rock joints are surprisingly complex. They dilate and contract with shearing; their apparent coefficient of friction changes drastically with normal stress and with the history of shearing displacements; their behaviour is sensitive to any filling material, and whether or not it squeezes out during shearing. Any constitutive law must take account of these effects if it is to be realistic.

A bibliography has been assembled on rock joints, and covers laboratory tests, field tests and theories of behaviour. A few references have also been included on numerical and physical models of jointed rock. The next section highlights some of the results from the literature and demonstrates the range of behaviour that is exhibited by rock joints. Based on this information, a simple constitutive law for joints is developed.

The papers referred to in the next section are located in the bibliography of Appendix II, and selected plots from these papers are reproduced at the end of this Chapter.

5.2 LITERATURE SURVEY

Pore water, surface roughness, normal stress, degree of weathering, joint infilling, joint orientation, joint spacing, normal stress and rate of shearing all affect the behaviour of jointed rocks.

Rock joint properties under a variety of conditions have been measured. For example, Iida (1974) ran insitu loading tests on a jointed rock foundation and compared the recorded strains with strains obtained from elasticity equations (see Figure 5.1). Krsmanovic and Popovic (1966a) studied the insitu behaviour of fissured limestone with various degrees of joint infilling. Gale (1975) studied fracture flow in deformable rocks and and found that injection and withdrawal are non-reversible processes, which implies that there must be hysteresis in the normal direction of loading.

Direct shear with controlled normal loads and triaxial compression are two of the most common tests used to study jointed rock properties. Shear stress vs. shear displacement, and normal displacement vs. shear displacements are typically recorded during a direct shear test (see Figure 5.2). Repeated loadings and unloadings may be prescribed and the normal load may be changed several times during the test (see Figure 5.2). During triaxial compression tests, the principal stresses vs. axial displacement are recorded (see Figure 5.3). Pore pressure measurements may be recorded during both shear and compression tests.

Triaxial failure envelopes may be linear or curvilinear (see Figure 5.4). Jaeger (1959) found dry granitic gneiss to give reasonably linear failure envelopes and yet when the samples were soaked in water just prior to testing, the failure envelopes usually showed considerable curvature.

Some rocks dilate linearly during extensive shearing but commonly compact initially (see Figure 5.5). Martin and Millar (1974) reversed the shearing direction in the middle of direct shear tests and found the greywacke samples first to contract and then to dilate (see Figure 5.6).

Shear stress vs. shear displacement curves generally fall into two classifications. Some jointed rocks are stiff during initial shearing, reach a peak value of shear strength which then drops off rapidly, and finally taper off in strength to some residual value (see Figure 5.7a). Others exhibit elastic behaviour during initial shearing and then behave more or less plastically with little or no distinct peak (see Figure 5.7b). The slope of the elastic portion of stress-displacement curves has been designated by Goodman, Taylor and Brekke (1968) as the shear stiffness. For example, the shear stiffness of garnet schist with quartz beds (see Figure 5.7) equals 10.4MN/m^2cm. Values obtained from various shear tests are tabulated in Table 5.1. The type of stress-displacement curve is greatly influenced by the shear surface roughness and the joint infilling properties of the joint. For example, Krsmanovic, Tufo, and Langof (1966b) found

the presence of clay infilling material to change the shape of the stress-displacement curve for fissured limestone (see Figure 5.8). Jaeger (1971) recorded dramatic changes in the shear behaviour of bowral trachyte upon wearing of the shear surface (see Figure 5.9). Kutter (1974) also found reshearing to modify the shear stress-shear displacement curve (see Figure 5.10). These changes can be related to the selective destruction of the asperities present on both rock faces.

Using plaster models, Patton (1966b), Krsmanovic et. al. (1966b) Barton (1971c), Schneider (1974) and others have studies how asperities control shear at different normal loads. Patton (1966) and Krsmanovic et. al (1966) sheared plaster models across planes having teeth of various inclinations and frequencies. At low normal stresses, Patton found the effective angles of friction to equal the internal angle of friction of the plaster plus the asperity angle (see Figure 5.11). At high normal stresses, Patton found the asperities to be sheared off and the effective angle of friction to equal the internal angle of friction of the plaster (see Figure 5.11). Barton (1971c, 1974) statistically analysed several hundred direct shear tests run on plaster and jointed rocks: he found the following relationship between the peak shear strength, the surface roughness and the normal load:

$$\tau/\sigma_n' = \tan \left\{ JRC \cdot \log_{10}(JCS/\sigma_n') + \phi_b \right\}$$

where τ = peak shear strength

σ_n' = effective normal stress

JRC = joint roughness coefficient

JCS = effective joint wall compressive strength

ϕ_b = base friction angle

Schneider (1974) sheared plaster samples cast from granite, limestone, and sandstone samples. Dilation curves, shear stress-shear displacement curves, and shear stress-normal stress curves for the three surface types vary considerably (see Figure 5.13). These results are interesting because they show how the roughness alone affects the behaviour, for identical intact material strength. (See Figure 5.13).

Patton (1966a) recognised that two scales of roughness govern the shear behaviour of rocks (see Figure 5.14). The behaviour during small displacements and the large second order irregularities govern the shearing behaviour for large displacements.

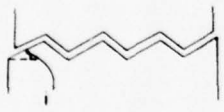
In general it can be said that at low normal loads, the behaviour of the joint will be influenced by the highest-angle (smallest) asperities while at higher normal stresses, these asperities will be sheared off, so that the longer, lower-angle asperities will be the major influence. This type of behaviour must be recognised by any constitutive law for joints.

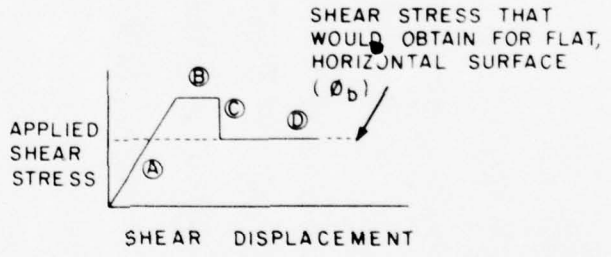
5.3 CONSTITUTIVE LAWS

The preceding discussion has shown that joint roughness plays a major part in the mechanical behaviour of joints. However, roughness is not a constant attribute, and account must be taken of the effects that changes in roughness produce, due to asperities being worn down or destroyed.

5.3.1 MECHANICS OF AN IDEALISED JOINT

As a preliminary to the development of a constitutive law, consider the mechanics of a simplified joint, with a single wavelength of asperity; and its idealised stress/displacement curve for a given normal load:



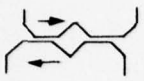


The various parts of the curve may be identified with the following mechanisms:

A. elastic straining - no slip

B. sliding up on asperities - apparent friction angle = $\varphi_b + i$, where φ_b = basic friction angle of the material.

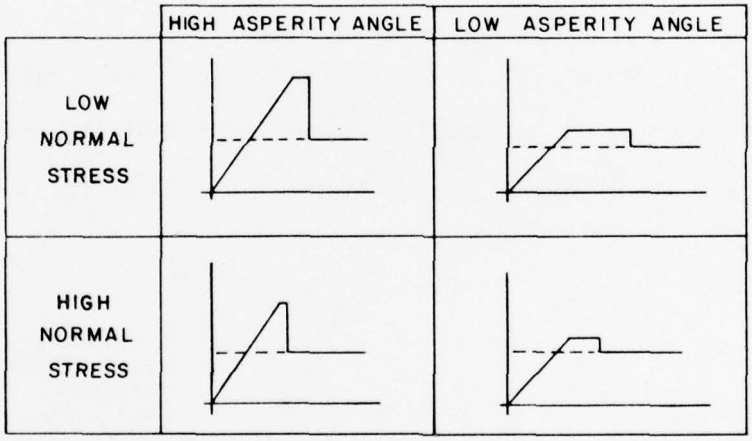




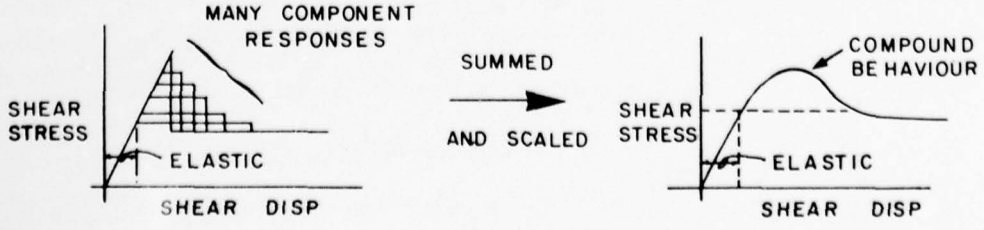
C. shearing off of asperity-tops

D. sliding on horizontal surfaces created by sheared asperities.

The curve shown above is modified for changes in asperity angle and normal stress:



Clearly, a real joint is made up of many sizes of asperity, so that the resultant stress/displacement curve is more or less a summation of many basic curves:



In conjunction with the shear behaviour described above, the joint also tries to dilate, the major part of which occurs during phase B - the sliding-up phase, characterized by the flat-topped part of the shear stress/displacement curve.

5.3.2 SIMPLIFIED BEHAVIOUR

At any stage during an arbitrarily-complex loading history, a joint will have some of its asperities sheared, while others (of longer wavelength) will be intact. Of the asperities that have been sheared, some will have only their top parts broken, but others may be almost completely destroyed. In order for a numerical scheme to be able to model completely the mechanics of such a system, a memory would be needed of the proportion of each size of asperity that had been destroyed at any given time. Clearly, such computer storage requirements would be impracticable and indeed unwarranted, given the very limited experimental data on rock joints.

An alternative, approximate, approach will be considered here that reproduces several of the observed characteristics of real joints, plus some effects that have not been the subject of testing, but could well be verified, or otherwise, in the future. Several of the concepts used in what follows are due to Barton (see, for example, Barton and Choubey, 1978).

Before developing the constitutive law, several (perhaps obvious) points will be stated:

1) The peaks (or "bumps") on shear stress/displacement curves are caused by asperities.

2) Shearing removes asperities and causes damage to the joint surface.

3) The damage is greater for greater σ_n (normal stress) and for increasing U_s (shear displacement).

4) If a joint is sheared at very low σ_n, it will be almost undamaged; consequently, its subsequent shearing behaviour for high σ_n will hardly be affected, except that the shear stress/displacement curve will be displaced bodily by the amount of low-σ_n shearing. (This ignores the possibility that mating joints become non-mating.)

These points lead to the concept of a variable, D, that accumulates the damage done to the asperities during shearing. Clearly there is no unique way to define D, since it is an approximate measure anyway, but as a first attempt, consider the following expression:

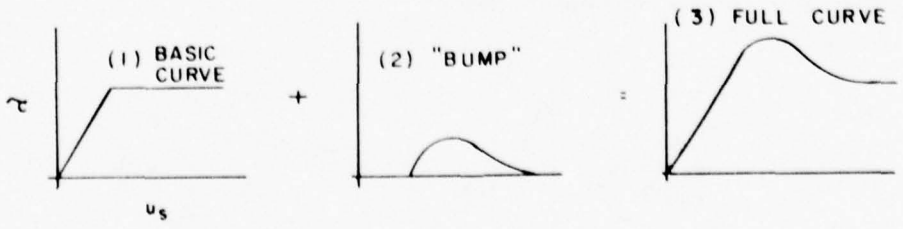
$$D = \Sigma\, \sigma_n \Delta U_s \quad , \text{ where}$$

the sum is performed for all loading increments during which the shear stress is above the residual.

This expression satisfies the obvious conditions that D must vanish for either $\sigma_n = 0$ or $\Delta U_s = 0$. It may be normalised if necessary to the range $0.0 < \bar{D} < 1.0$ by dividing by the maximum normal stress times the maximum shear diplacement for residual strength under that normal stress:

$$\bar{D} = \frac{\Sigma \sigma_n \Delta U_s}{\sigma_{n\,(jcs)} U_{s\,(max)}}$$

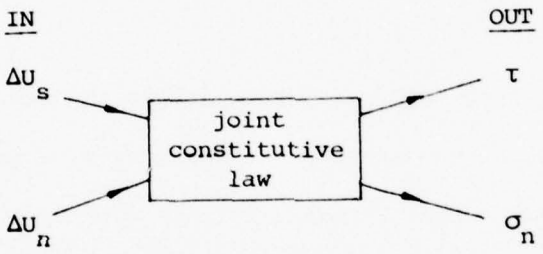
The main utility of D is that it allows the damage done at one value of normal stress to be carried over and used as a basis for subsequent shearing at a different normal stress. This is done by shifting the shear stress/displacement curve over until its value of D (i.e. the value of D that would have obtained from shearing at constant σ_n) equals the current accumulated D. The shear stress/displacement curve referred to above is actually that part of the full curve that lies above the residual-strength part: i.e. only the "bump" of the full curve is stored, for the purpose of computation.



It is the "bump" that reflects the effects of the asperities; after all asperities have been removed, only the basic curve, (1), remains.

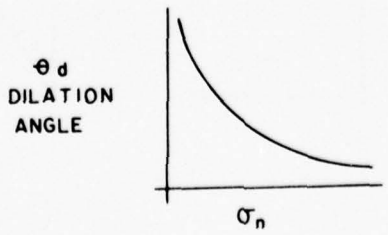
5.3.3 DETAILS OF CONSTITUTIVE LAW

In simplified form the proposed constitutive law for joints can be represented by the flow diagram on the following page. Note that the constitutive law works from displacements to stresses:

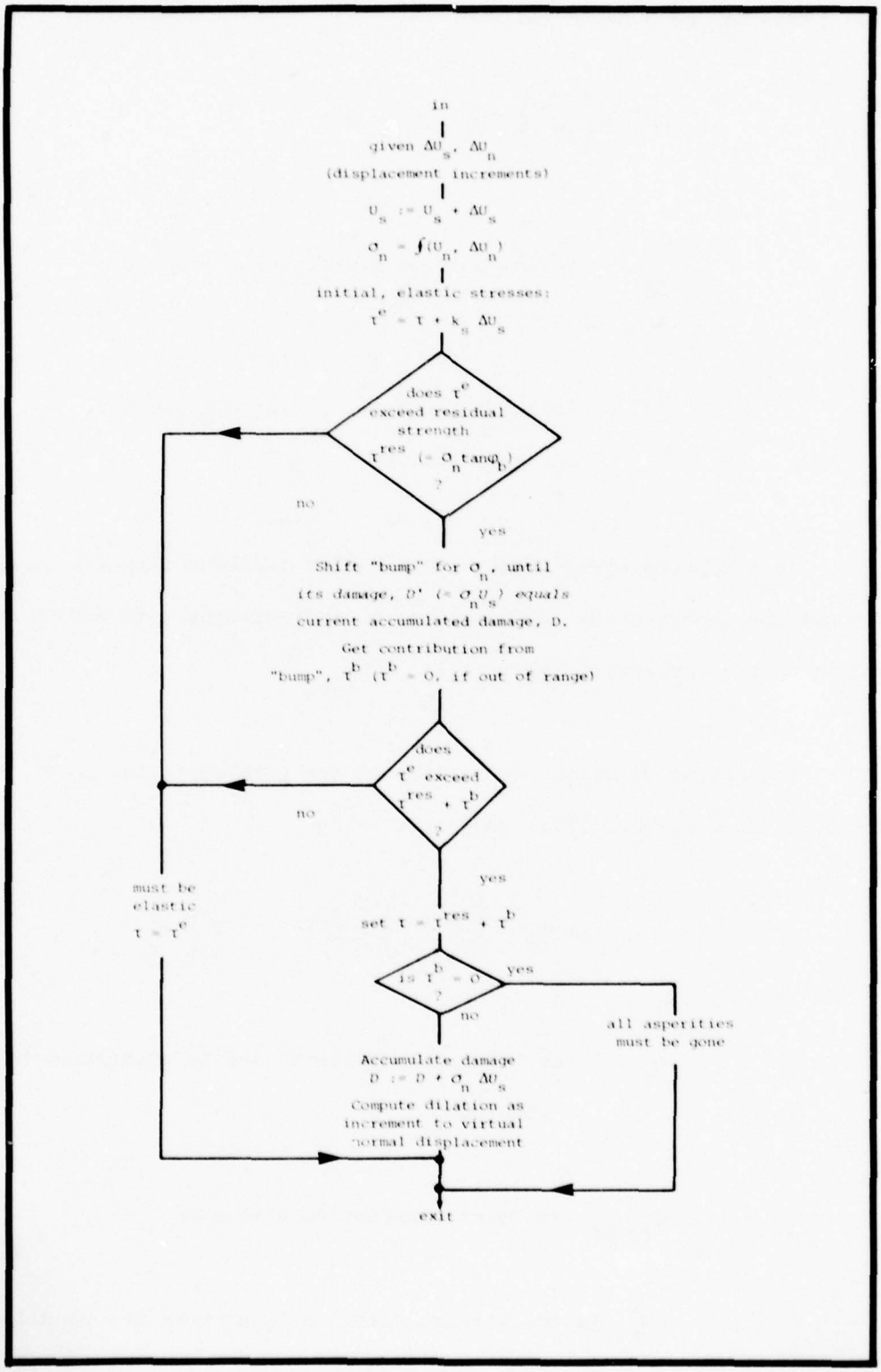


5.3.4 DILATION

Dilation is assumed to occur whenever damage is accumulated, according to some user-defined law. In general, the dilation angle is related inversely to the normal stress:



Since, in the explicit method, both displacements U_s and U_n are given, the dilation must be expressed as an equivalent increase in normal


in
given ΔU_s, ΔU_n
(displacement increments)
$U_s := U_s + \Delta U_s$
$\sigma_n = f(U_n, \Delta U_n)$
initial, elastic stresses:
$\tau^e = \tau + k_s \Delta U_s$
does τ^e exceed residual strength τ^{res} (= $\sigma_n \tan\phi_b$) ?
no
yes
Shift "bump" for σ_n, until its damage, D' (= $\sigma_n U_s$) equals current accumulated damage, D.
Get contribution from "bump", τ^b (τ^b = 0, if out of range)
does τ^e exceed $\tau^{res} + \tau^b$?
no
yes
must be elastic
$\tau = \tau^e$
set $\tau = \tau^{res} + \tau^b$
is τ^b = 0 ?
yes
no
all asperities must be gone
Accumulate damage
$D := D + \sigma_n \Delta U_s$
Compute dilation as increment to virtual normal displacement
exit

stress. This may be done as follows:

$$\Delta\sigma_n = \left(\frac{\partial\sigma_n}{\partial U_n}\right) \Delta U_n^d \quad , \text{ where}$$

$\frac{\partial\sigma_n}{\partial U_n}$ is the current normal joint stiffness

ΔU_n^d is the incremental normal displacement due to dilation

In the plots given later, the dilation is shown directly as ΔU_n^d, but in use the constitutive law would use this displacement to modify σ_n according to the expression above.

The actual dilation law built into the program is based on Figure 5.12 (from Barton, 1971) and is given by:

$$\tan \Theta_d = \frac{\Delta U_n^v}{\Delta U_s} = \left\{ \frac{\sigma_{(jcs)}}{\sigma_n} - 1 \right\} k_d$$

k_d is the "dilation constant" and is prescribed by the user

$\sigma_{(jcs)}$ is joint compressive strength

ΔU_n^v is the virtual normal displacement due to dilation

Θ_d is the dilation angle.

Dilation, U_n^v , is only accumulated during periods when "damage", D, is being accumulated.

5.3.5 COMPUTER PROGRAM FOR CONSTITUTIVE LAW

The steps described above have been embodied in a computer subroutine, JOINT, which is listed in Appendix XV. The program is quite simple and is probably easier to follow than a written description of the procedure. All important variables are defined in the listing, but the following notes should be consulted before trying to understand the program:

1) The program is written in terms of normalised displacements and stresses.
"Real" stresses may be obtained by multiplying output stresses by $\sigma_{(jcs)}$, where $\sigma_{(jcs)}$ is Barton's "joint compressive strength".
"Real" displacements are obtained by multiplying internal displacements by $U_s^{(max)}$, where $U_s^{(max)}$ is the shear displacement at which the shear stress drops to residual for a test performed for $\sigma_n = \sigma_{(jcs)}$.

2) The basic curve for the "bump" on the shear stress/displacement curve is built into function FSS as a DATA statement. It comprises 10 intervals, with maximum value of 1.0 units, but is subsequently scaled according to σ_n using Barton's law (see note 3).

3) The peak shear strength is given by Barton's equation:

$$\tau = \sigma_n \tan \left[JRC \log_{10} \left\{ \frac{JCS}{\sigma_n} \right\} + \varphi_b \right]$$

(see Section 5.2)

It should be remembered that φ_b is in degrees.
JRC is input to the program via common block /FRIC/.
Any other law can be substituted in FSS if desired.

4) The shear displacement at which the shear stress falls to residual is adjusted according to the law:

$$U_s^{(max)} = K \sigma_n ,$$

where K = 1.0 at present.

Any other law may be substituted.

5) At present dilation is not coupled up in such a way as to modify σ_n (see Section 5.3.4), purely for purposes of presentation of test results. The virtual normal displacement due to dilation (UND(JID)) is computed internally, but not used, except for display.

6) The normal stress/displacement function at present is linear, and is evaluated in Function FSN. Another law may be substituted.

7) The normal stress must lie in the range $0.0 < \sigma_n < 1.0$, otherwise errors will occur. No traps have been incorporated. Also inaccurate results may be expected for σ_n close to zero.

8) The dimension of 50 refers to the number of joints that can be handled -- JOINT has to remember stresses, displacements and damage for each joint.

5.3.6 EXAMPLE RUNS WITH SUBROUTINE JOINT

The following parameters were used in all example runs:

$\varphi_b = 30^o$ basic friction angle

$k_s = 1.0$ initial shear stiffness

$k_n = 1.0$ normal stiffness

$k_d = 0.1$ dilation constant

JRC = 60 joint roughness coefficient

As mentioned earlier, the basic shear stress/displacement curve was built into the program in a DATA statement in Function FSS. The listing should be consulted for the values that were used.

Five example runs are presented in Figures 5.14 to 5.18, and are more or less self-explanatory.

In particular, Run 5 shows that the stress/displacement curves look quite realistic for complex loading paths.

5.4 CONCLUSIONS

A first attempt has been made to develop a constitutive law that will handle the complex loading paths that occur when running computer simulations of jointed rock behaviour. Although the results look plausible, there is not sufficient experimental data from tests with complex paths to check the assumptions made. More data is needed for cases where normal stresses are varying during a test in which shear displacement follows a time history with repeated reversals.

The concept of "damage" is thought to be promising, and it is hoped that it will be refined in the future as more experimental results become available.

TABLE 5.1 SHEAR STIFFNESS VALUES

ROCK TYPE - TEST DESCRIPTION	SOURCE	NORMAL STRESS (MN/m^2)	MAXIMUM SHEAR STRENGTH IN ELASTIC REGION	MAXIMUM DISPLACEMENT IN ELASTIC REGION (cm)	SHEAR STIFFNESS ($MN/m^2/cm$)
QUARTZ DIORITE	PRATT ET. AL. (1974a) Fig.3 p.308	NOT GIVEN			
Joint Area=142cm^2			8.75	0.024	3.6×10^2
Joint Area=5130cm^2			1.40	0.12	1.2×10^1
WEATHERED GREYWACKE	MARTIN & MILLAR (1974) Fig.4 p.266	0.49	0.9	0.043	2.1×10^1
FISSURED LIMESTONE	KRSMANOVIC ET.AL. (1966b)				
(No Infilling)	Fig 3	3.6	2.45	0.038	6.5×10^1
(No Infilling)	Fig 3	1.9	2.45	0.077	3.2×10^1
(No Infilling)	Fig 3	1.12	2.15	0.115	1.87×10^1
(No Infilling)	Fig 3	0.15	0.665	0.150	4.4
(Clayey Infilling)	Fig 4	2.4	0.42	0.025	1.7×10^1
(Clayey Infilling)	Fig 4	1.6	0.34	0.021	1.63×10^1
(Clayey Infilling)	Fig 4	0.77	0.176	0.042	4.2
FISSURED LIMESTONE (Insitu Shear Test)	KRSMANOVIC ET. AL. (1966a) Fig. 3p.775	1.47	0.51	0.10	5.1
		0.49	0.51	0.065	7.9
		0.98	0.64	0.125	5.1
ATALAYA PORPHYRY (Sample No. 8)	KUTTER (1974) p.10				
(1st run)		0.86	1.36	0.15	9.1
(2nd run)		1.81	0.65	0.105	6.2

TABLE 5 (Cont/d.)

ROCK TYPE - TEST DESCRIPTION	SOURCE	NORMAL STRESS (MN/m^2)	MAXIMUM SHEAR STRENGTH IN ELASTIC REGION	MAXIMUM DISPLACEMENT IN ELASTIC REGION (cm)	SHEAR STIFFNESS ($MN/m^2/cm$)
DARLEY DALE SANDSTONE (Sample No.1)	KUTTER (1974) P.14				
(1st Run)		1.40	1.05	0.29	3.6
(2nd run)		1.38	1.05	0.29	3.6
(3rd run)		1.34	0.71	0.23	3.1
DELABOLE SLATE (Sample No. 1)	KUTTER (1974) P.32				
(1st run)		0.66	0.57	0.18	3.1
(2nd run)		0.668	0.273	0.097	2.8
(3rd run)		1.32	0.161	0.064	2.5
SCHISTOSE BANDED HEMITITE QUARTZITE	KUTTER (1974) p.39	0.979	1.04	0.25	4.2
GARNET SCHIST WITH QUARTZ BEDS (Sample No. 5)	KUTTER (1974) p.41	5.54	3.8	0.64	5.9

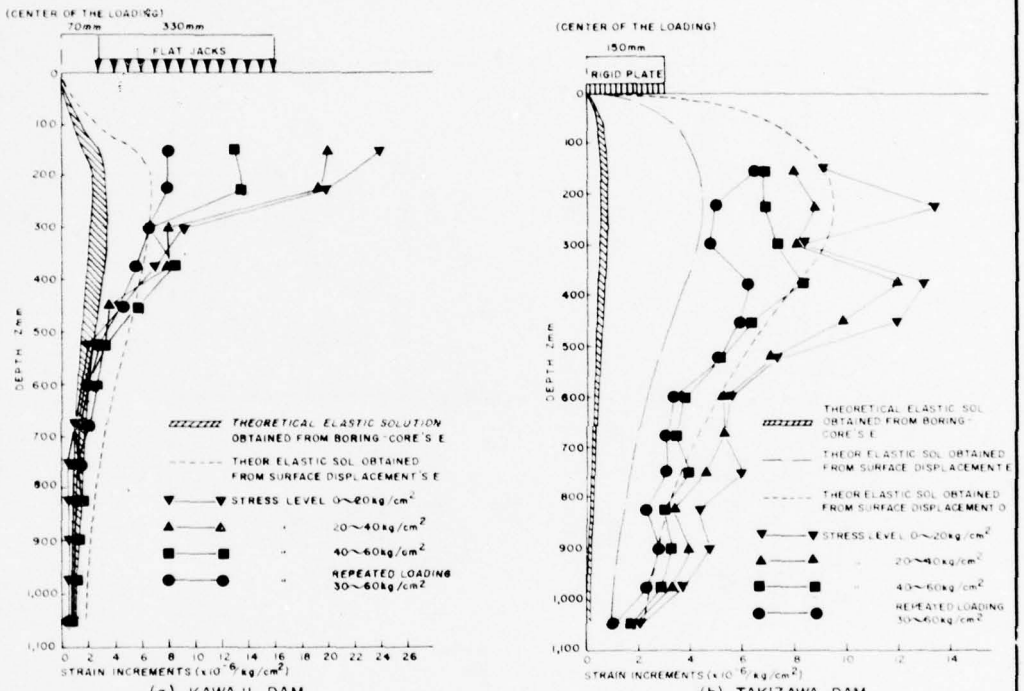


FIG 5.1 RECORDED STRAINS IN JOINTED ROCK VERSUS STRAINS OBTAINED FROM ELASTICITY EQUATIONS (AFTER IIDA, HOJO AND HARADA, 1974)

DAMES & MOORE

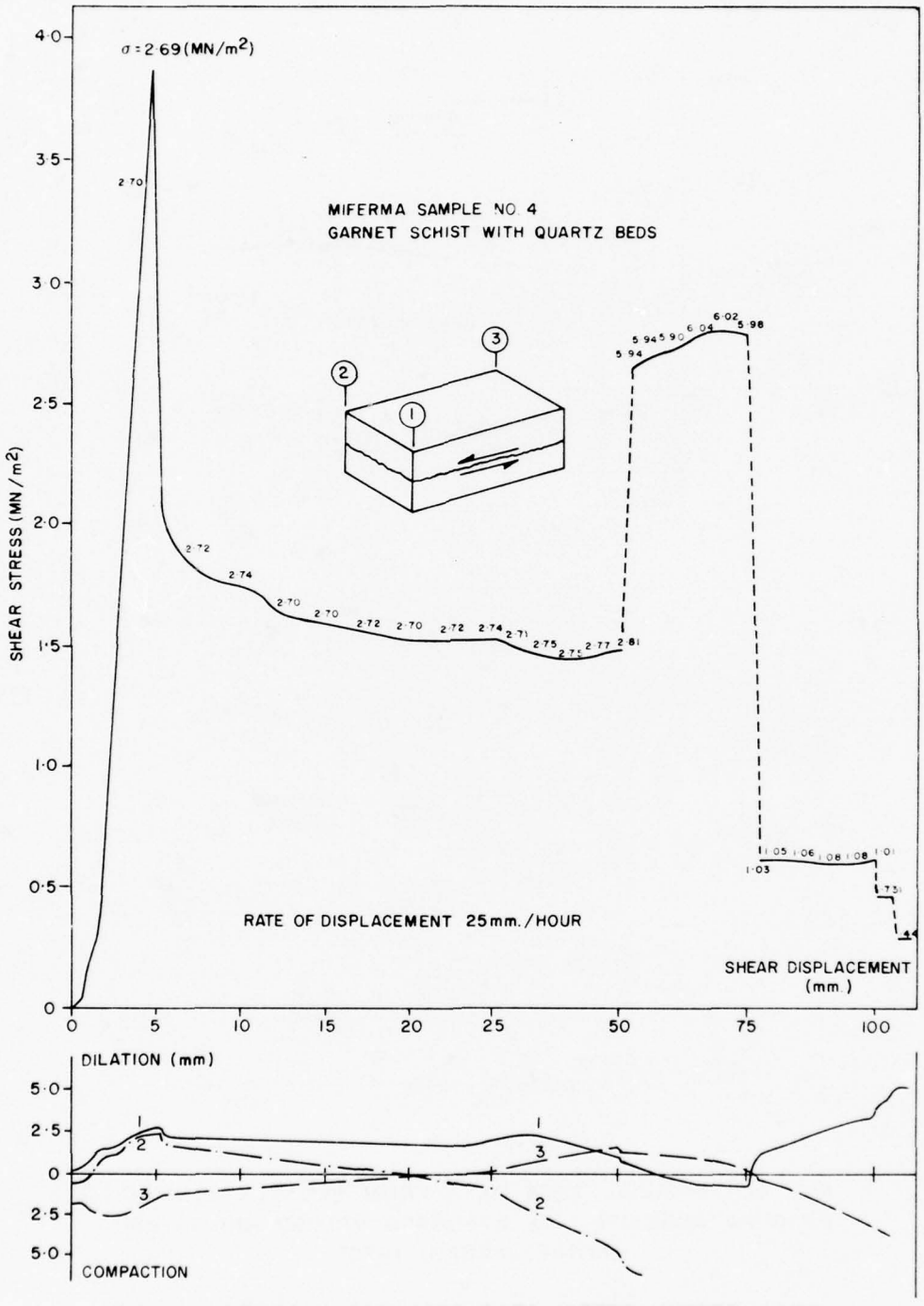
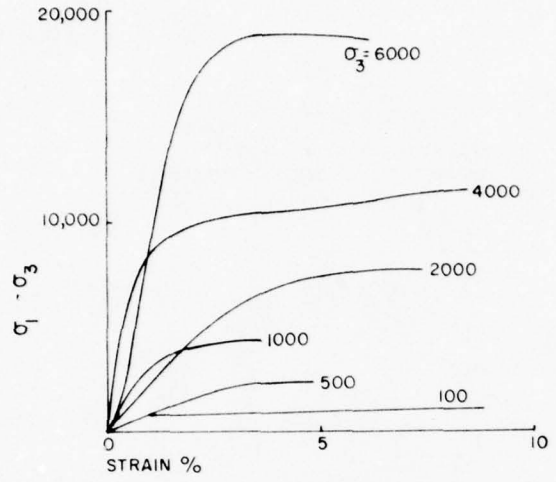


FIG. 5.2 DIRECT SHEAR TEST DATA (AFTER KUTTER, 1974)
NOTE CHANGES IN NORMAL LOAD DURING THE TEST

DAMES & MOORE



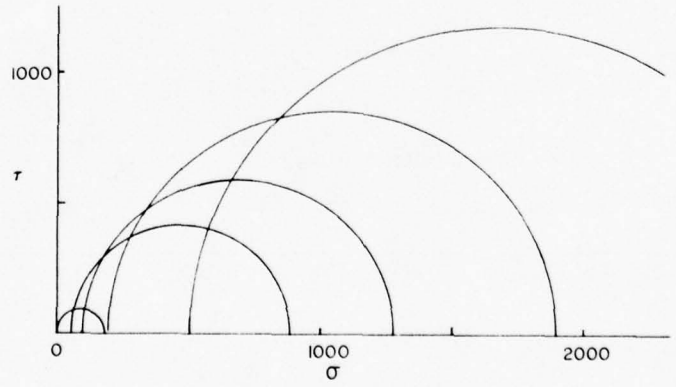


FIG. 5.3 TRIAXIAL TEST DATA FROM CLOSELY JOINTED PANGUNA ANDESITE. ALL STRESSES VALUED ARE IN PSI (AFTER JAEGER, 1970)

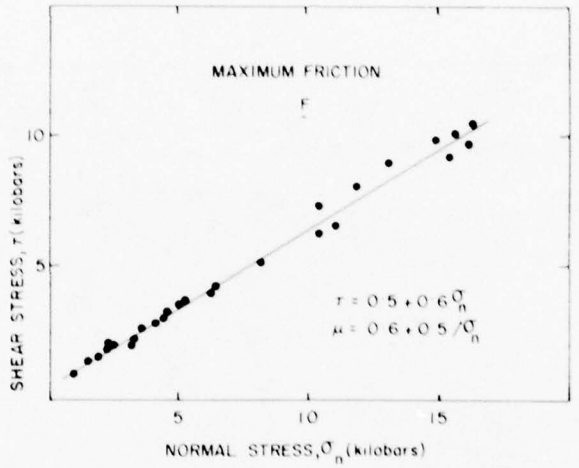


FIG. 5.4A SHEAR STRESS VERSUS NORMAL STRESS FOR FRACTURED GRANITE SPECIMENS FAILED IN TRIAXIAL COMPRESSION TESTS (AFTER BYERLEE, 1967)

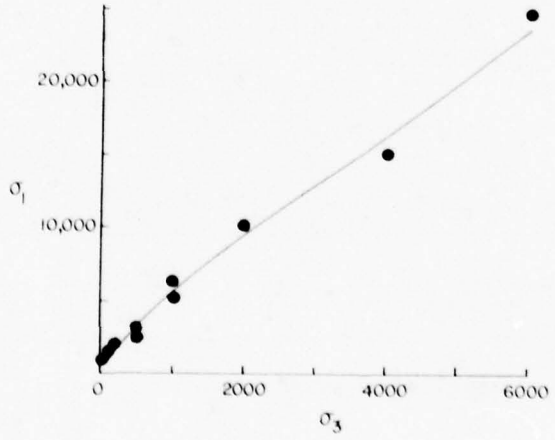


FIG. 5.4B SHEAR STRESS VERSUS NORMAL STRESS FOR 6 INCH CORES OF CLOSELY JOINTED PANGUNA ANDESITE (AFTER JAEGER, 1970)

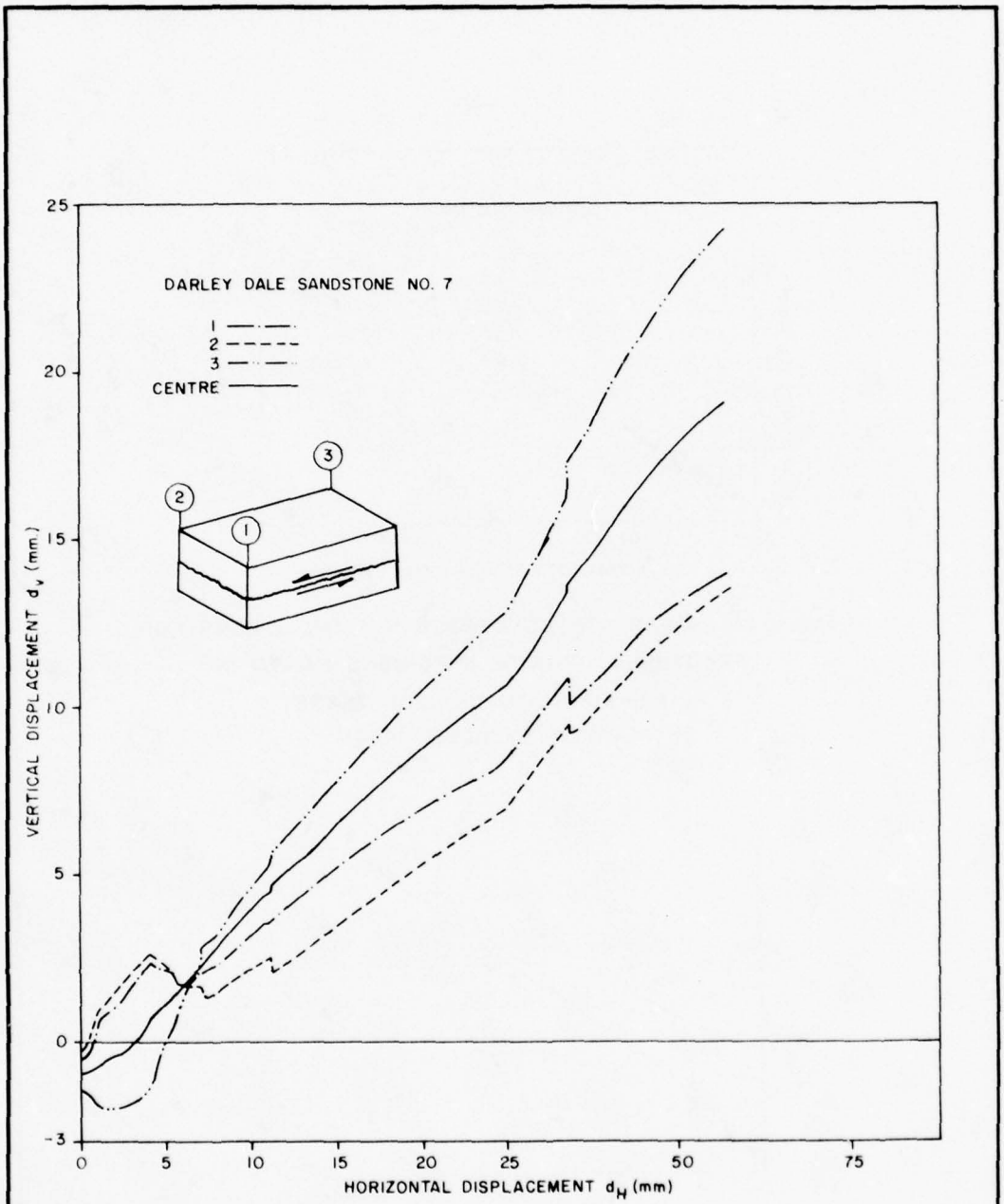


FIG. 5.5 DILATION CURVES FOR DARLEY DALE SANDSTONE (AFTER KUTTER, 1974)

DAMES & MOORE

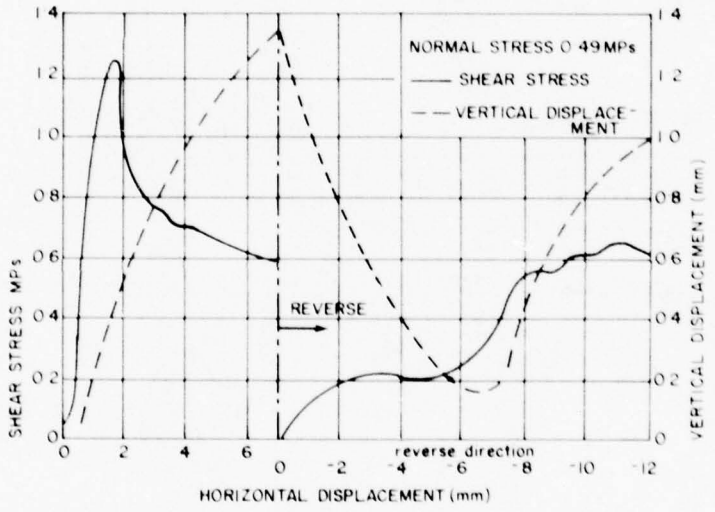


FIG. 5.6 DIRECT SHEAR DATA ON WEATHERED GREYWACKE
(AFTER MARTIN AND MILLAR, 1974)
NOTE THE CHANGE IN THE DILATION UPON REVERSAL
OF THE SHEARING DIRECTION

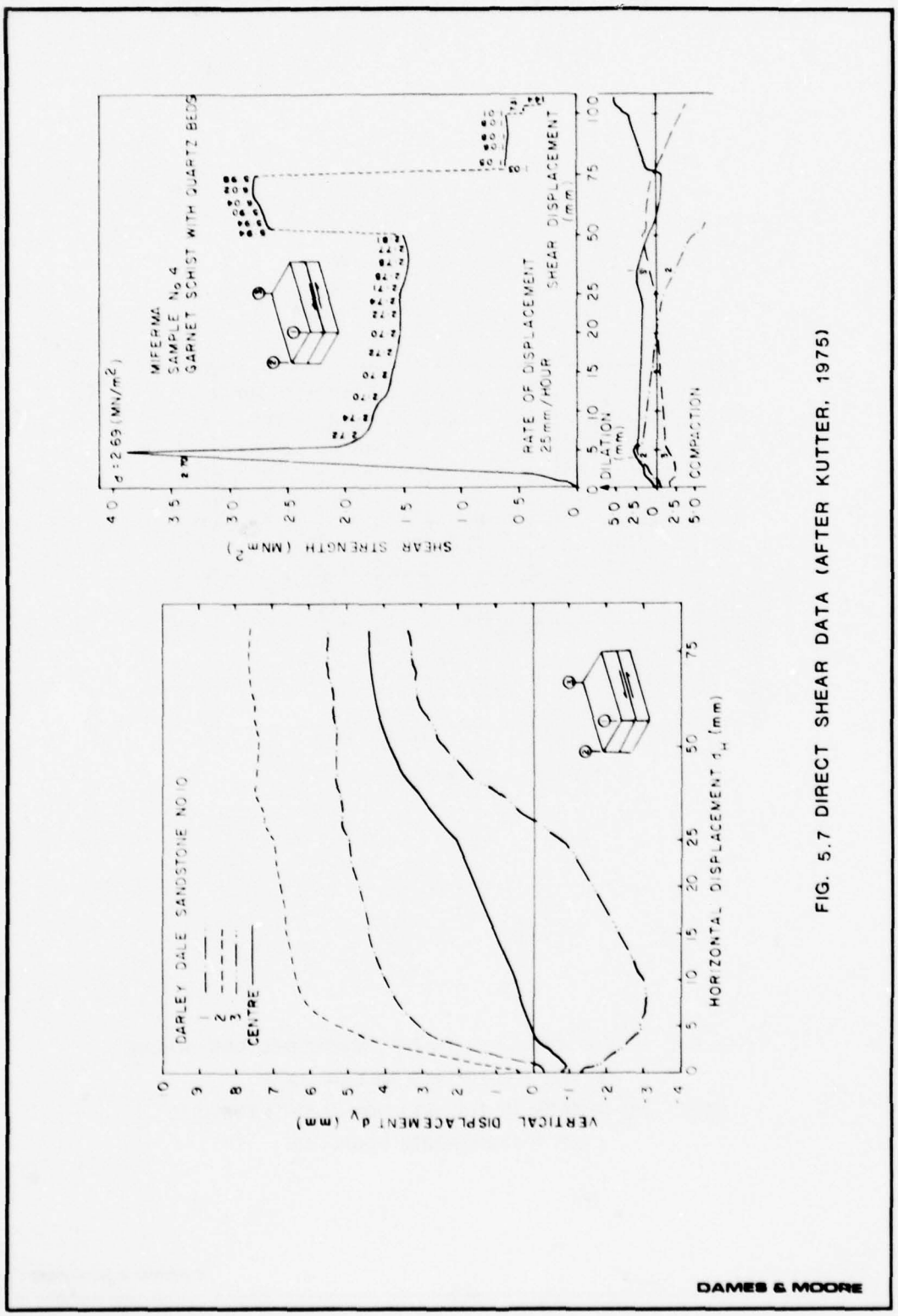


FIG. 5.7 DIRECT SHEAR DATA (AFTER KUTTER, 1975)

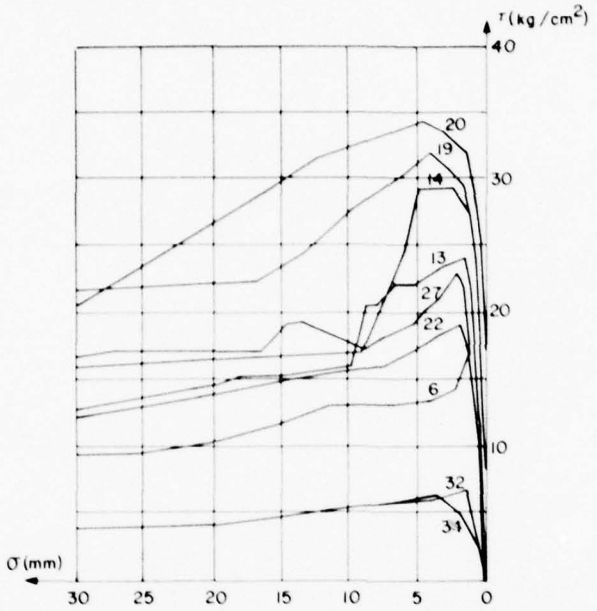


FIG. 5.8A DIRECT SHEAR DATA ON FISSURED LIMESTONE.
WITH NO FILLING MATERIAL
(AFTER KRSMANOVIC, TUTO AND LANGOF, 1966)

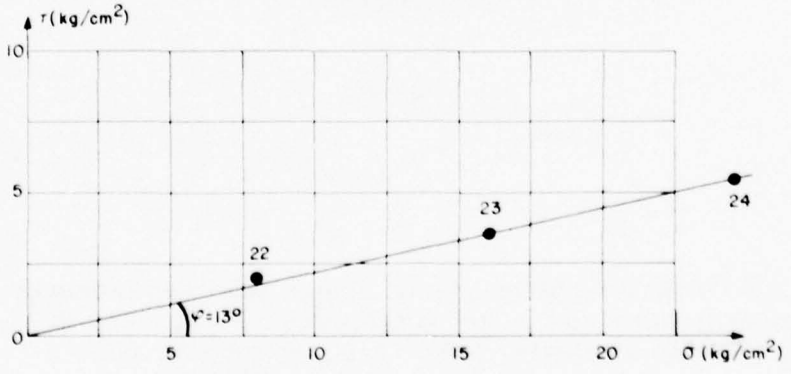


FIG. 5.8B DIRECT SHEAR DATA ON FISSURED LIMESTONE.
WITH CLAYEY INFILLING MATERIAL
(AFTER KRSMANOVIC, TUTO AND LANGOF, 1966)

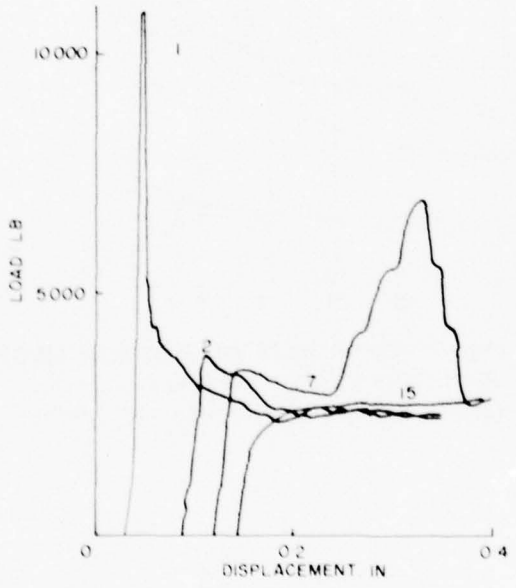


FIG. 5.9 VARIATION OF FRICTIONAL FORCE WITH DISPLACEMENT AND WITH WEAR FOR A SURFACE OF TENSILE FRACTURE IN BOWRAL TRACHYTE: AREA 5.2 SQ. IN.: NORMAL LOAD 1350 LB. NUMBERS ON CURVES INDICATE THE TEST NUMBER (AFTER JAEGER, 1971)

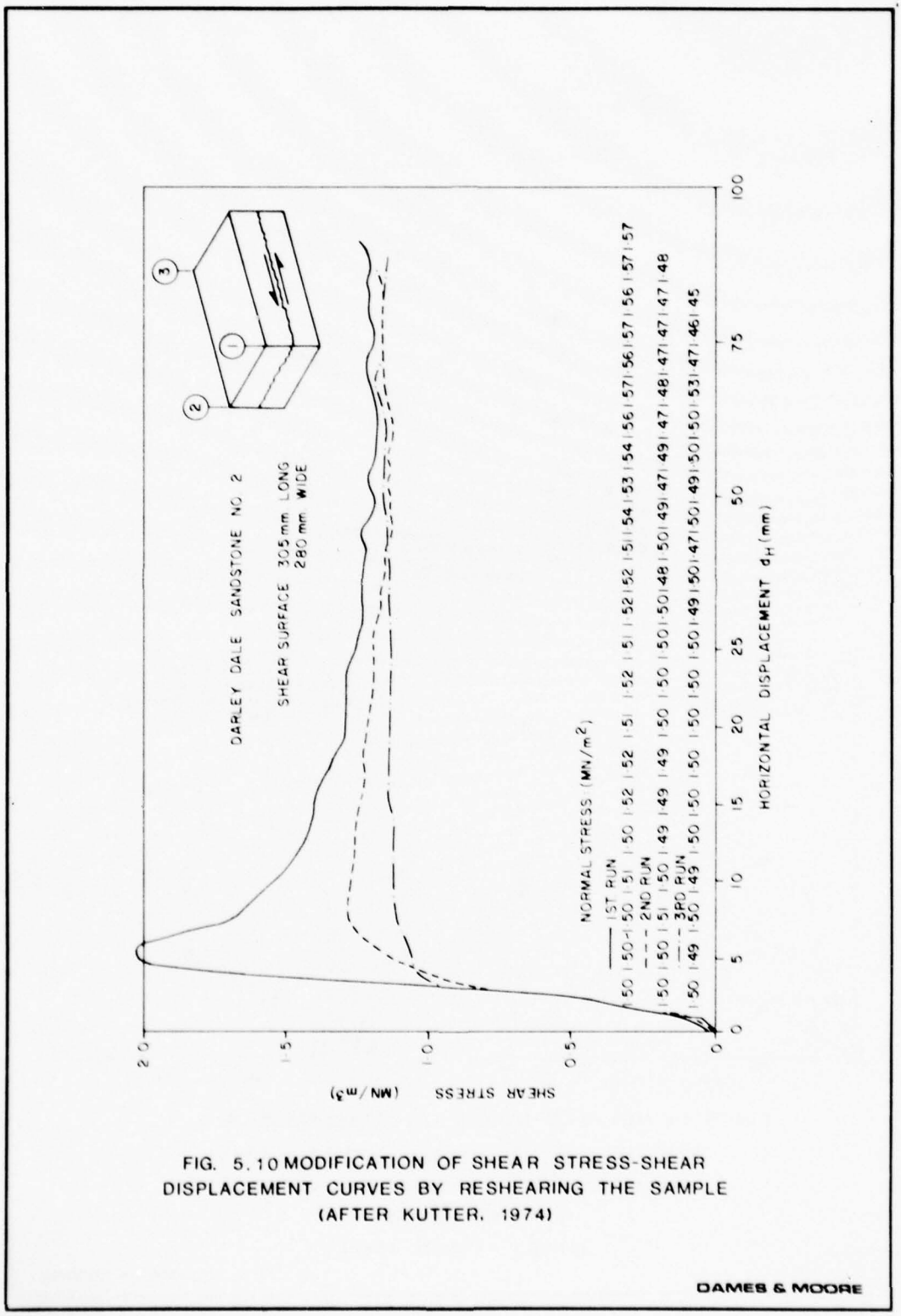


FIG. 5.10 MODIFICATION OF SHEAR STRESS-SHEAR DISPLACEMENT CURVES BY RESHEARING THE SAMPLE (AFTER KUTTER, 1974)

DAMES & MOORE

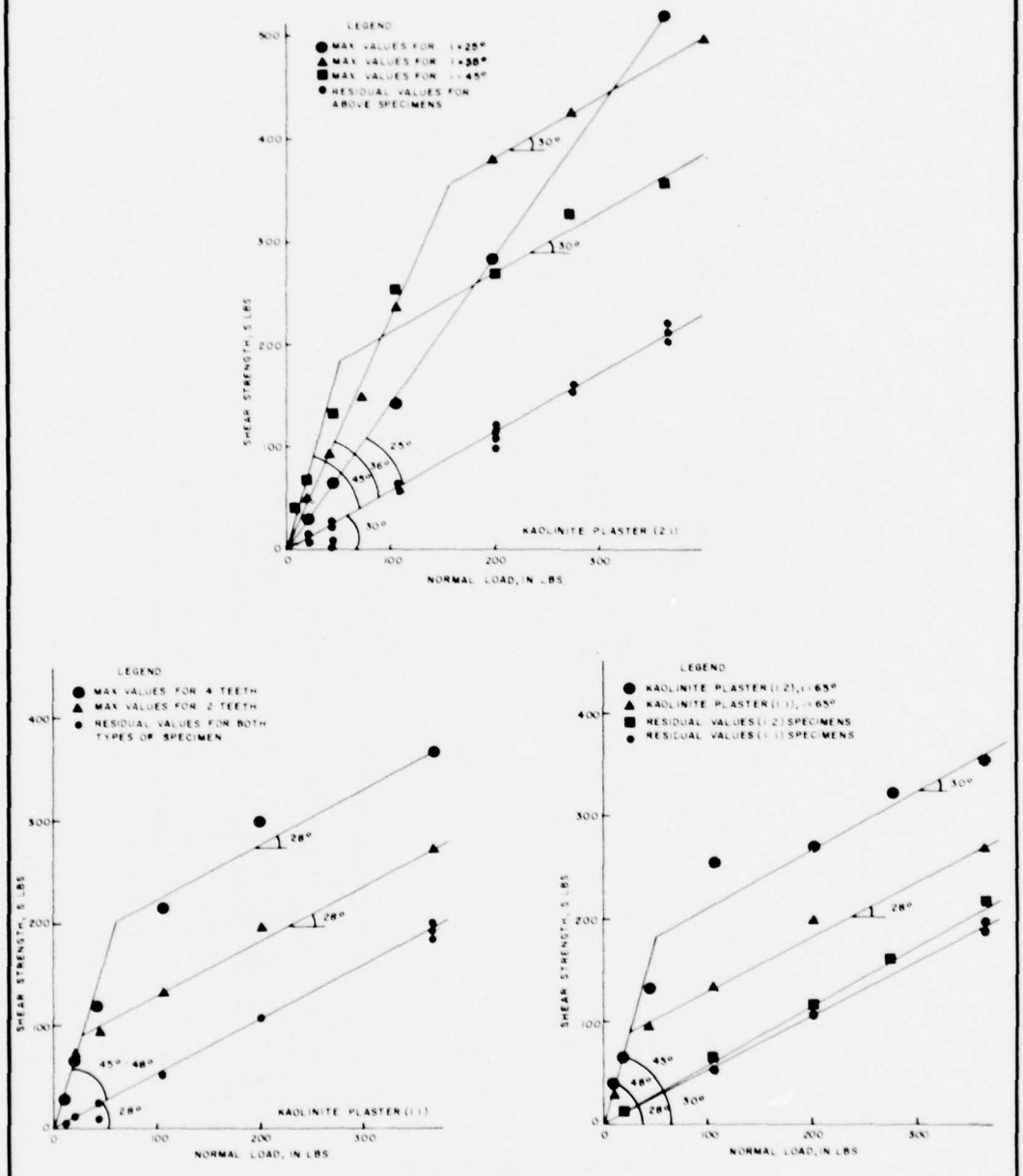


FIG. 5-11 MULTIPLE MODES OF SHEAR FAILURE IN PLASTER MODELS SHEARED ACROSS PLANES HAVING TEETH OF VARIOUS INCLINATIONS AND FREQUENCIES (AFTER PATTON, 1966)

DAMES & MOORE

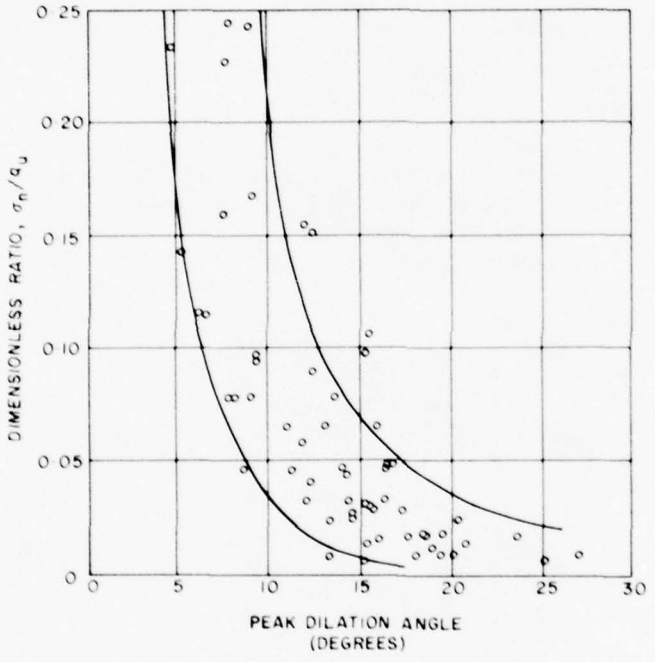


FIG. 5.12 PEAK DILATANCY ANGLE (i) AS A FUNCTION OF THE RATIO OF NORMAL STRESS TO COMPRESSIVE STRENGTH FOR MODEL EXTENSION JOINTS (AFTER BARTON, 1971)

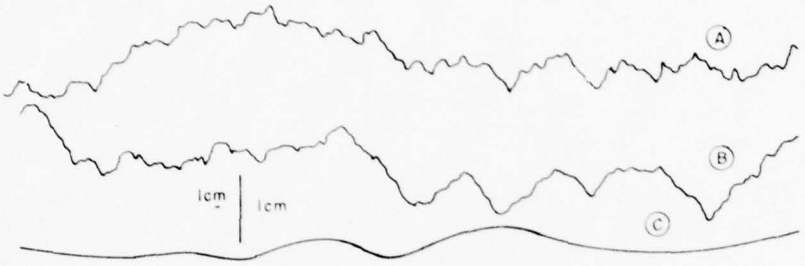


FIG. 5.13A PROFILES OF SURFACE GEOMETRIES OF JOINTS CAUSED BY TENSION IN GRANITE (A) SANDSTONE (B) AND LIMESTONE (C) (AFTER SCHNEIDER, 1974)

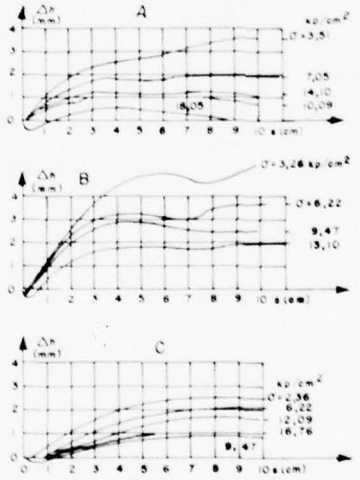


FIG. 5.13B DILATION CURVES OF THE FRICTION MODEL TESTS WITH PLASTER CASTS OF GRANITE (A) SANDSTONE (B) AND LIMESTONE (C) SURFACE GEOMETRIES (AFTER SCHNEIDER, 1974)

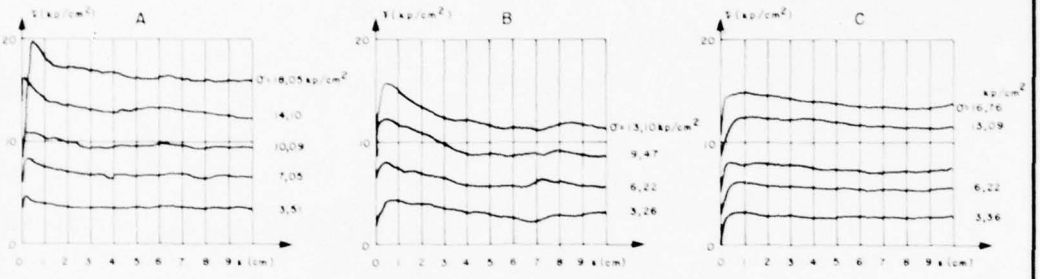


FIG. 5.13C STRESS STRAIN CURVES OF FRICTION TESTS ON MODELS WITH PLASTER CASTS OF GRANITE (A) SANDSTONE (B) AND LIMESTONE (C) JOINT FRAGMENTS (AFTER SCHNEIDER, 1974)

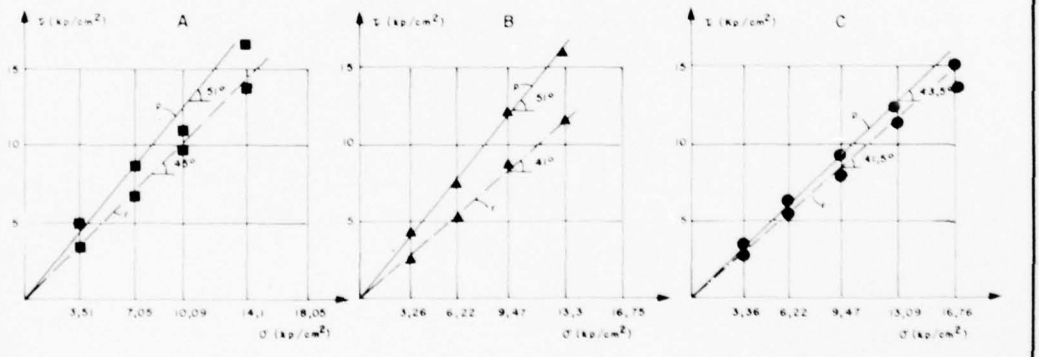


FIG. 5.13D SHEAR STRESS - NORMAL STRESS DIAGRAMS OF FRICTION TESTS ON MODELS WITH SURFACE GEOMETRIES OF GRANITE (A) SANDSTONE (B) AND LIMESTONE (C) (AFTER SCHNEIDER, 1974)

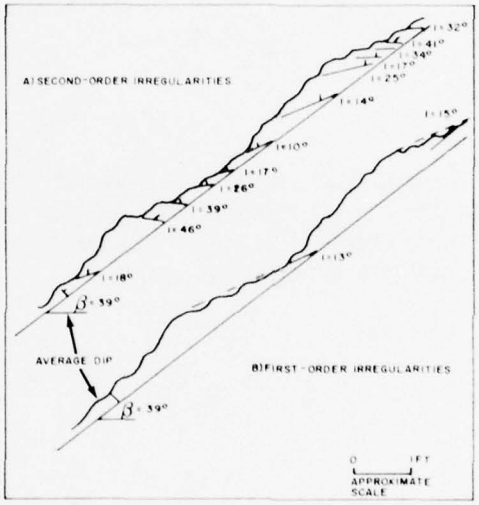


FIG. 5.14 FIRST AND SECOND ORDER IRREGULARITIES
(AFTER PATTON, 1966)

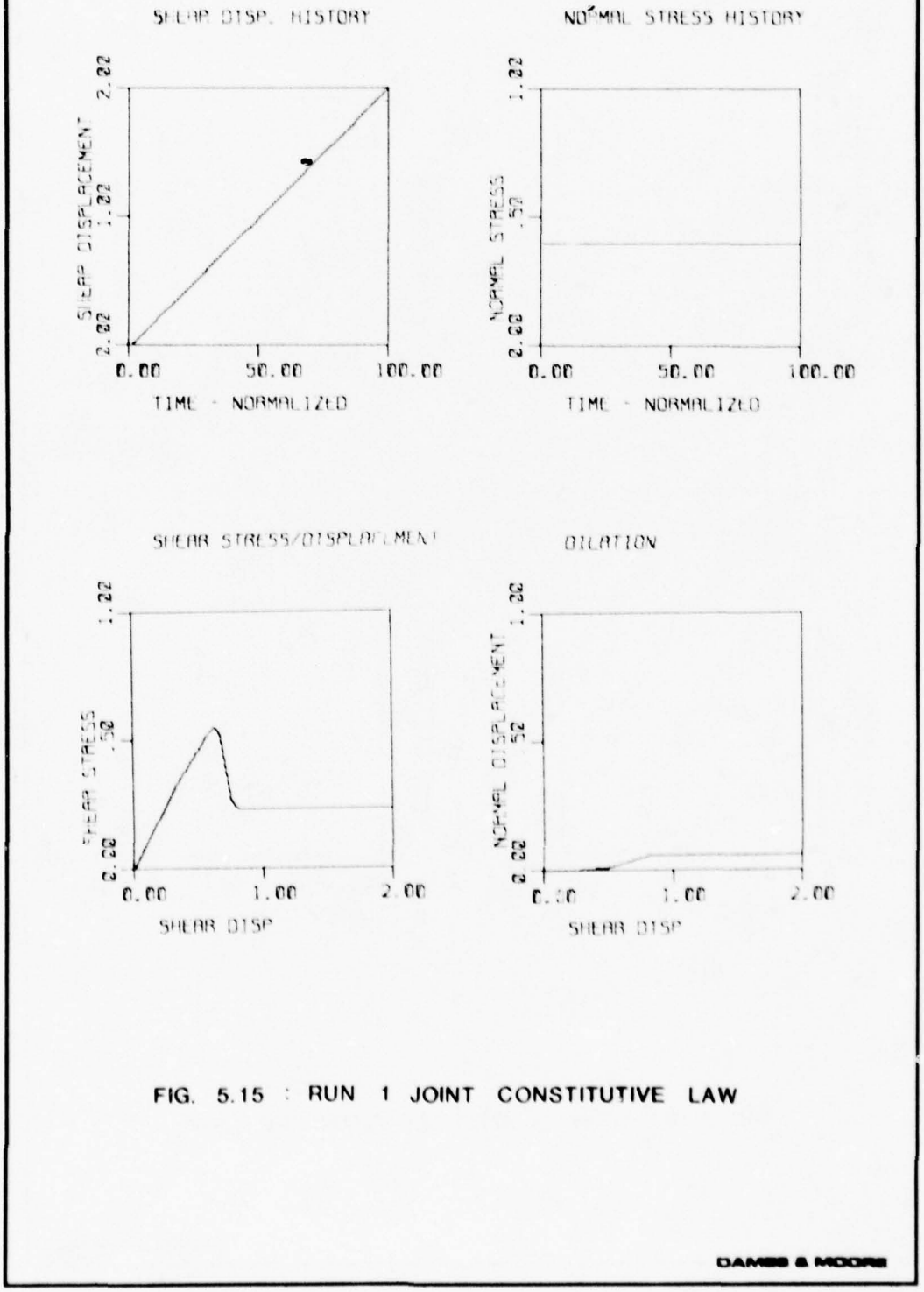


FIG. 5.15 : RUN 1 JOINT CONSTITUTIVE LAW

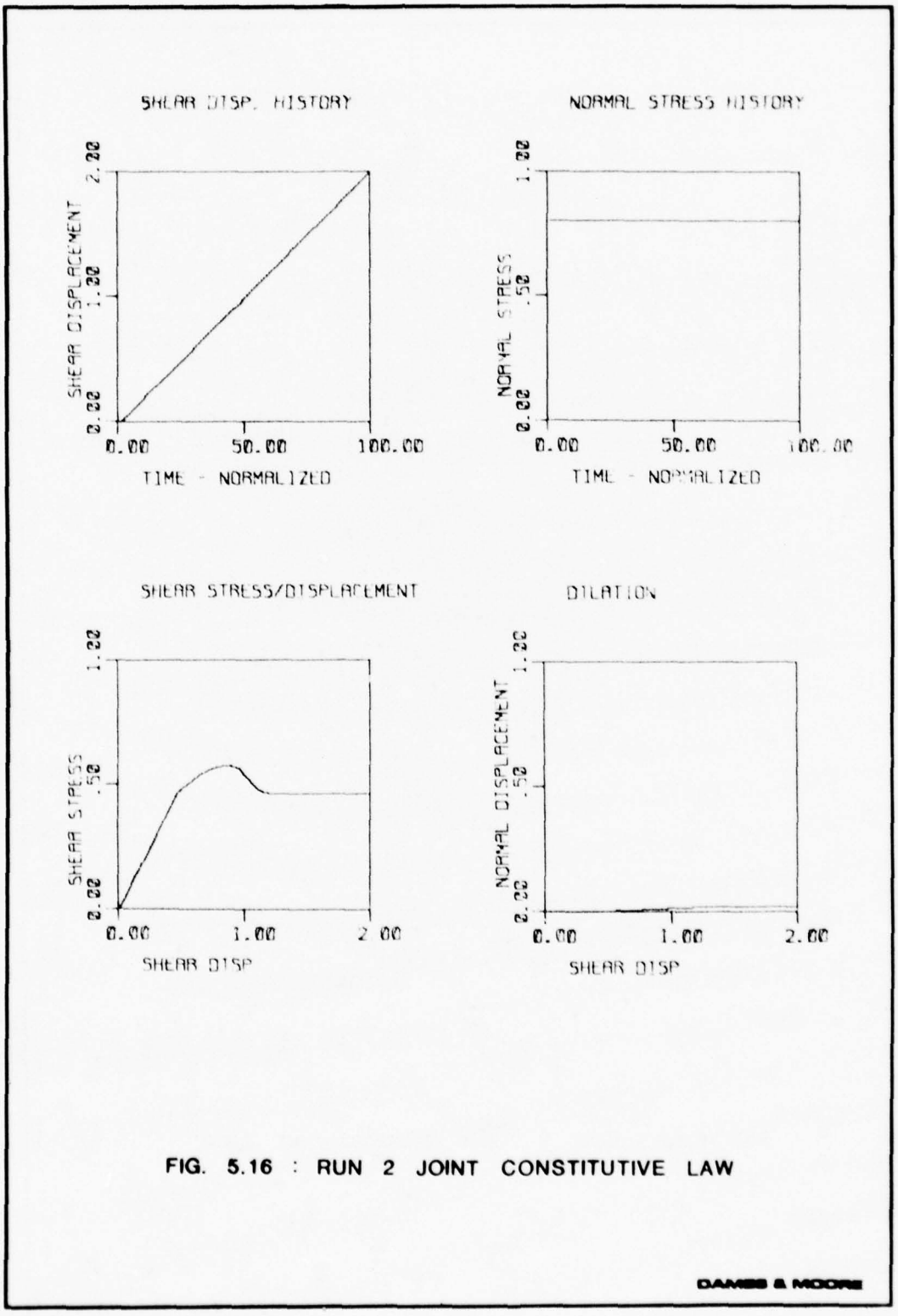


FIG. 5.16 : RUN 2 JOINT CONSTITUTIVE LAW

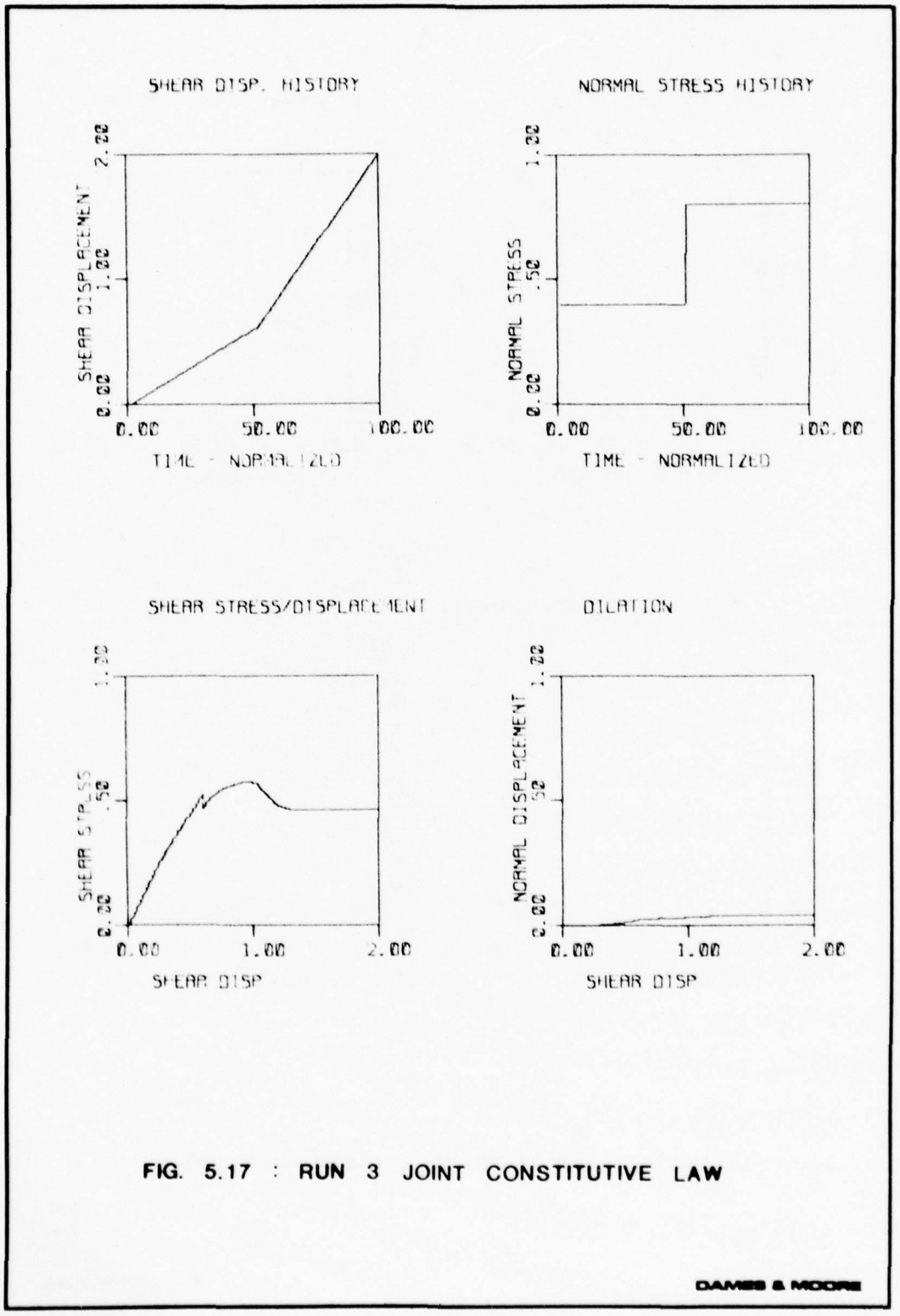


FIG. 5.17 : RUN 3 JOINT CONSTITUTIVE LAW

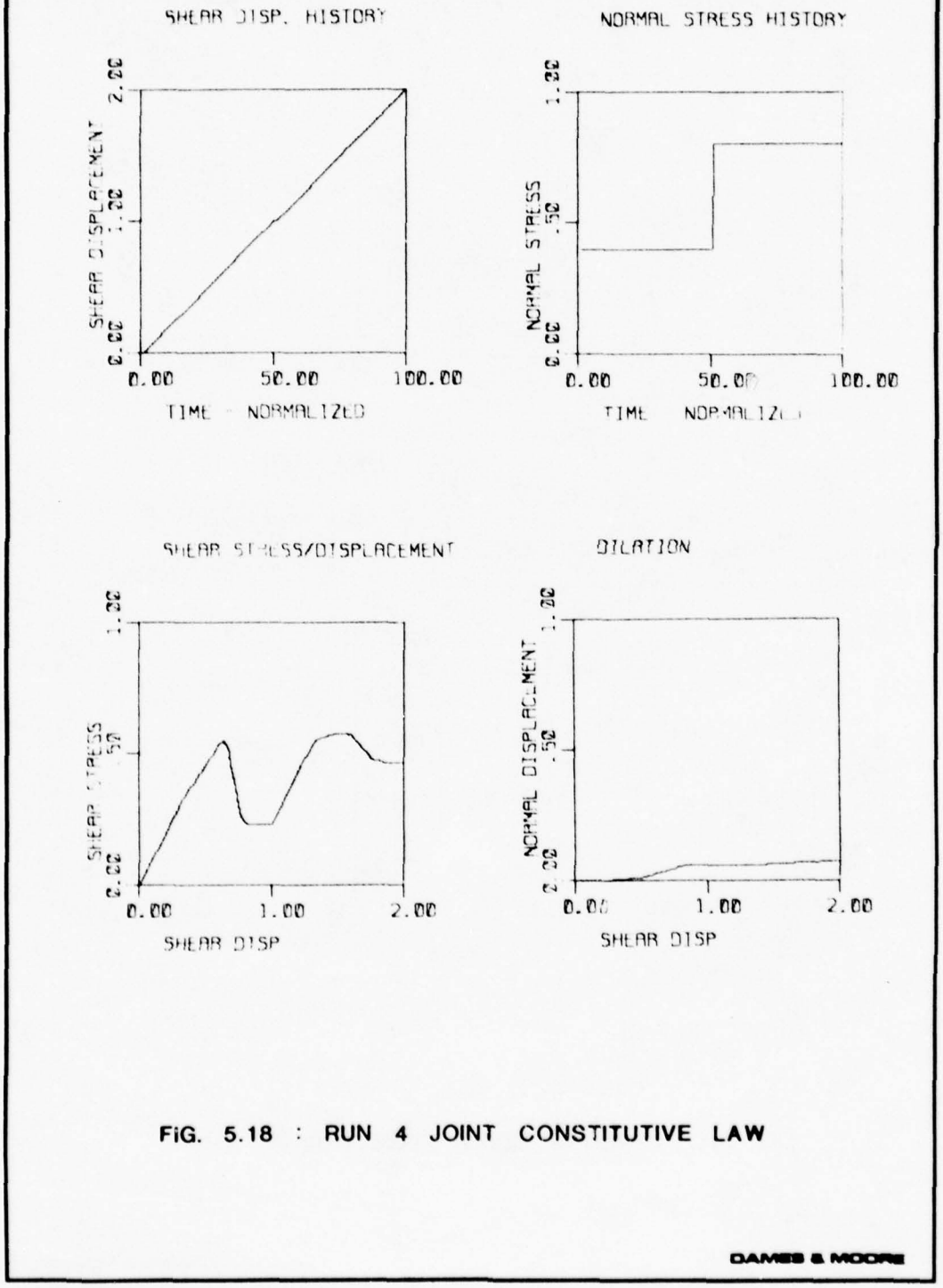


FIG. 5.18 : RUN 4 JOINT CONSTITUTIVE LAW

DAMES & MOORE

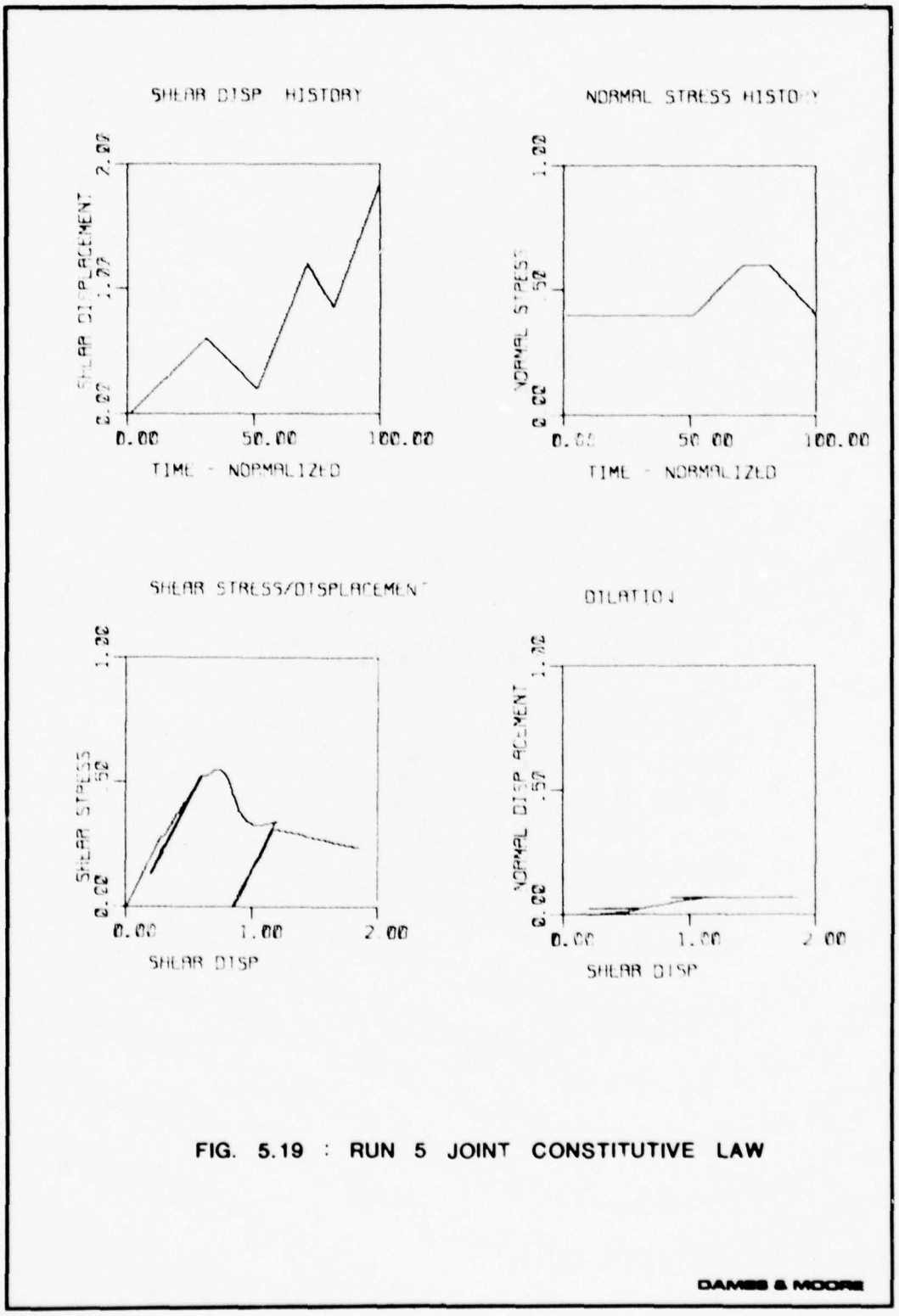


FIG. 5.19 : RUN 5 JOINT CONSTITUTIVE LAW

DAMES & MOORE

CHAPTER 6: FULLY-DEFORMABLE BLOCKS

6.0 A new approach is described that could model accurately both continua as well as discontinua. The method is assessed with the aid of a simple "test-bed" program. Several examples and validations are presented.

6.1 INTRODUCTION

6.1.1 GENERAL APPROACH

The method presented in this chapter is very general in that it can be applied to problems ranging from deformable continua to rigid or deformable discontinua. Computer programs dealing with deformable continua or rigid discontinua were already commercially available at the time of undertaking this project, but the generality of programs for analysis of fully deformable discontinua was very limited. The object of the present work was to investigate a method general enough to handle continua and discontinua as particular cases, but especially oriented to the solution of problems concerning deformable discontinua. In particular, the resulting computer program should be able to examine the interaction of arbitrary arrays of blocks with arbitrary laws governing the behaviour of the intact material and the interaction across discontinuities.

However, it must be made clear that the program described in this Chapter is not an end product. It is rather the result of a first phase of work. The purpose of this phase was the validation of the numerical approaches proposed for handling each of the problems arising in the analysis of deformable discontinua. The different assumptions are individually examined and validated in the following sections. Consequently, the program is not general in the sense that it is not provided with sophisticated input/output capabilities, and no special effort has yet been performed to optimise the data structure and the coding in order to reduce the computer time to a minimum. The program constitutes the "test-bed" for what is intended to become in the future a general user-oriented program.

The first application envisaged for the program will be the anaylsis of jointed rock masses at large depths subjected to high transient loads. By large depth it is understood here to be the depth at which the behaviour of a rock mass cannot be determined from the knowledge of its discontinuities alone. That is to say, the magnitude of the confining pressure or transient loads are so high that displacements arising from deformation of intact rock blocks are not negligible compared to those generated by rigid body movements of the blocks. This situation appears when plastification of the intact rock is likely to occur; the minimum depth required for such phenomenon is a direct function of the mechanical characteristics of the intact rock material.

6.1.2 SUMMARY OF THE METHOD

The system represented by the program is that of a set of deformable blocks. Each block is decomposed into constant-strain triangular elements and modelled as a continuum.

The continuum part of the program is a two-dimensional, finite-difference, explicit, large-deformation, Lagrangian code using triangular zones, and is based on the work of Wilkins (1969). Differences are central in time and space, and plane strain conditions are assumed. The general logic is the usual: in each time step, a) stresses are integrated about grid points to provide forces and hence accelerations, velocities and displacements; b) strains are obtained from displacements, and stresses from strains, by using the constitutive relations.

The novelty of the program lies in the treatment of contacts between blocks. When a node N from a given block contacts any other block, a new node N' is created at the contact point and the contacted triangle is decomposed into two. This procedure allows the use of practically the same treatment for contacts and continua, since in both cases a constitutive law is used to link grid-points. The only difference is that three grid-points are used for a continuum element, while two grid-points delimit a joint, with the constitutive law for the joint in between. As the node N slides on the surface of the other block, the node N' created at the contact is repositioned to follow N; in this manner, the interactive forces are always transmitted through mass points and felt as accelerations. At present the force-displacement relations for the contact are a bilinear/rigid law in the normal direction and an elastoplastic one in the tangential direction. However, these can be easily substituted in the program by any other explicit laws simply by adding a routine that embodies the desired constitutive laws. More specific details for all of the above schemes will be found in the next Sections 6.2 and 6.3.

The constitutive equations used at present for the continuum are those of an elastic material, although elastoplastic laws with strain hardening only require the addition of a few statements at the place indicated in the program.

A feature of the method described above is that both a true continuum and a true discontinuum can be accommodated as special cases. For a continuum each pair of mass-points forming a joint may be locked together rigidly, leaving the triangular zones free to deform. At the other extreme, the three mass-points of each triangle can be constrained to move together, with the joints alone being capable of deformation.

6.1.3 LAYOUT OF CHAPTER 6

Section 6.2 deals with the numerical treatment of the basic equations involved for the case of triangular zones. Mass lumping and numerical stability are also examined.

Section 6.3 is concerned with the initial discretization of the continuum and the subsequent rezoning associated with the contacts between blocks. Conservation of mass, momentum and energy for such processes is thoroughly examined.

The general organization of the program and a summarised description of each routine is provided in Appendix X and a Users' guide given in Section 6.4. Section 6.5 presents a series of examples which are intended both as checks on the program and as orientation for further use.

Conclusions and recommendations for further work are listed in Section 6.6.

The appendices VIII, IX and XVI restrict themselves to definitions of symbols and variables used, some necessary properties of triangles and a compiled listing of the program as it presently stands.

6.2 BASIC EQUATIONS AND PARAMETERS

6.2.1 CONSTITUTIVE RELATIONS

The constitutive relations are used in an incremental form, so that implementation on non-linear problems can be accomplished easily. As present, the program has been provided only with an elastic law. The place where the failure criterion should be included for treatment of elastoplastic problems with strain hardening and associated or non-associated flow rules is indicated in the listing.

The actual form of the equations used is:

$$\Delta\tau^{e}_{ij} = \lambda\ \Delta\varepsilon_{v}\delta_{ij} + 2\ \mu\Delta\varepsilon_{ij} \quad (*)\ (**)$$

where λ, μ are the Lamé constants

$\Delta\tau^{e}_{ij}$ are the elastic increments of the stress tensor

$\Delta\varepsilon_{ij}$ are the incremental strains

$\Delta\varepsilon_{v} = \Delta\varepsilon_{11} + \Delta\varepsilon_{22}$ is the increment of volumetric strain

6.2.2 EQUATIONS OF MOTION

The meaning of the symbols used is illustrated in Figure 6.1.

(*) All symbols are defined in their first appearance and in Appendix VIII.

(**) i,j,k are used as indices describing tensorial character. The usual summation convention for repeated indices applies to the rest of Chapter 6.

The sum of forces at a node yields:

$$F_i^N = \int_p \tau_{ij} n_j \, ds$$

$$= \int_{p_1} \tau_{ij} n_j \, ds \text{ due to stresses being constant within zones}$$

$$= \sum_{M=1}^{N_M} \tau_{ij}^M \, \Delta s^M n_j^M$$

where F_i^N are the components of the resultant force on node N

p and p_1 are integration paths shown in Figure 6.1.

n_j are the components of the normal to the path.

s is length along the integration path.

M, ranging from 1 to N_M, covers all zones surrounding N.

In each direction:

$$F_x^N = \sum_M \tau_{xx}^M (y_2^M - y_1^M) - \sum_M \tau_{xy}^M (x_2^M - x_1^M)$$

$$F_y^N = \sum_M \tau_{xy}^M (y_2^M - y_1^M) - \sum_M \tau_{yy}^M (x_2^M - x_1^M)$$

where x,y are the coordinates and the positions of 1 and 2 in zone M as shown in Figure 6.1.

It is clear that:

$$x_2^M - x_1^M = \frac{x_N + x_E^M}{2} - \frac{X_N + x_S^M}{2}$$

$$= \frac{x_E^M - x_S^M}{2}$$

$$y_2^M - y_1^M = \frac{y_E^M - y_S^M}{2}$$

where the positions of E and S in zone M are shown in Figure 6.1.

Therefore:

$$F_x^N = \frac{1}{2} \sum_M \left[\tau_{xx}^M (y_E^M - y_S^M) - \tau_{xy}^M (x_E^M - x_S^M) \right]$$

$$F_Y^N = \frac{1}{2} \sum_M \left[\tau_{xy}^M (y_E^M - y_S^M) - \tau_{yy}^M (x_E^M - x_S^M) \right]$$

Note that these formulae are independent of the order of summation through the surrounding zones, as long as all of them are included. Also, the zones need not surround completely the node considered.

6.2.3 STRAIN-DISPLACEMENT RELATIONS

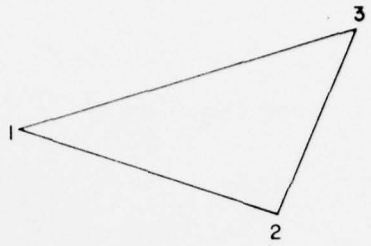
These are obtained in the following manner:

$$\dot{\epsilon}_{xx} = \frac{\partial \dot{u}_x}{\partial x}$$

$$\dot{\epsilon}_{yy} = \frac{\partial \dot{u}_y}{\partial y}$$

$$\dot{\varepsilon}_{xy} = \frac{1}{2}\left[\frac{\partial \dot{u}_x}{\partial y} + \frac{\partial \dot{u}_y}{\partial x}\right]$$

$$\dot{R}_{xy} = \frac{1}{2}\left[\frac{\partial \dot{u}_x}{\partial y} - \frac{\partial \dot{u}_y}{\partial x}\right]$$



where u_i is the displacement tensor

R_{ij} is the rotation tensor (positive clockwise)

dots represent time derivatives

In constant strain elements:

$$\begin{Bmatrix} \dot{u}_x \\ \\ \dot{u}_y \end{Bmatrix} = \begin{bmatrix} \dot{\varepsilon}_{xx} & \dot{\varepsilon}_{xy} + \dot{R}_{xy} \\ \\ \dot{\varepsilon}_{xy} - \dot{R}_{xy} & \dot{\varepsilon}_{yy} \end{bmatrix} \begin{Bmatrix} x \\ \\ y \end{Bmatrix} + \begin{Bmatrix} \dot{u}_{xo} \\ \\ \dot{u}_{yo} \end{Bmatrix}$$

where $\dot{u}_{xo}$, $\dot{u}_{yo}$ correspond to a rigid body movement.

Referring all coordinates and velocity components to those of node 1 in the zone:

$x_2 = x(2) - x(1)$ at time $t = (n + \frac{1}{2})\,\Delta t$, etc.

$\dot{u}_{x2} = \dot{u}_x(2) - \dot{u}_x(1)$ at time $t = n\Delta t$, etc.

Then:

$$\begin{bmatrix} \dot{u}_{x2} & \dot{u}_{x3} \\ \dot{u}_{y2} & \dot{u}_{y3} \end{bmatrix} = \begin{bmatrix} \dot{\epsilon}_{xx} & \dot{\epsilon}_{xy} + \dot{R}_{xy} \\ \dot{\epsilon}_{xy} - \dot{R}_{xy} & \dot{\epsilon}_{yy} \end{bmatrix} \begin{bmatrix} x_2 & x_3 \\ y_2 & y_3 \end{bmatrix}$$

$$\dot{\epsilon}_{xx} = (\dot{u}_{x2} y_3 - \dot{u}_{x3} y_2) \ / \ \text{DET}$$

$$\dot{\epsilon}_{yy} = (-\dot{u}_{y2} x_3 + \dot{u}_{y3} x_2) \ / \ \text{DET}$$

$$\dot{\epsilon}_{xy} = \tfrac{1}{2}(-\dot{u}_{x2} x_3 + \dot{u}_{x3} x_2 + \dot{u}_{y2} y_3 - \dot{u}_{y3} y_2) \ / \ \text{DET}$$

$$\dot{R}_{xy} = \tfrac{1}{2}(-\dot{u}_{x2} x_3 + \dot{u}_{x3} x_2 - \dot{u}_{y2} y_3 + \dot{u}_{y3} y_2) \ / \ \text{DET}$$

where DET = $x_2 y_3 - y_2 x_3 \neq 0$ unless 1,2,3 are aligned.

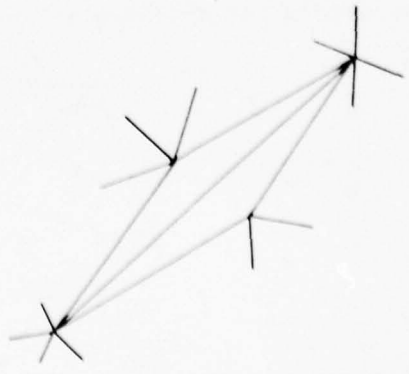
6.2.4 COMPUTATION OF THE TIME STEP

Numerical stability requires that no information be transmitted from a node to another node or edge in less than one time step. The maximum velocity of propagation is that of p-waves, which have the following velocity:

$$v_p = \sqrt{\frac{\lambda + 2\mu}{\rho}}$$

where λ, μ are the Lamé constants

ρ is the mass density



The time step is computed as:

$$\Delta t = \min_{(m,n)} \left(\frac{d(mn)}{v_p(mn)} \right)$$

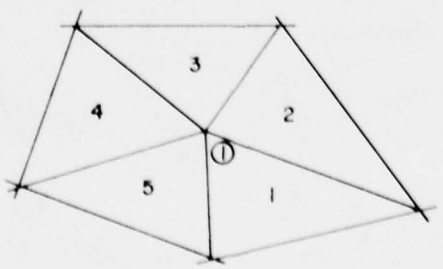
where d(mn) is the distance between two corners m and n of the same traingle and $v_p(mn)$ is the maximum P-wave velocity possible between those two corners.

To account for situations such as the one shown in the sketch in which the minimum travel time may not occur between nodes of the same triangle, the above Δt result is multiplied by 2α (if $\alpha < 0.5$) where α is the minimum ratio of height to base for all triangles (minimum aspect ratio).

In order to guarantee stability without recomputing the time step, even after the dimensions have been altered by deformations, the result is also multiplied by FRAC, where FRAC < 1 is fixed by the user. Note that damping could also require a decrease of the FRAC value since it increases the apparent stiffness of the material.

6.2.5 MASS LUMPING PROCEDURE

As justified in Appendix IX, each grid point is assigned one third of the mass of all surrounding triangles (lower case letters will be used for grid point masses and capitals for zone masses).



MASS LUMPING SCHEME

For example, for the case shown above:

$$m_1 = \frac{M_1 + M_2 + M_3 + M_4 + M_5}{3}$$

where m_1 is the mass of node 1

M_1, M_2 ... are the masses of zones 1,2,...

6.2.6 CONSTITUTIVE LAWS ACROSS CONTACTS

Two basic types of contacts have been assumed together with their corresponding class of equations governing the interaction at the contact. Both types are examined. It is proposed that the criterion for using one type rather than the other be based on the angle between the two edges involved in the contact: if the relative angle is less than a pre-set minimum, the edge-to-edge logic would be used.

6.2.6.1 Corner-to-edge Contacts

For convenience, the normal and tangential components of the interaction will be considered independently. The tangent plane at a surface grid point is taken parallel to the line linking the two adjacent grid points.

a) Normal Direction

The normal force - normal relative displacement law postulated is shown in Figure 6.2(a).

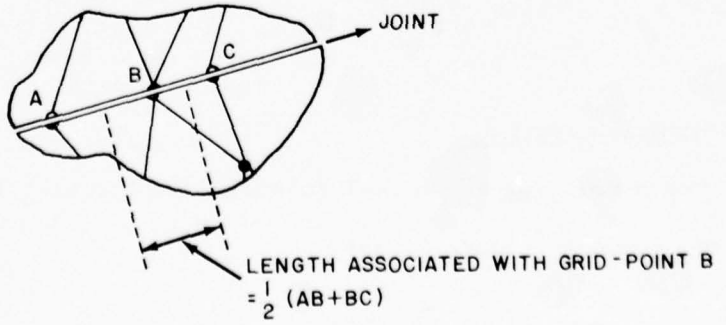
A bilinear law with no tension is assumed for relative displacements less than DN. When greater than DN, the contact is assumed rigid; i.e. the two mass-point move as one (see Section 6.2.6.3).

(b) Tangential Direction

The relationship between tangential force and relative displacement is shown in Figure 6.2(b), that is, an elastoplastic law with yield proportional to the normal component.

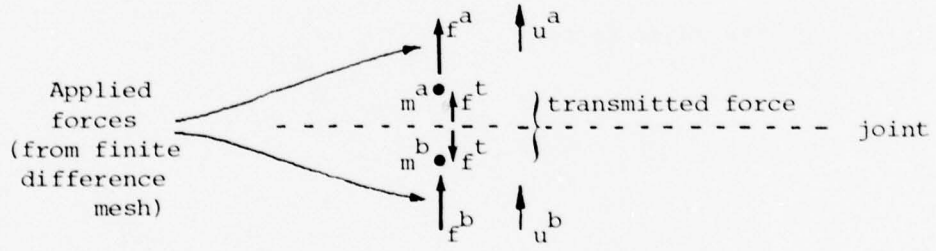
6.2.6.2 Edge-to-edge Contacts

Two edges in contact are regarded as a "joint", and the constitutive laws developed in Chapter 5 are used to derive the new normal and shear stresses from the given normal and shear displacements. Since the program needs grid-point forces, and not stresses, the stresses are multiplied by a length in order to convert them to forces. This length is assumed to be associated with the grid-point, and is taken to be the average of the distances of the grid-point from its two neighbours:



6.2.6.3 Flexible/Rigid Transition for Contacts

The normal stiffness of a rock joint characteristically increases with increased normal load. When the stiffness becomes great in comparison with that of the surrounding rock, the joint becomes "transparent" in the normal direction, and ceases to influence the mechanics of the rock mass. However, the time-step in an explicit scheme is forced to be uneconomically small by the high stiffness. For this reason program DBLOCK allows contacts to become rigid in the normal direction if the relative displacement across the joint exceeds a user-defined limit. It is still possible to evaluate the normal force between the locked grid-points, so that the calculations of slip in the shear direction can be performed as usual. The derivation of normal force is as follows:



$$m^a \ddot{u}^a = f^a + f^t$$

$$m^b \ddot{u}^b = f^b - f^t$$

If the two masses are locked together,

$$\ddot{u}^a = \ddot{u}^b$$

Hence $$\frac{f^a + f^t}{m^a} = \frac{f^b - f^t}{m^b}$$

$$\boxed{f^t = \frac{m^a f^b - m^b f^a}{m^a + m^b}}$$

This expression gives the normal force between the two opposing nodes, on the assumption that they are locked together in the normal direction. The normal force can then be utilized in the usual way by the joint constitutive law in order to derive the shear force. The transition back to flexibility in the normal direction can be determined from the magnitude of the normal force.

If a rigid/plastic law is needed for the shear direction, a similar approach may be taken in order to derive the shear force between nodes locked in the shear direction. In this case the contact is switched from the sliding state to the locked state when the relative shear velocity passes through zero. The transition in the reverse direction is determined by the magnitude of the shear force.

6.3 MESH GENERATION AND REZONING

6.3.1 AUTOMATIC MESH GENERATION

The program allows the user to supply his own mesh or to simply give the boundaries of the blocks, in which case the computer will generate automatically a triangular mesh of a given maximum edge length. This is achieved in three stages:

- The corners of the boundary are linked until the block is decomposed into triangles. The links are always established between the two closest corners. Although no holes are allowed, the number of blocks is only limited by the dimensions in COMMON and no special problems arise from locating several corners along straight or concave lines.

- The triangles are divided until all triangle edges are smaller than the length specified by the user. The edge divided is always that of maximum length in the triangle.

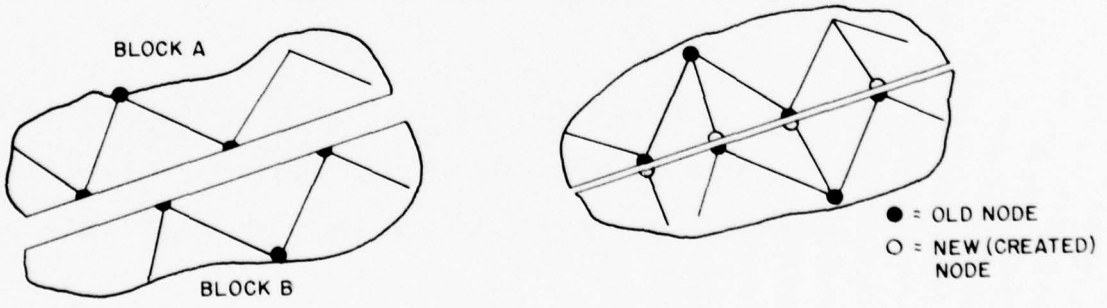
- All internal nodes are rezoned until their coordinates coincide with the average of the coordinates of the surrounding nodes. An iterative procedure is used here, since the relocation of one internal node will affect its neighbour.

In the light of the tests performed the above algorithms seem to be sufficient to provide satisfactory shapes of triangles in most instances. Note for example that it would be useless to define the block geometries with details which are small compared to the maximum triangle dimension specified by the user.

Examples of mesh generation for an odd-shaped block are given in Section 6.5.3.

6.3.2 CREATION OF A NEW NODE

When a node of a block contacts another block, a new node is created in the contacted block, in the same location as the impacting node:

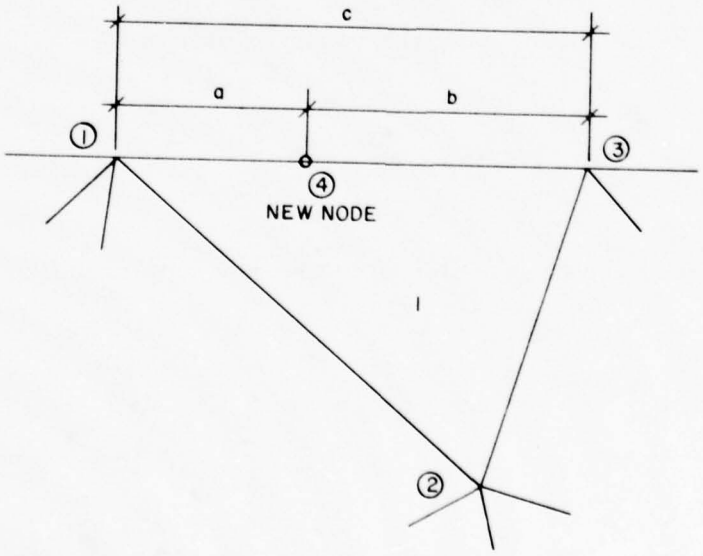


The advantage of this scheme is that it permits the natural transmission of momentum across blocks by means of forces of interaction which are felt as accelerations by the pair of nodes in contact. The creation of a new node implies a re-lumping of mass and momentum, and in

general, a change in the local energy since the momentum and energy equations cannot generally be simultaneously satisfied. In the procedures described below, mass and momentum are preserved and upper bounds in the energy increment are established proving that the errors introduced are negligible compared to other errors inherent in the discretization of the continuum.

6.3.2.1 Mass

As discussed in Appendix IX, masses are reassigned upon contact as follows (new masses indicated by an asterisk).



LET $\alpha = \frac{a}{c}$

$\beta = \frac{b}{c}$

Then:

$$m_1^* = m_1 - \frac{M_1}{3} + \frac{M_1}{3}\,\frac{a}{c} = m_1 - \beta\,\frac{M_1}{3}$$

$$m_1^* = m_2$$

$$m_1^* = m_3 - \alpha\,\frac{M_1}{3}$$

$$m_4^* = \frac{M_1}{3}$$

6.3.2.2 Momentum

In order to preserve momentum:

$$(m_1 - m_1^*)\,\bar{v}_1 + (m_3 - m_3^*)\,\bar{v}_3 = m_4^*\,\bar{v}_4$$

where $\bar{v}_N$ is the vector describing the velocity of node N

$$\frac{M_1}{3}\,\beta\bar{v}_1 + \frac{M_1}{3}\,\alpha\bar{v}_3 = \frac{M_1}{3}\,\bar{v}_4$$

Hence the velocity assigned to the new node must be:

$$\bar{v}_4 = \beta\bar{v}_1 + \alpha\bar{v}_3$$

6.3.2.3 Energy

The creation of a new grid point in the form described produces a loss of kinetic energy ΔK_E:

$$\Delta K_E = \frac{M_1}{3} \beta v_1^2 + \frac{M_1}{3} \alpha v_3^2 - \frac{M_1}{3} \left[\beta \bar{v}_1 + \alpha \bar{v}_3 \right]^2$$

$$= \frac{M_1}{3} \left[(\beta - \beta^2) v_1^2 + (\alpha - \alpha^2) v_3^2 - 2\alpha\beta \, \bar{v}_1 \bar{v}_3 \right]$$

$$= \frac{M_1}{3} \alpha\beta \left[v_1^2 + v_3^2 - 2\bar{v}_1 \bar{v}_3 \right]$$

$$= \frac{M_1}{3} \alpha\beta \quad (\bar{v}_1 - \bar{v}_3)^2$$

This result shows that:

a) energy is never gained

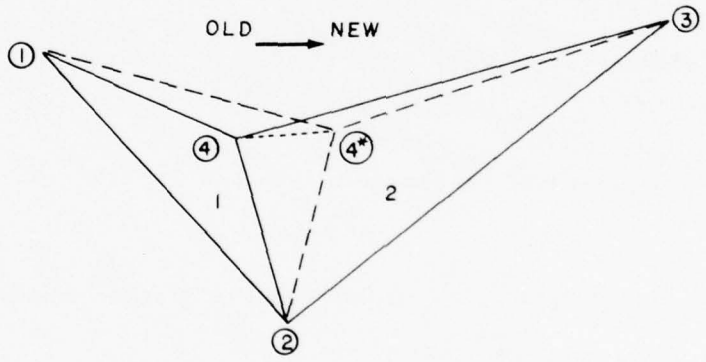
b) the energy loss goes to zero as the contact point approaches a grid point

c) the energy loss is proportional to the mass of the triangle

d) if velocities are continuous functions of the space coordinates, the energy loss goes to zero with the square of the zone size; that is, the error committed is of the same order of magnitude as the other errors associated with the discretization of the continuum.

6.3.3 "TICKLE" REZONING

Whenever the two contacting nodes separate tangentially further than TOL while maintaining normal contact, the created node is relocated. Its coordinates are made equal to those of the other node (allowing for the appropriate normal and tangential displacements corresponding to the interacting forces at that time). At present, TOL is obtained as a user-specified fraction of the distance between the two adjacent nodes.



TICKLING OF NODE 4

When the tangential distance between the two nodes in contact is less than TOL, the forces of interaction are transmitted as if the two nodes were precisely opposite each other. This introduces a moment error, which however can be made arbitrarily small as TOL is reduced.

The above sketch will be used as a guide to describe the characteristics of the tickle rezoning process.

6.3.3.1 Mass

The new zone and node masses are:

$$M_1^{\star} = M_1 + \Delta$$

Δ = mass (424*)

$$M_2^{\star} = M_2 - \Delta$$

$$m_1^{\star} = m_1 + \frac{\Delta}{3}$$

$$m_2^{\star} = m_2$$

$$m_3^{\star} = m_3 - \frac{\Delta}{3}$$

$$m_4^{\star} = m_4$$

6.3.3.2 Momentum

After "tickle" rezoning nodes 1, 2 and 3 maintain their velocities. Then the conservation of linear momentum yields:

$$\frac{\Delta}{3}\bar{v}_1 - \frac{\Delta}{3}\bar{v}_3 + m_4(\bar{v}_4^{\star} - \bar{v}_4) = 0$$

$$\bar{v}_4^{\star} = \bar{v}_4 + \frac{\Delta}{3m_4}(\bar{v}_3 - \bar{v}_1)$$

6.3.3.3 Energy

The energy loss associated with "tickle" rezoning is:

$$2\Delta E_K = m_1 v_1^2 + m_3 v_3^2 + m_4 v_4^2 - m_1^{\star} v_1^{\star 2} - m_3^{\star} v_3^{\star 2} - m_4^{\star} v_4^{\star 2}$$

$$= m_1 v_1^2 + m_3 v_3^2 + m_4 v_4^2 - (m_1 + \frac{\Delta}{3})\, v_1^2 - (m_3 - \frac{\Delta}{3})\, v_3^2$$

$$- m_4 \left[\bar{v}_4 + \frac{\Delta}{3m_4}(\bar{v}_3 - \bar{v}_1) \right]^2$$

$$\frac{2\Delta E_K}{m_4} = -\frac{m}{m_4} v_1^2 + \frac{m}{m_4} v_3^2 - 2mv_4 (\bar{v}_3 - \bar{v}_1) + m_4 \left[\frac{m}{m_4}\right]^2 \left[\bar{v}_3 - \bar{v}_1 \right]^2$$

where $m = \frac{\Delta}{3}$

Since $\frac{m}{m_4} << 1$, $\left[\frac{m}{m_4}\right]^2$ can be neglected. Then:

$$\frac{2\Delta E_K}{m} = v_3^2 - v_1^2 - 2\bar{v}_4 \quad (\bar{v}_3 - \bar{v}_1)$$

$$= (\bar{v}_3 - \bar{v}_1) \quad (\bar{v}_3 + \bar{v}_1) - 2\bar{v}_4 \quad (\bar{v}_3 - \bar{v}_1)$$

$$= (\bar{v}_3 - \bar{v}_1) \quad (\bar{v}_3 + \bar{v}_1 - 2\bar{v}_4)$$

$$\Delta E_K = m \quad (\bar{v}_3 - \bar{v}_1) \left[\frac{\bar{v}_1 + \bar{v}_3}{2} - \bar{v}_4\right]$$

which means that:

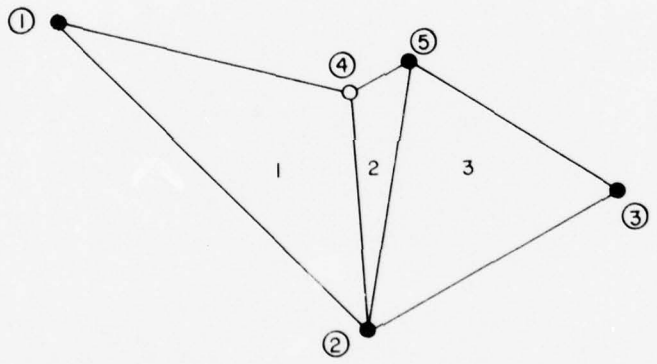
1) the energy loss is never negative (m changes sign with the dot product).

2) the energy loss is proportional to the mass transferred and goes to zero when the two outer corners (1,4) have equal velocities, or when their average is equal or perpendicular to the velocity of the created node (4).

3) as the mesh becomes smaller, the energy loss approaches zero as the product of the transferred mass times the square of the zone dimension.

6.3.4 DELETION OF TRIANGLES

Due to the tickle-rezoning accompanying sliding, one of the nodes created at a contact may approach another pre-existing node. When the zone common to those two nodes becomes too small, that zone is deleted. Following the nomenclature in the sketch below, zone 2 and node 4 will disappear when node 4 approaches node 5.



DELETION OF NODE 4

In this sketch, nodes 1, 4 and 5 are boundary nodes; 4 was created at a contact with another block.

The criterion used by the program to decide whether to delete a node or not is based on the aspect ratio of the diminishing triangle; specifically, the triangle is deleted when its aspect ratio becomes smaller than that needed to guarantee stability for the time increment used.

6.3.4.1 Mass

The zone masses become:

$$M_3^{\star} = M_3$$

$$M_1^{\star} = M_1 + M_2$$

And the node masses:

$$m_1^{\star} = m_1 + m_o$$

$$m_2^{\star} = m_2$$

$$m_3^{\star} = m_3$$

$$m_5^{\star} = m_4 + m_5 - m_o$$

where $m_o = \frac{M_2}{3}$

6.3.4.2 Momentum

All but node 5 preserve their previous velocities. Then, conservation of momentum implies:

$$m_1\bar{v}_1 + m_2\bar{v}_2 + m_3\bar{v}_3 + m_4\bar{v}_4 + m_5\bar{v}_5 = m_1^{\star}\bar{v}_1^{\star} + m_2^{\star}\bar{v}_2^{\star} + m_3^{\star}\bar{v}_3^{\star} + m_5^{\star}\bar{v}_5^{\star}$$

$$m_1\bar{v}_1 + m_4\bar{v}_4 + m_5\bar{v}_5 = (m_1 + m_o)\,\bar{v}_1 + (m_4 + m_5 - m_o)\,\bar{v}_5^{\star}$$

and hence:

$$\bar{v}_5^* = \frac{m_4\bar{v}_4 + m_5\bar{v}_5 - m_o\bar{v}_1}{m_4 + m_5 - m_o}$$

6.3.4.3 Energy

The loss in kinetic energy associated with this rezoning is:

$$2\Delta E_K = m_1 v_1^2 + m_2 v_2^2 + m_3 v_3^2 + m_4 v_4^2 + m_5 v_5^2 - m_1^* v_1^{*2} - m_2^* v_2^{*2} - m_3^* v_3^{*2} - m_4^* v_4^{*2}$$

$$= m_o v_1^2 + m_4 v_4^2 + m_5 v_5^2 - (m_4+m_5-m_o) \left[\frac{m_4\bar{v}_4 + m_5\bar{v}_5 - m_o\bar{v}_1}{m_4 + m_5 - m_o} \right]^2$$

By simple reorganization of terms, this can be expressed as:

$$2\Delta E_K = \frac{m_4 m_5 (\bar{v}_4-\bar{v}_5)^2}{m_4 + m_5 - m_o} - m_o \cdot \frac{m_4(\bar{v}_4-\bar{v}_1)^2 + m_5(\bar{v}_5-\bar{v}_1)^2}{m_4+m_5-m_o}$$

Assuming that velocities are continuous functions of the coordinates:

$$\Delta E_K = O(d^2) - O(d)O(e^2)$$

where d = distance from 4 to 5

e = zone typical dimension

O (') = ' of the order of'

Note that the first term represents an energy loss and the second represents a gain. Also that the errors associated with a constant strain triangular configuration are of the order of e^2. Since d is small compared to e, the error introduced by the rezoning described is negligible compared to that inherent in the method of discretization of the continuum.

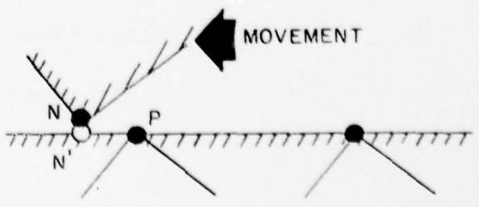
6.3.5 OTHER DETAILS

The main procedures for creation of the mesh and its rezoning have been described in earlier sections. To complete the description, some minor details should also be mentioned.

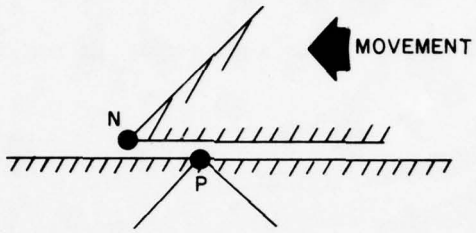
6.3.5.1 Recreation of Contact Nodes

It has been shown before how a node created at a contact is "tickled" to follow its partner; also, how the former is deleted when, due to successive "tickling" one of the triangles about the "tickled" node becomes very slender.

Consistent with those procedures, once the impacting node N goes past a certain distance from the opposite (non tickleable) node P, a new node N' is created to continue opposing N'. The distance N'P minimum for the re-creation of the node is taken 50% greater than the distance for deletion of the node. This hysteresis avoids continuous creation and deletion of the same node.



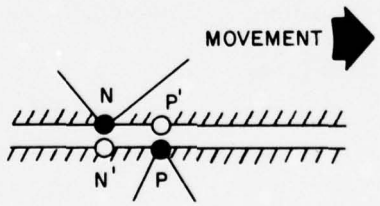
The same formulation given in Section 6.3.2 for the creation of a new node applies for the re-creation. Note that two nodes are re-created at the same time



unless two contacts (N to P and P to N) already exist (see Section 6.3.5.2). Clearly, once N goes past P, nodes opposing both N and P will need re-creation.

6.3.5.2 Redundant Contacts

The deletion of nodes in the case of two adjacent contacts must be handled with special care.



From the above sketch it is clear that nodes P' and N' will soon be deleted. From the deletion of P', a contact P to N will appear; from deletion of N', another contact N to P will appear. If nothing is done, the same contact will be taken into account twice per time step. This is not considered realistic and, besides, it is likely to create problems of stability.

6.4 USE OF PROGRAM DBLOCK

As remarked earlier, the program is not intended as a "user-oriented" package, but is simply a test-bed for proving the numerical scheme. In consequence the input and output is rudimentary.

6.4.1 FILES

No file numbers are referred to in the program since simple PRINT and READ statements are used for output and input. The format for the input stream is given in Appendix X.

6.4.2 PRINTED OUTPUT

The computer provides an echo of all input data as well as the values of all velocities, coordinates and stresses at the iteration cycle numbers requested by the user. However, this type of output is lengthy and difficult to visualise.

6.4.3 PLOTTED OUTPUT

The output is much more manageable in the form of plots. The computer can provide two different types of plots:

6.4.3.1 Meshes

A plot of the complete finite difference mesh will be plotted at the required scale at the cycle numbers pre-selected by the user. The number of plots is at present restricted to 10 by the dimensions in COMMON but can be increased as desired within the core capabilities of the computer used.

6.4.3.2 Histories

The program is prepared to provide time histories of two variables. The two variables of which the history is requested are specified in the routine OUTPUT by statements of the form:

```
BUF1 (NPL) = YD(6)
BUF3 (NPL) = XD(6)
```

The two variables desired should replace XD(6) and YD(6) in the two statements mentioned. The program will then create a file for plotting which can then be plotted by a post-processor.

This is in contrast with the mesh plots which are controlled directly by the program.

6.4.4 BOUNDARY CONDITIONS

The boundary conditions must be inserted explicitly as FORTRAN statements in the routine BOUNDY. This routine is simply:

```
SUBROUTINE BOUNDY
INCLUDE 'COMMON.FTN'
.........
.........
RETURN
END
```

If node 3 is to be fixed in the X-direction, then the user will replace the dots in the above listing by:

```
XD(3) = 0
```

Similarly for any other boundary conditions.

6.4.5 DAMPING

Only stiffness-proportional damping is provided, and is similar to that embodied in program RBM (see Section 2.4.6).

6.5 EXAMPLES AND VALIDATIONS

In this section several examples are presented. Their purpose is two fold: on the one hand, they provide a verification of the adequacy of the program while, on the other hand, they provide a clarification of the type of results generated by the program as well as of its capabilities.

6.5.1 STATIC TESTS UNDER UNIFORM CONDITIONS

The mesh used for these tests was taken to be very irregular with the purpose of verifying the accuracy of the results under difficult conditions. The mesh is shown on Figure 6.3.

Several tests were performed with this mesh. Figure 6.4 shows the theoretical and numerical results when the mesh is subjected to gravity.

It was also verified that all nodes reached equilibrium and stresses were uniform when the domain considered was subjected to uniform boundary stresses.

Another static test performed consisted of imposing a relative vertical displacement between the two horizontal boundaries. In one case, the nodes on the vertical boundaries were forced to the positions they should reach under deformation; in the second case, they were allowed vertical freedom to attain such positions naturally. The results are compared with the true solution in the following tables:

Constrained Case

	Theoretical	Numerical
τ_{xx}	3.75	3.7707
τ_{yy}	1.25	1.2569
τ_{xy}	0	$\leq 6.7 \times 10^{-6}$

Unconstrained Case

	Theoretical	Numerical
τ_{xx}	3.7500	3.7707
τ_{yy}	1.2500	1.2569
τ_{xy}	0	$\leq 9.8 \times 10^{-6}$

6.5.2 WAVE PROPAGATION TESTS

The theoretical velocity of propagation of different waves was compared to that observed from the results of the program. For this purpose, a twenty-story bar was generated and subjected to gravity in different positions and under different boundary conditions. In this way, three types of waves were generated: a) P-waves in a constrained material, b) P-waves in material constrained in one transversal direction but free in the other, and c) S-waves.

A comparison of the accuracy of the resulting periods of oscillation is provided in the next table.

		THEORETICAL VALUE (sec)	OBSERVED VALUE (sec)
P-WAVE	FULLY CONSTRAINED	3.88	3.89
	SEMI-CONSTRAINED	7.12	7.00
S-WAVE		13.66	13.68

The reason why the semi-constrained case does not appear to be as accurate as the others is that the theoretical propagation assumes no lateral inertia in the unconstrained direction. Such inertia, though small, does exist in the numerical representation, thus slightly biasing the result towards the fully constrained value (infinite lateral inertia).

An example of the velocity history is also presented in Figure 6.5. It corresponds to the case of the propagation of the shear wave. Note that the theoretical history is composed of straight lines.

6.5.3 MESH GENERATION TESTS

Several tests were made to illustrate the capabilities of the mesh generating process. Figure 6.6 presents the information initially given to the computer, namely, twelve boundary nodes. Figure 6.7 presents the mesh at the end of the triangularization by MESH. Figures 6.8A and 6.9A show the mesh as obtained by REDUCE for the cases of maximum edge sizes of 3.0 and 2.4 cm, respectively. Finally, Figures 6.8B and 6.9B depict the meshes corresponding to the two cases mentioned after the rezoning carried out by NICE.

6.5.4 BLOCK SLIDING ON INCLINED PLANE

The interest of this test, besides that of providing a clear display of the rezoning activities of the program, is to check the conservation of momentum and energy throughout the rezoning procedure.

The initial configuration is displayed in frame A of Figure 6.10. A three zone block is pushed downslope with vertical and horizontal velocities equal to -1.0. Gravity exists and is equal to 0.98 (units are arbitrary). Under those conditions the vertical velocity of the block should be:

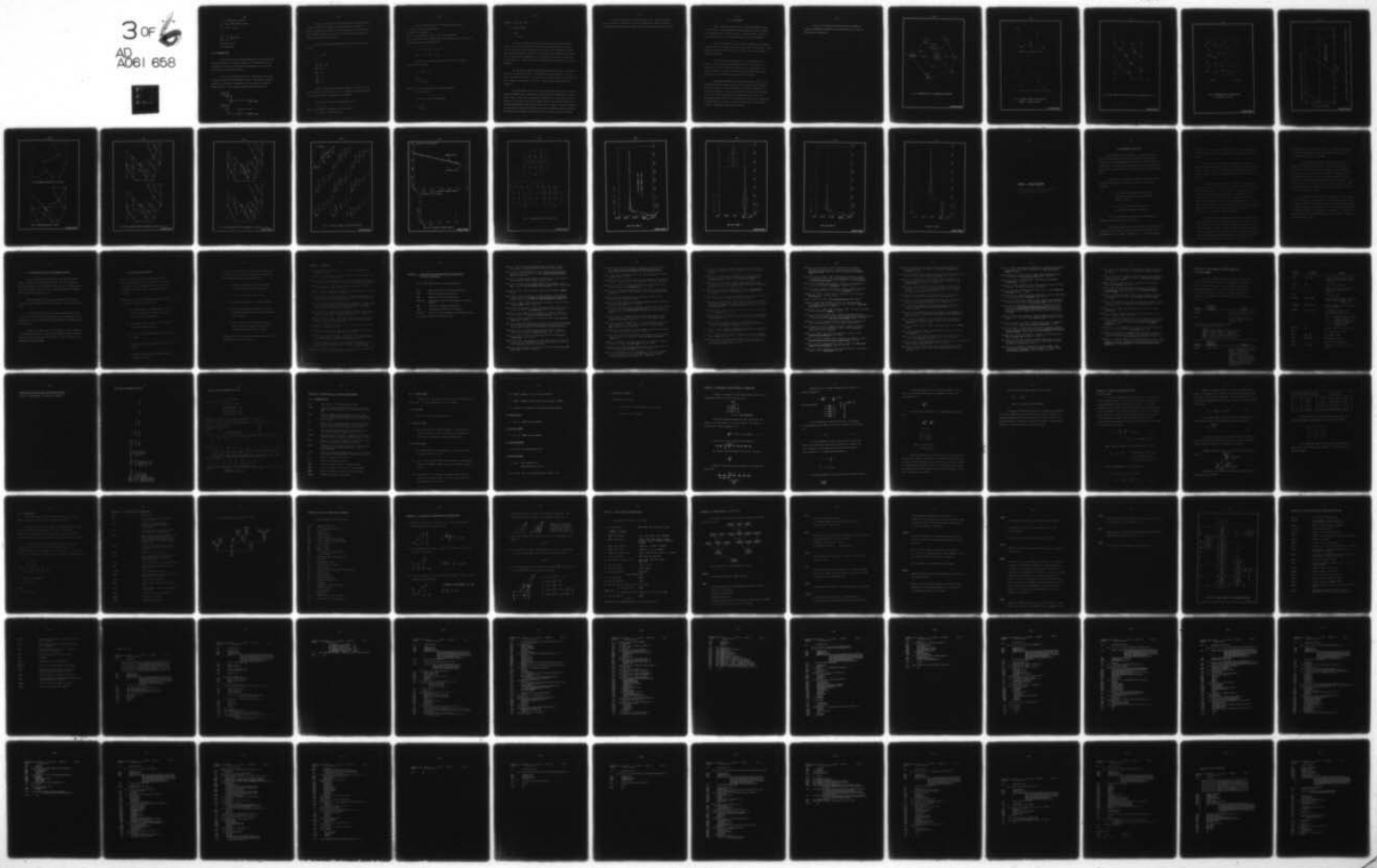
$$v_y = 1 - 0.49t$$

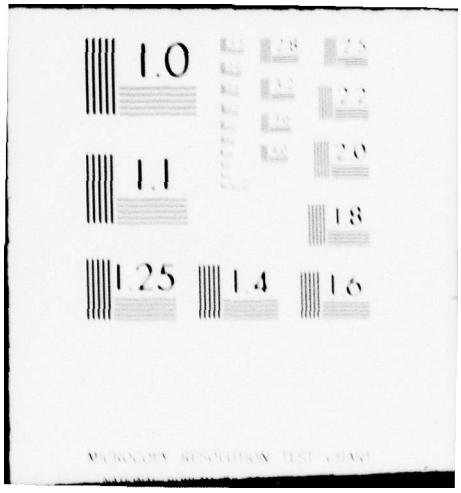
The subsequent frames (6.10B to 6.10I) illustrate the rezoning process. The plot in Figures 6.11 shows a comparison of the theoretical velocity with that computed for the back node of the sliding block. As can be seen, the agreement is very satisfactory and no appreciable energy or momentum losses have been accumulated in the process. Figure 6.12 presents a plot in which the block is pushed without gravity. As shown, no energy is spent in the rezoning procedures.

For the runs described above, the following properties were used:

AD-A061 658 DAMES AND MOORE LOS ANGELES CA F/G 8/7
COMPUTER MODELING OF JOINTED ROCK MASSES.(U)
AUG 78 T MAINI, P CUNDALL, J MARTI DACA39-77-C-0004
UNCLASSIFIED WES-TR-N-78-4 NL

3 OF 6
AD
A061 658




1.0
1.1
1.25
1.4
1.6
2.8
2.5
2.2
2.0
1.8

$\lambda = \mu = 1000$ (Lame's constants)

$K_{NL} = K_{NU} = 5000$ normal stiffness

δ_n = rigid transition

$\rho = 1$

$\lambda_{min} = 5\%$
$f_{min} = 10.0$ } damping

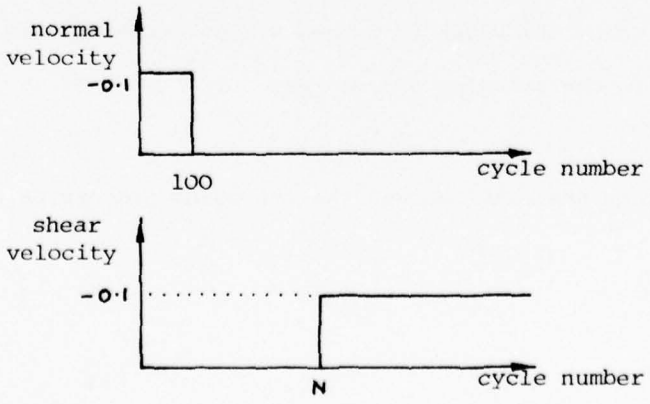
slope angle = 45°

g = 0.98 gravity

6.5.5 SHEARBOX TEST

Similar tests to be those performed with the block sliding down-slope were undertaken to study the behaviour of a simulated shear test configuration. Figure 6.13A presents the arrangement of two blocks and their initial finite difference grids.

As seen in the referenced figure, all five nodes in the lowest horizontal plane remain fixed during the tests. The boundary conditions, specified as velocities, are applied on the five nodes in the highest horizontal plane. Those boundary conditions were as shown below:



Clearly, the two blocks are first compressed together; when this situation stabilizes, the upper plane of the top block is forced to displace horizontally with respect to the lower plane of the bottom block. Total resultant horizontal and vertical forces along the top horizontal plane were monitored during the test.

Material properties and parameters used in these tests were as follows:

$\lambda = \mu = 1000$

$\rho = 1$

$g = 0$

$K_{NL} = K_{NU} = 5000$

$K_{SH} = 1000$

XMU = 0.1

$\lambda_{min} = 0.7$

$f_{min} = 2.0$

FRAC = 0.5

Figures 6.14 to 6.17 present the results of the tests together with the theoretical static results. It is probably worth mentioning how the theoretical result for compression was obtained.

The total relative vertical displacement d is:

$$d = d_1 + d_2$$

where d_1 corresponds to compression of the continuum

d_2 occurs at the discontinuity.

Since the resultant force F is equal across any section:

$F = n K_{NL} d_2$ across the discontinuity

$F = K_1 d_1$ in the continuum

where n is the number of contacts across the discontinuity.

K_1 is similar to a Young's modulus under plane-strain conditions and can be obtained from the following relations:

$$\tau_{xx} = \lambda (\varepsilon_{xx} + \varepsilon_{yy}) + 2\mu \varepsilon_{xx} = 0$$

$$\tau_{yy} = \lambda (\varepsilon_{xx} + \varepsilon_{yy}) + 2\mu \varepsilon_{yy}$$

Eliminating ε_{xx} between both equations, and for the values of the parameters proposed,

$$\tau_{yy} = 2666.7 \varepsilon_{yy}$$

Hence:

$$F = 2666.7 \frac{A}{H} d_1$$

Where A is the cross-sectional area (assumed constant)

H is the height

In terms of the total displacement:

$$F = \frac{d}{\frac{1}{K_1} + \frac{1}{K_2}}$$

Where $K_1 = 2666.7 \frac{A}{H} = 2666.7$

$K_2 = n K_{NL} = 35000$

Hence:

$F = 135.72$

It can be seen that the dynamic solutions converge to the theoretical static ones. Figures 6.14 and 6.15 present the normal and shear forces during Test A, in which the shear displacement started after 1000 iterations (N = 1000). The initial oscillations of the shear force coupled to the compression are thought to be originated by the asymmetry of the mesh and decay due to the stiffness damping incorporated. A small but similar coupling effect is observed at the onset of shearing.

The same test was repeated with shearing velocity equal to -0.05 starting after only 300 iterations (Test B). Normal and shear forces for this test are presented in Figures 6.16 and 6.17. The shear force seems to converge to a slightly low value compared to the theoretical solution. Apart from that the results are as expected.

A special feature of the two tests described was that the nodes of the top and bottom boundaries were allowed free relative horizontal displacement until the onset of shear. Other tests (not presented) showed that a slightly larger value is obtained for the normal load if these boundary nodes are horizontally fixed, due to the obvious constraints on the Poisson's ratio effect. However, as could be expected, such horizontal constraints decreased the effect of the unwanted coupled oscillations introduced by mesh asymmetry and the shear forces converged to the correct value.

Finally, a test was also run without normal force, simply by providing an initial velocity. The constant velocity observed proves again that energy is conserved during rezoning; such rezoning can be visualised in Figures 6.13A to 6.13C.

6.6 CONCLUSIONS

A basic program has been written to analyse deformable continua and discontinua. The program, DBLOCK, works explicitly in the time domain by a Lagrangian finite difference scheme. The implementation of arbitrary constitutive relations for continuous materials is relatively simple.

The contacts between blocks are handled by special logic depending on whether they are considered corner-to-edge or edge-to-edge contacts. Again, arbitrary laws governing the interaction at those contacts are straightforward to introduce.

The main novelty of the program lies on the handling of the contacts. Two grid points are permanently opposed in each contact, the contact forces, hence, being felt by those nodes as accelerations. The meshes are rezoned as required to maintain the two nodes in each contact opposing each other throughout the duration of the contact.

The hypotheses embodied in the program have been tested on static and dynamic problems against analytical solutions, with special emphasis on the conservation of energy and momentum. The results obtained to date are very encouraging. For quasi-static problems, where velocity gradients are low, the errors associated with rezoning tend to zero, since the equations presented in Section 6.3 show that the energy error is proportional to the difference in velocity of the nodes either side of the node being relocated. Momentum is always conserved exactly.

Finally, the program has been provided with a method for generating automatically meshes of the desired densities to cover the domains under consideration.

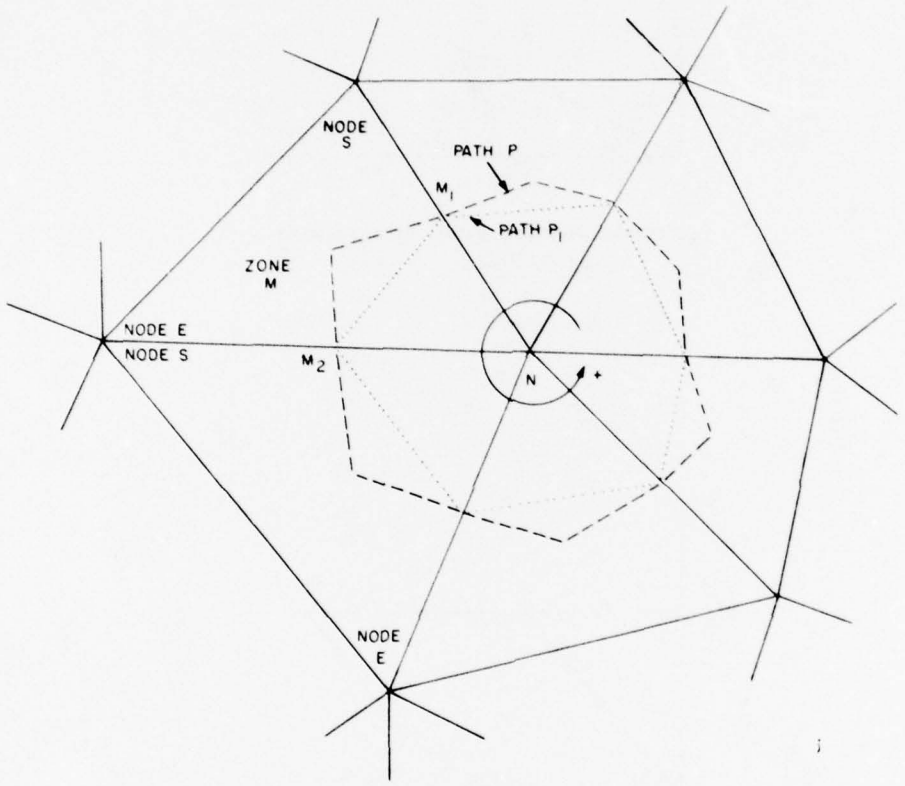
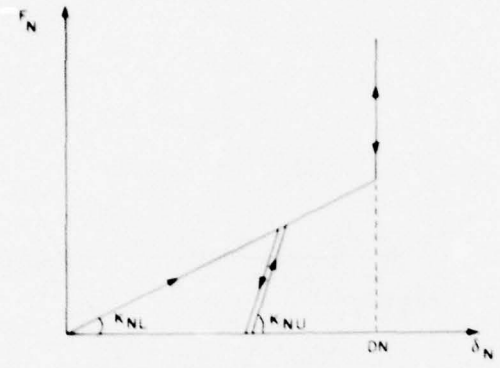


FIG. 6.1 INTEGRATION OF THE MOMENTUM EQUATIONS

DAMES & MOORE



(a) NORMAL FORCE vs NORMAL DISPLACEMENT



(b) TANGENTIAL FORCE vs TANGENTIAL DISPLACEMENT

FIG. 6.2 ASSUMED CONSTITUTIVE LAWS FOR CORNER - TO EDGE CONTACT

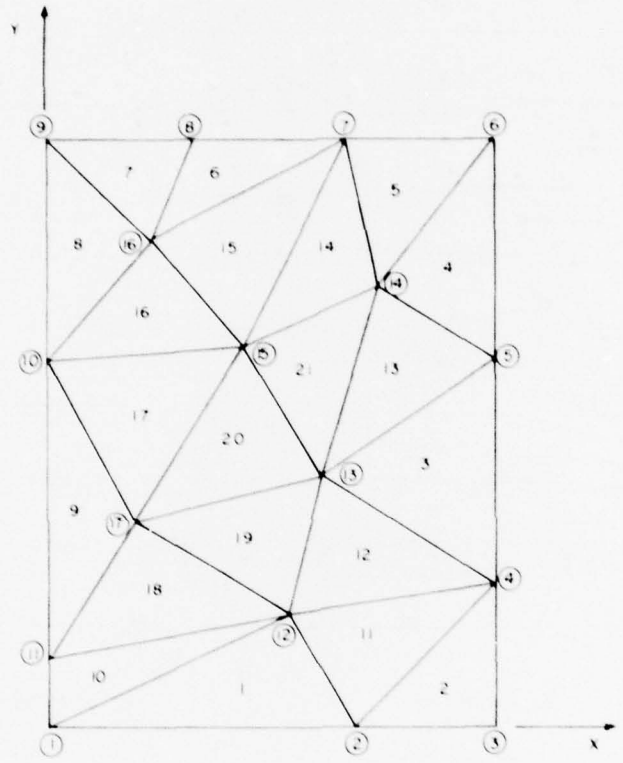
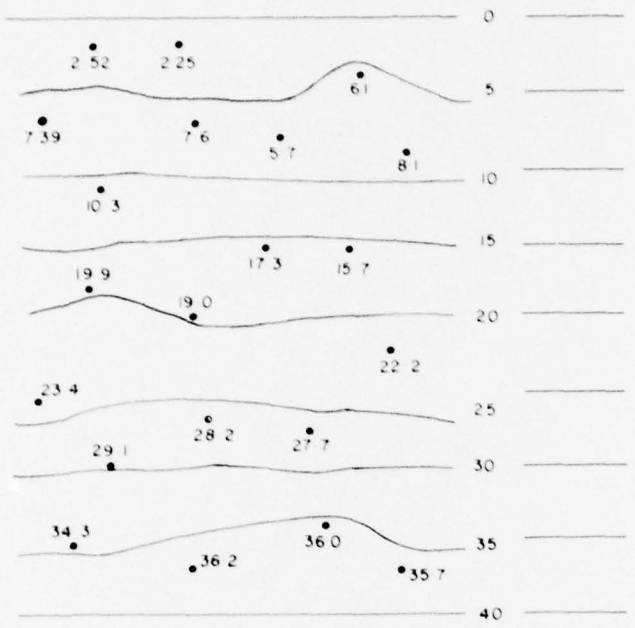


FIG. 6.3 MESH USED FOR THE STATIC TESTS ON PROGRAM D BLOCK

DAMES & MOORE



a) NUMERICAL

b) THEORETICAL

FIG. 6.4 STRESSES DUE TO GRAVITATION
ON THE MESH OF FIG. 6.3

DAMES & MOORE

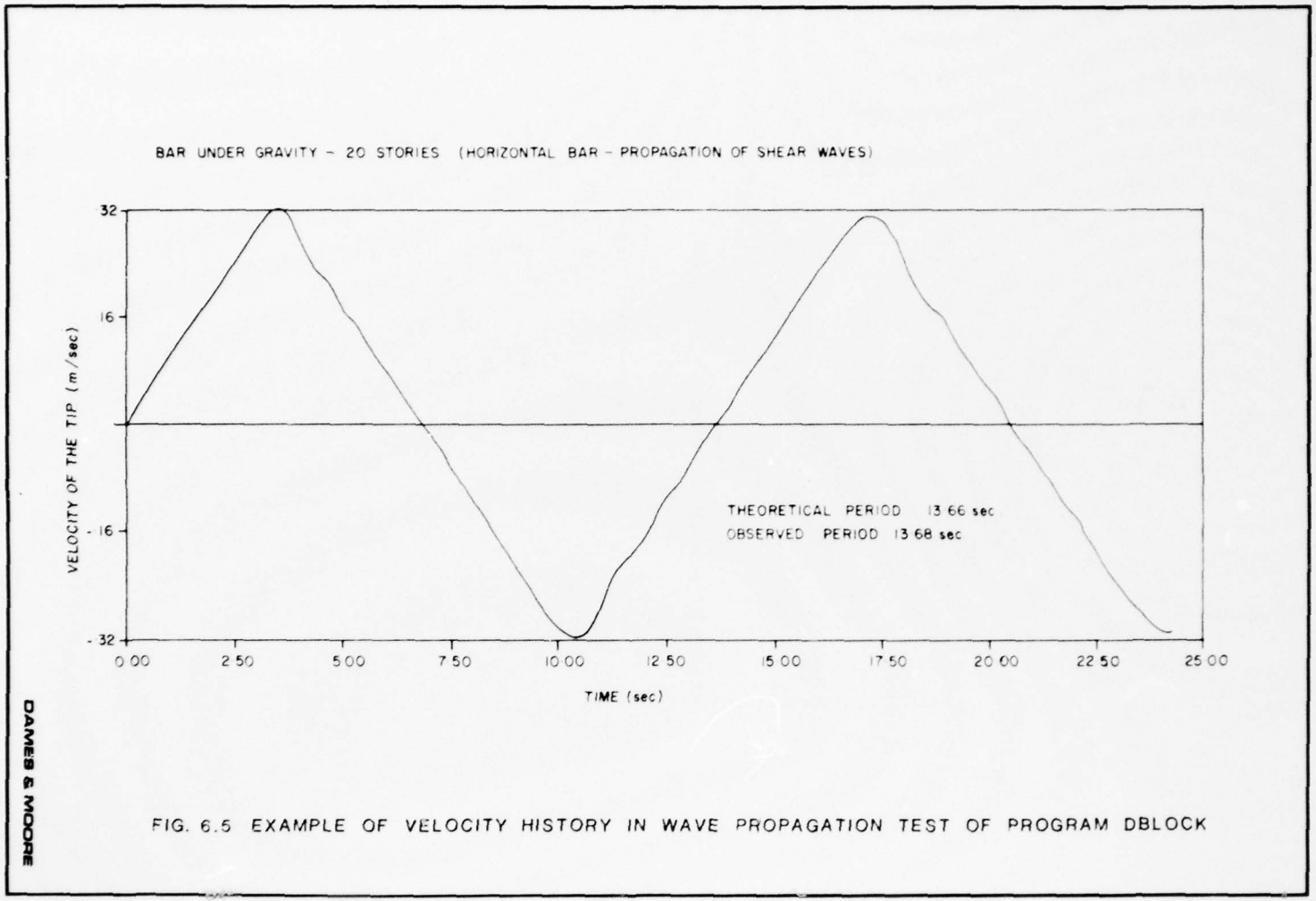


FIG. 6.5 EXAMPLE OF VELOCITY HISTORY IN WAVE PROPAGATION TEST OF PROGRAM DBLOCK

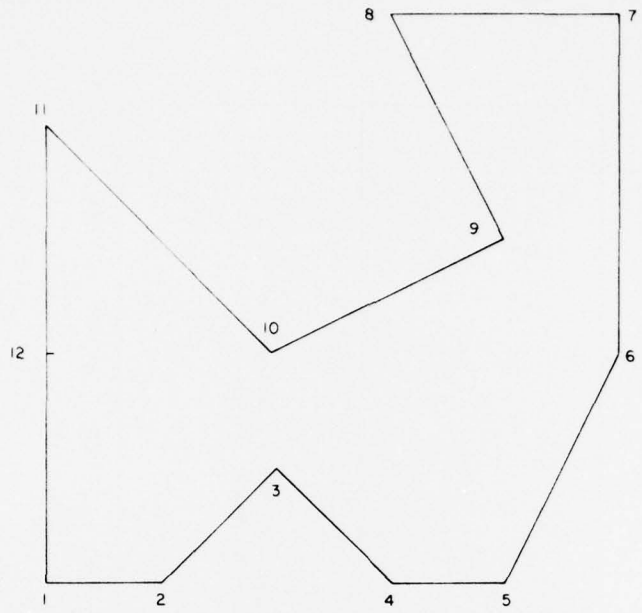


FIG. 6.6 BOUNDARY SUPPLIED BY THE USER

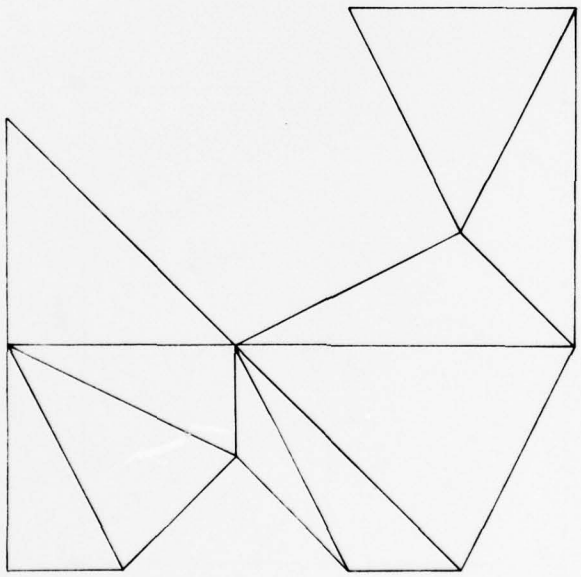
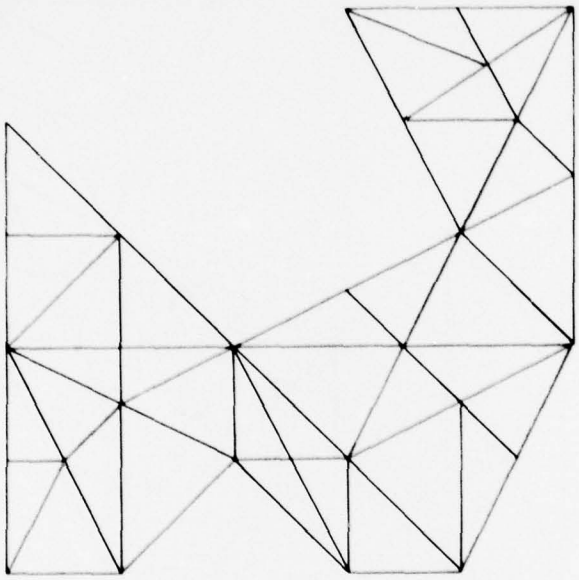
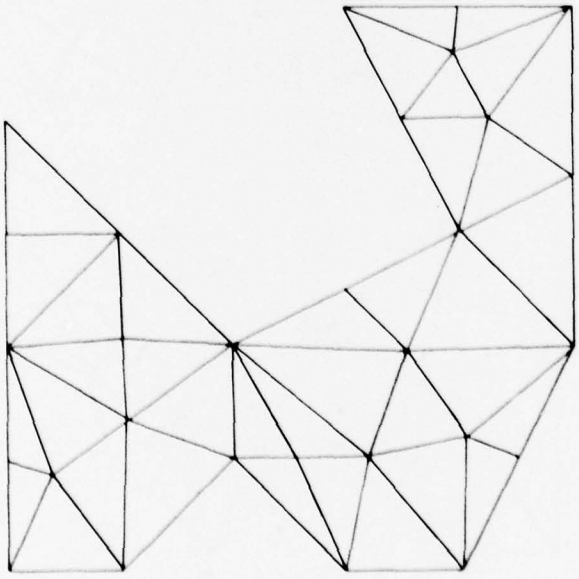


FIG. 6.7 TRIANGULARIZATION BY 'MESH'



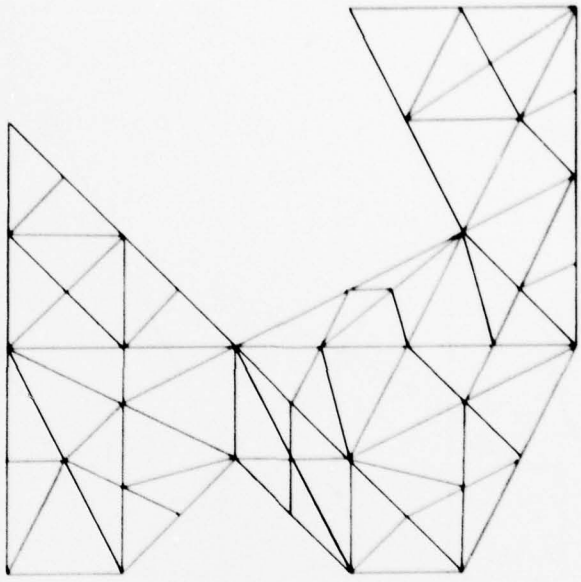
A) MESH AT THE END OF 'REDUCE'



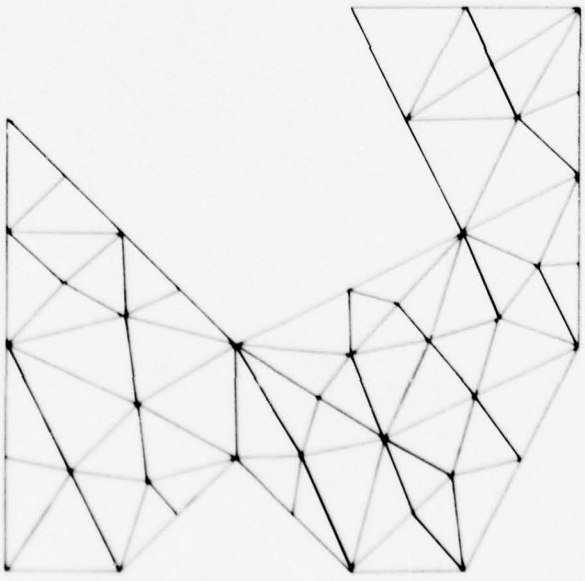
B) MESH AT THE END OF 'NICE'

FIG. 6.8 MESH GENERATED WITH A MAXIMUM EDGE LENGTH OF 3.0 CM

DAMES & MOORE



A) MESH AT THE END OF 'REDUCE'



B) MESH AT THE END OF 'NICE'

FIG. 6.9 MESH GENERATED WITH A MAXIMUM EDGE LENGTH OF 2.4 CM

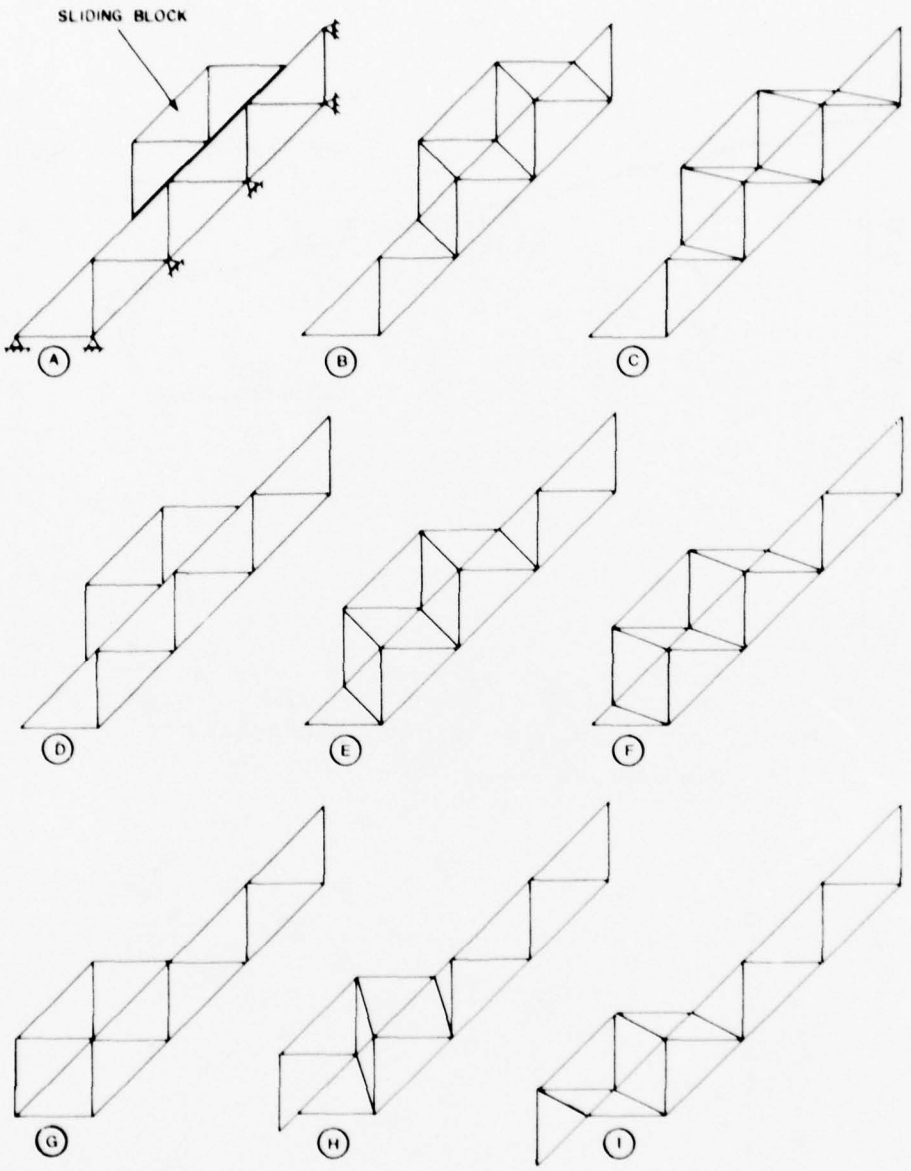


FIG. 6.10 BLOCK SLIDING ON AN INCLINED PLANE

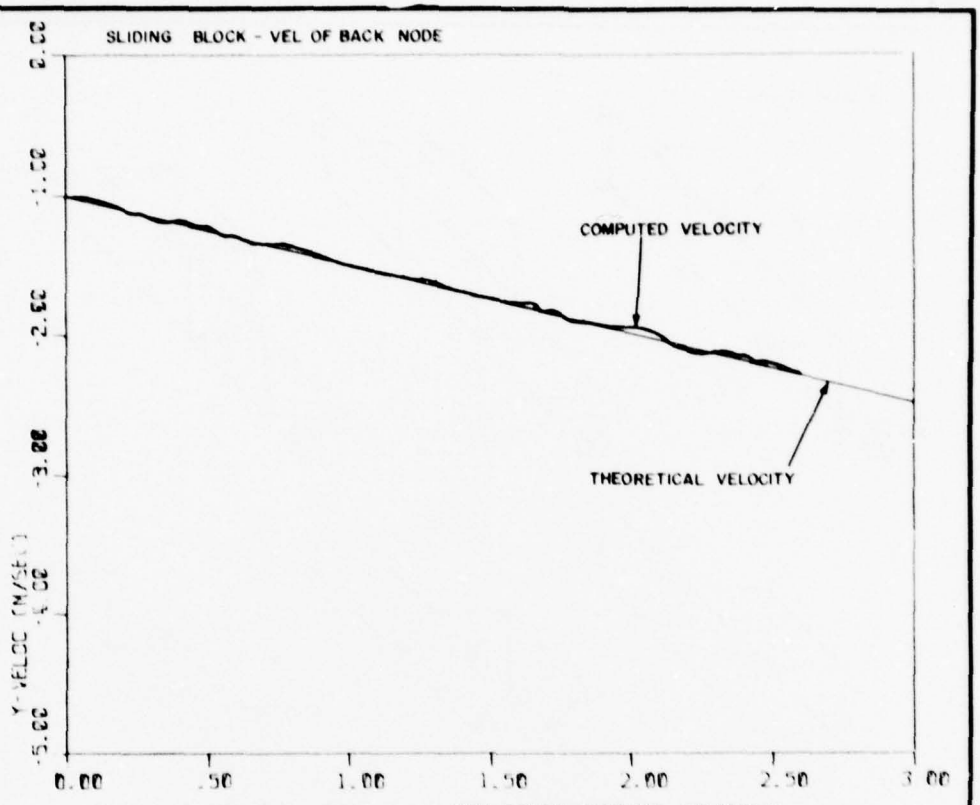


FIG. 6.11 VELOCITY UNDER GRAVITY

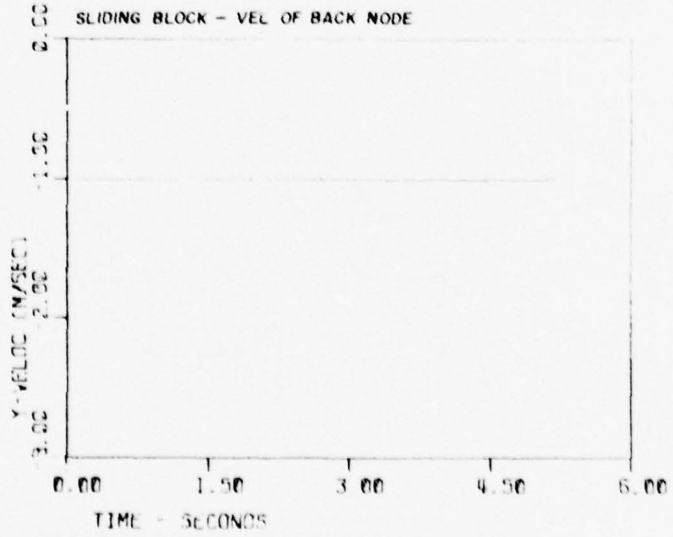


FIG. 6.12 VELOCITY WITHOUT GRAVITY

DAMES & MOORE

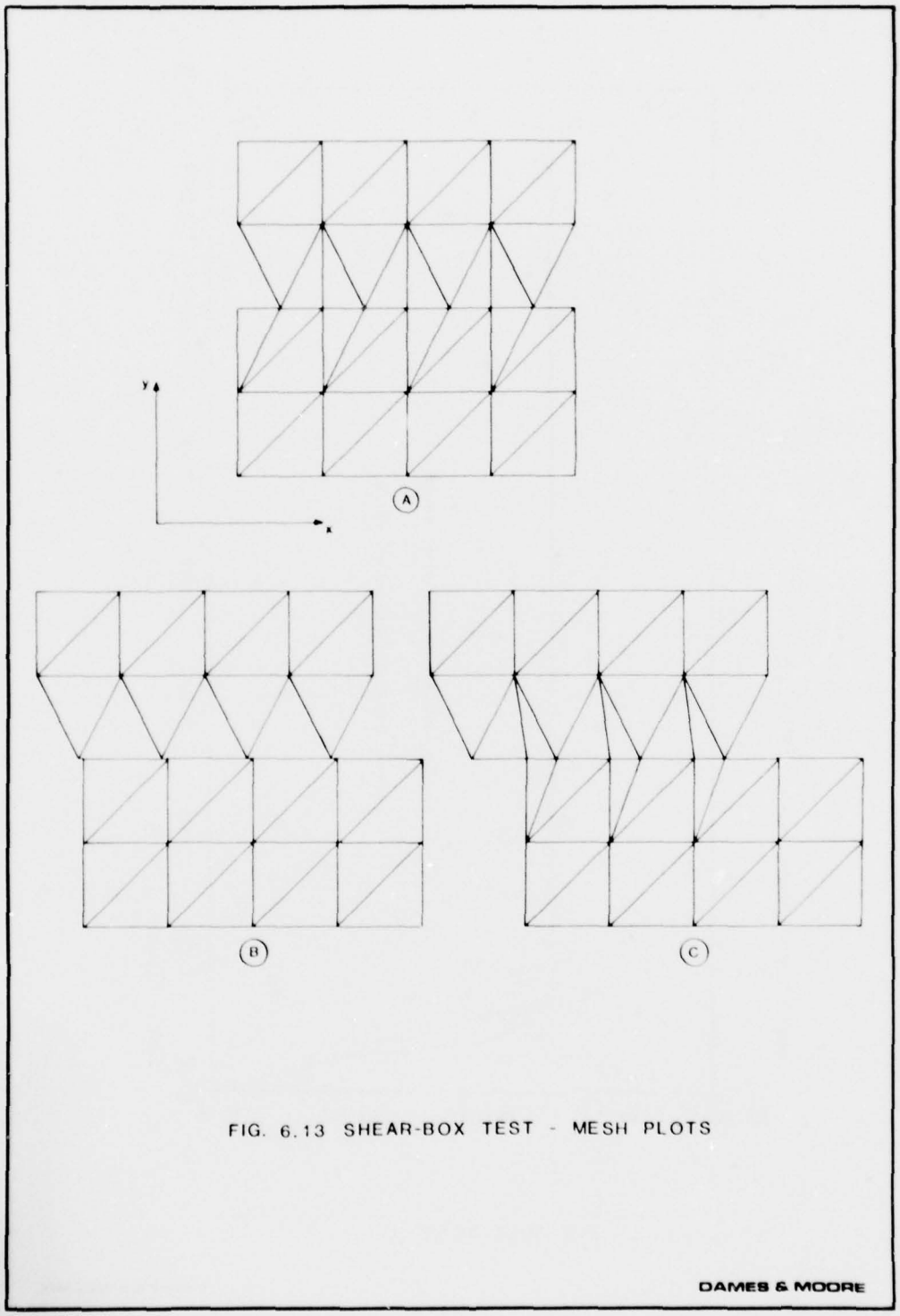


FIG. 6.13 SHEAR-BOX TEST - MESH PLOTS

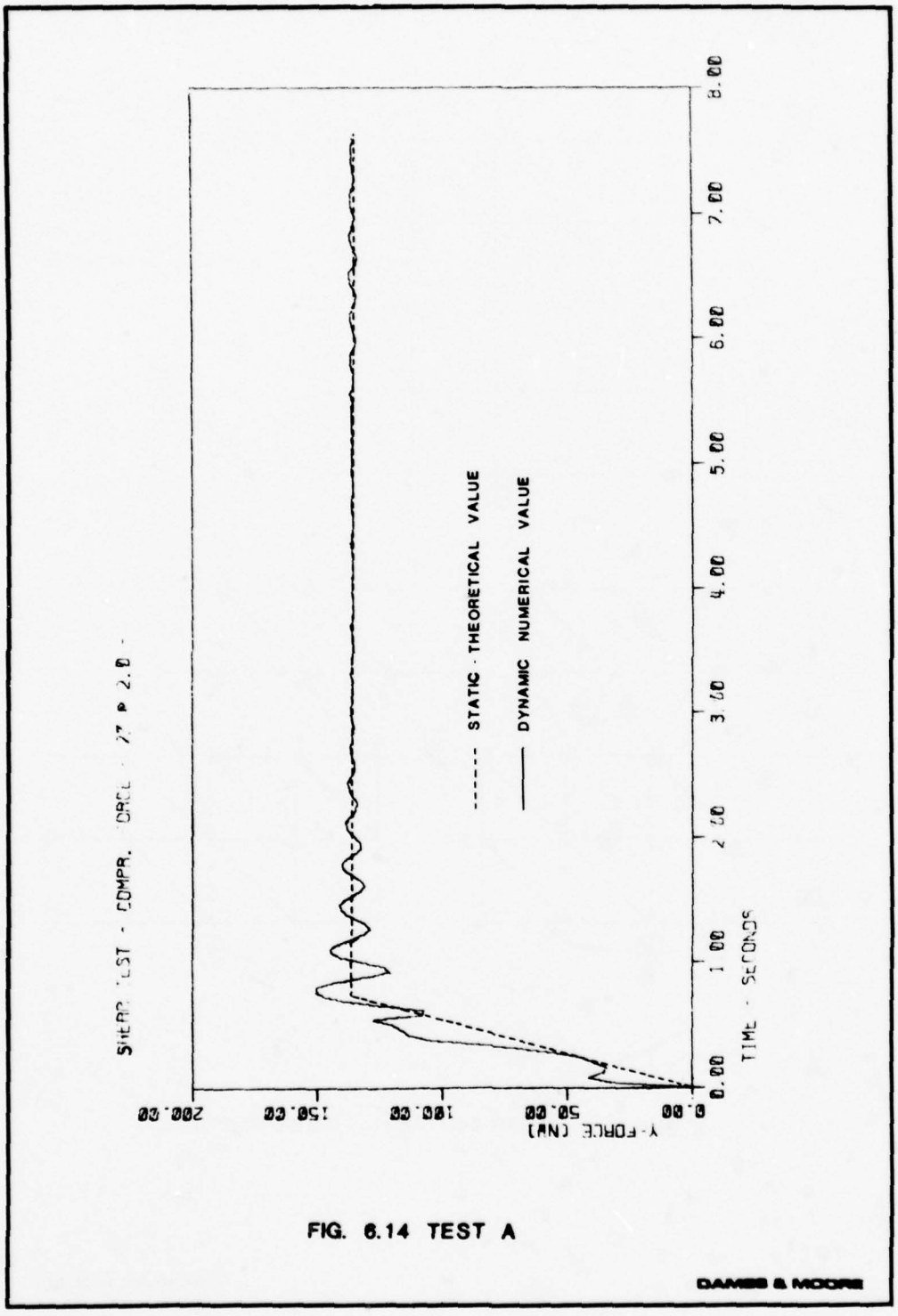


FIG. 6.14 TEST A

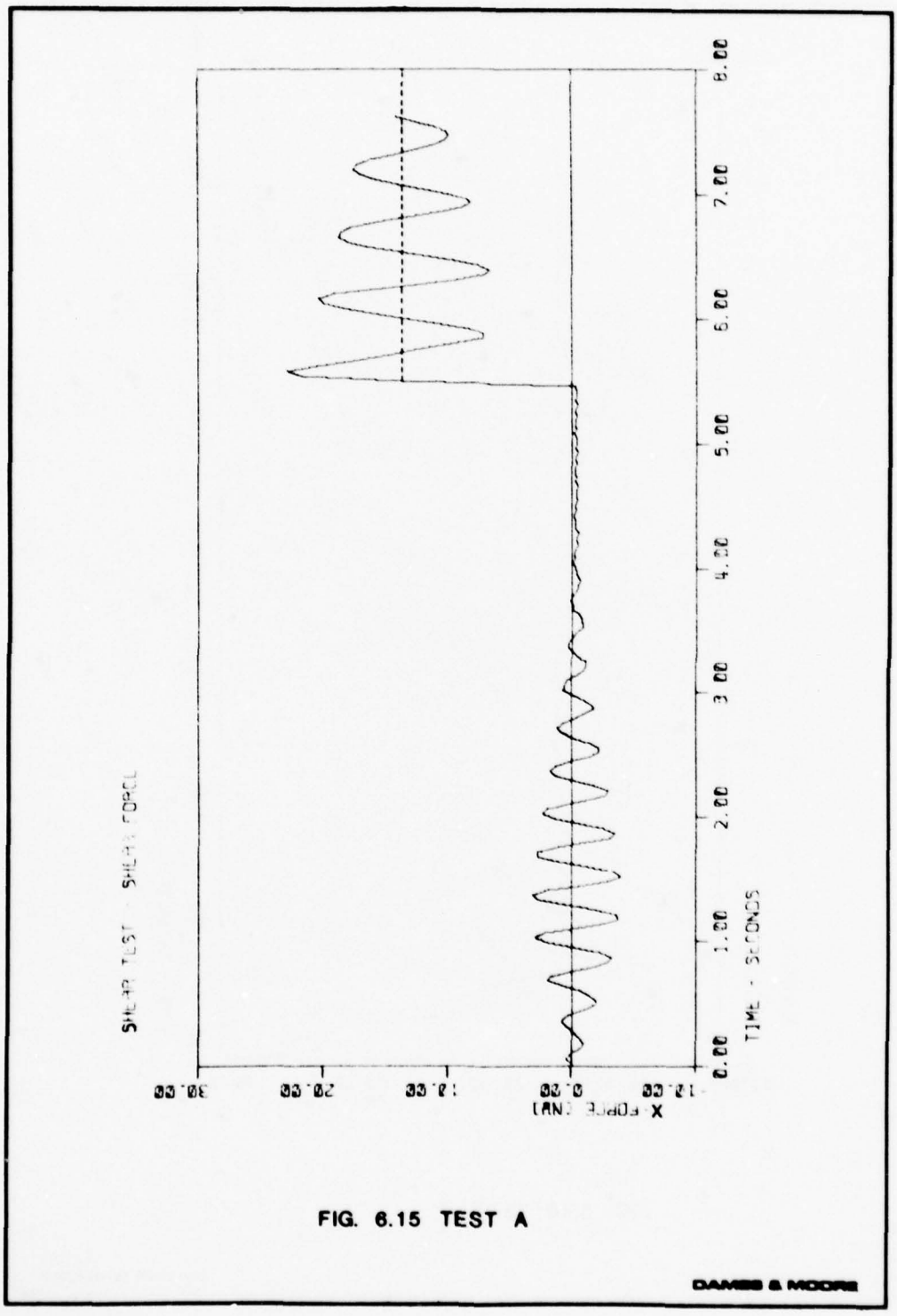


FIG. 6.15 TEST A

DAMES & MOORE

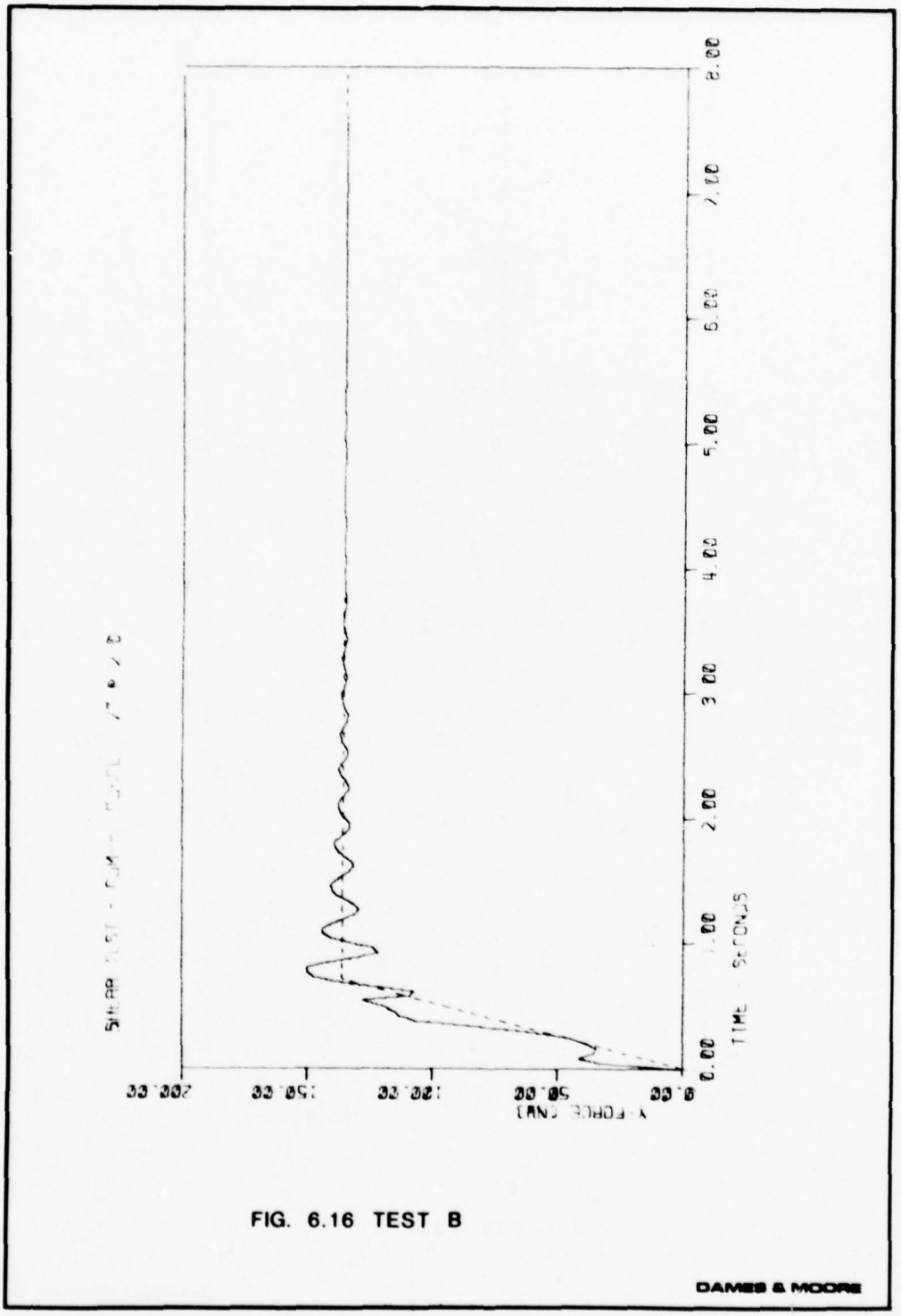


FIG. 6.16 TEST B

DAMES & MOORE

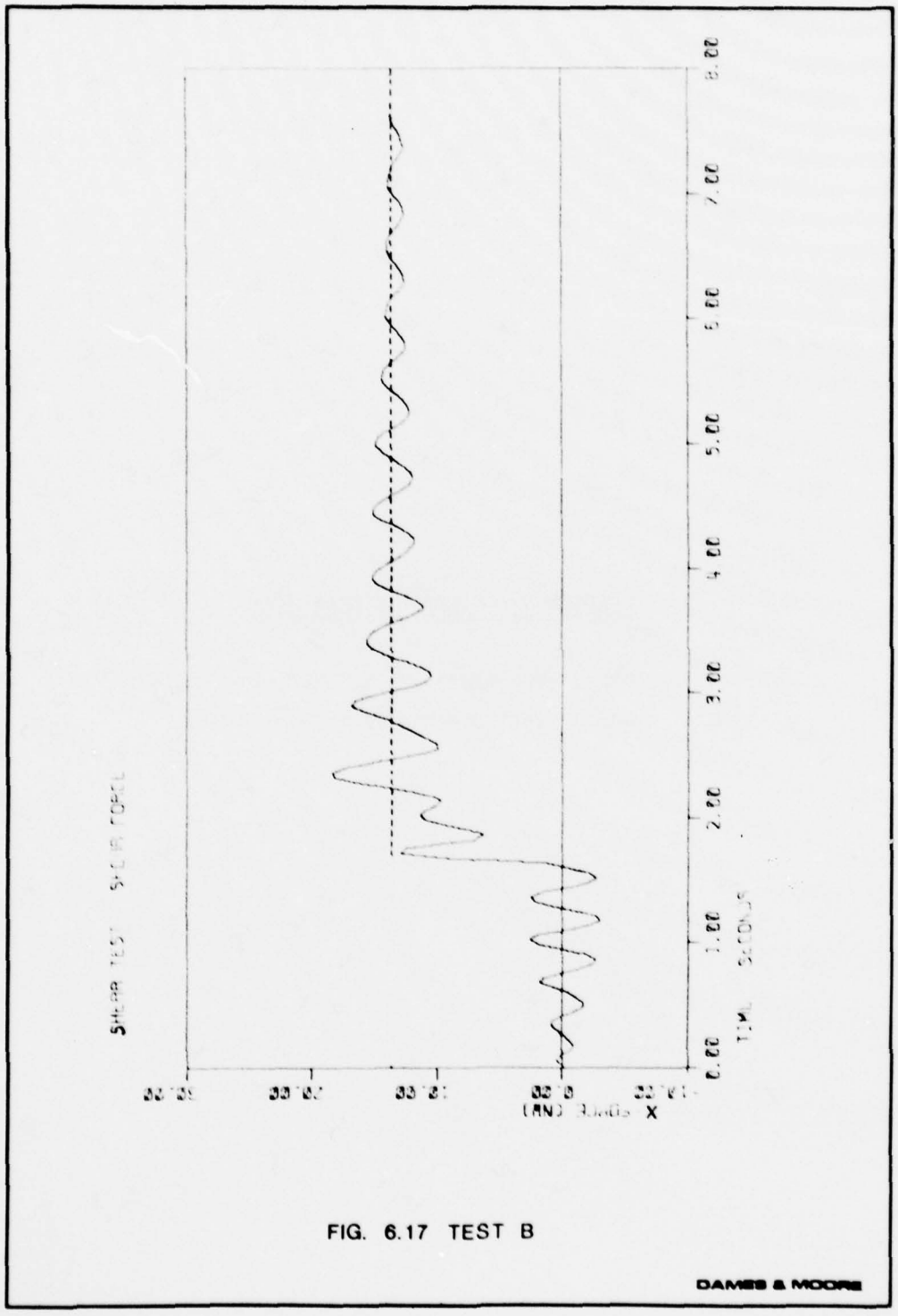


FIG. 6.17 TEST B

DAMES & MOORE

CHAPTER 7: OVERALL CONCLUSIONS

7.0 The achievements of the study are evaluated and suggestions made for future work.

7.1 ACHIEVEMENTS OF THE STUDY

The main object of the research was to demonstrate that the numerical schemes proposed for a fully deformable block program would actually work and give reasonably accurate results. This has now been accomplished. It has been shown that very little error is introduced into the calculation of sliding of rock joints by the various rezoning schemes used.

Although the new formulation is likely to be no more efficient than existing lagrange, finite-difference codes, it has two major advantages:

i) It is completely general, and can model any arbitrary jointing pattern, with no "tuning" needed by the user.

ii) Joints are modelled accurately, with no interpolation necessary at interfaces.

To be set against these advantages is the reputation that triangular zones have for being too "stiff".

The program, DBLOCK, that has been written to perform the verifications is only a "test-bed" program and does not include any of the housekeeping logic necessary for keeping track of interactions between

zones and joints. This logic can easily be built into a future program, since it will be based on that already tested in the original rigid-block program.

Several other goals have been accomplished: the original rigid-block program has been translated into Fortran, and has been extensively validated. It is in the form of a "base-line" program that can be used as a reference version for checking modified and extended versions against.

A new idea has been tried out, in which simple deformability is given to the blocks of the distinct element method. Each block is allowed three degrees of freedom to deform internally, with general constitutive laws being possible for the intact material. The method differs in a fundamental way from both finite elements and finite differences, and relies on the stiffnesses of joints to link neighbouring blocks, or zones. The new program, SDEM, is only slightly slower than the rigid-block program, RBM, and can be used in situations where the intact deformation of rock blocks, while not very large, cannot be neglected.

A modified version of RBM has been written that allows blocks to crack in response to the loads acting on them. The data structures have been changed so that blocks may divide and subdivide indefinitely (until limited by computer storage). A simple cracking criterion has been built into the program, based on semi-empirical results from point-load tests on irregular rock blocks. However, the program is structured so that the user may insert any other criterion with very little effort. It is

anticipated that the new program will be able to increase considerably the realism of simulations in which "hanging-up" of blocks would have occurred using the regular program, RBM.

A fairly extensive literature survey has been made on the properties and behaviour of rock joints. Based on the findings, a constitutive law has been proposed for rough rock joints, and coded up in the form of a Fortran subroutine called JOINT. The program seems to be capable of simulating the range of behaviour observed in the test results; it also gives plausible stress/strain curves for complex loading paths, for which test results do not exist. There is no doubt that when more comprehensive test results become available, the proposed constitutive law will need to be revised.

In summary, the work presented in this report should be regarded as the first phase in a continuing effort to understand and model jointed rock masses. Several tools have been developed, investigated and evaluated; in the next phases it is hoped that some or all of these tools will be used to simulate and predict real experiments on jointed rock. In this way the computer programs can be refined and corrected in the light of comparisons with experimental results.

7.2 APPLICABILITY AND USE OF THE COMPUTER PROGRAMS

Of all the programs that have been written during the present study, only RBM has been extensively validated. The other programs are only made available on the understanding that they are for experimental use only, and the results are not to be relied upon until further validations have been done and any bugs corrected.

RBM is applicable to jointed rock problems where the stresses are low, so that deformations at joints far exceed any intact rock deformations.

For situations involving higher stresses, SDEM should be used, up to the point where intact block deformations become comparable to joint deformations. At this stage SDEM will become inaccurate, and DBLOCK should be used instead.

If the rock is brittle and of low strength, cracks are probable, so that RBMC should be used. Parameter studies should be made in which the crack criterion is varied, since the mechanics of crack initiation is only approximately treated by RBMC.

7.3 SUGGESTIONS FOR FUTURE WORK

1. The programs developed in this study should be used to model existing laboratory tests on multiple-block systems, to determine the range of applicability of each program, and whether any important aspect of behaviour is being neglected.

2. RBM should be generally streamlined and made more efficient. In particular the program should include:

   i) the schemes embodied in SDEM and RBMC as options that can be "switched off" without degrading the efficiency of the standard RBM;

   ii) edge-to-edge contact, together with the constitutive law developed in Chapter 5;

   iii) logic to allow different properties for any joint or block;

   iv) water pressure in joints (without flow as a first option);

   v) optimised data structures (eliminating several redundant variables).

3. DBLOCK should be generalised, using the housekeeping logic of RBM, so that any geometry can be treated automatically. Other capabilities should be included, such as:

   i) a general constitutive law for the continuum zones (including non-associated plasticity);

   ii) a general constitutive law for joints;

   iii) non-reflecting boundaries for dynamic problems;

   iv) free-format input routine, general specification of boundary conditions and sophisticated graphical output displays;

   v) logic to allow cracks to propagate through the intact blocks, by creating new triangular zones as the crack moves through the continuum.

DBLOCK should also be extensively validated against existing experimental results on jointed rock.

APPENDIX I : REFERENCES

Broch, E., and J.A. Franklin (1972): "The point load strength test", IJRM&MS, V9, p669.

Brook, N. (1977): "The use of irregular specimens for rock strength tests", Int. J. RM&MS., Vol. 14, p193-202.

Burman, B.C. (1971): "A numerical approach to the mechanics of discontinua", Ph.D. Thesis, James Cook Univ. of N. Queensland, Townsville, Australia.

Chappel, B.A. (1972): "The mechanics of blocky material", Ph.D. thesis, Australian National Univ., Canberra.

Chappel, B.A. (1974): "Numerical and physical experiments with discontinua", Proc. 3rd Cong. ISRM, Denver, V2A, p118.

Cundall, P.A. (1971): "A computer model for simulating progressive, large scale movements in blocky rock systems", Symposium of International Society of Rock Mechanics, Nancy, France.

Cundall, P.A. (1974): "Rational design of tunnel supports: a computer model for rock mass behaviour using interactive graphics for the input and output of geometrical data", Technical Report MRD-2-74, Missouri River Division, U.S. Army Corps of Engineers.

Cundall, P.A. (1976): "Explicit finite-difference methods in geomechanics", Proceedings of ASCE Engineering Foundation Conference on Numerical Methods in Geomechanics - Virginia, June 1976.

Cundall, P.A. (1978): "BALL - A program to model granular media using the distinct element method", Dames & Moore Advanced Technology Group, Technical Note TN-LN-13, March, 1978.

Fairhurst, C. (1964): "On the validity of the Brazilian test for brittle materials", IJRM&MS, V1, p535.

Franklin, J.A., E. Brock and G. Walton (1971): "Logging the mechanical character of rock", Trans. IMM, Sect. A, V81, p43.

Goodman, R.E. and J.W. Bray (1976): "Toppling of rock slopes", paper in "Rock engineering for foundations and slopes", Vol. II, proc. ASCE Speciality conference, Boulder, Colorado, August, 1976.

Seed, H.B. and I.M. Idriss (1970): "Soil moduli and damping factors for dynamic response analysis", Report No. EERC 70-10, Earthquake Engineering Research Center, Univ. of Calif., Berkeley, December, 1970.

Wilkins, M.L. (1969): "Calculation of elastic-plastic flow", Report UCRL-7322, Lawrence Radiation Laboratory, Livermore.

APPENDIX II : BIBLIOGRAPHY ON JOINT PROPERTIES AND JOINTED MODELS (PHYSICAL AND NUMERICAL)

The following abbreviations are used in Appendix II:

AIME	American Institute of Mining and Metallurgy
ARPA	Advanced Research Project Agency
ASCE	American Society of Civil Engineers
IJRM & MS	International Journal of Rock Mechanics and Mining Science
IMM	Institution of Mining and Metellurgy
ISRM	International Society for Rock Mechanics
JSM & FD	Journal of the Soil Mechanics and Foundation Division
U.S.B. Mines	United States Bureau of Mines

Ashby, J. (1971): "Sliding and toppling modes of failure in models and jointed rock slopes" MsC Thesis, Imperial College, London.

Barton, N.R. (1971a): "Estimation of in-situ shear strength from back analysis of failed rock slopes" Proc. Symposium on Rock Fracture, Nancy 1971, paper 2-27.

Barton, N.R. (1971b): "A model study of the behaviour of steep excavated rock slopes" PhD thesis, Imperial College, London.

Barton, N.R. (1971c): "A relationship between joint roughness and joint shear strength" Proc. Int. Symp. on Rock Fracture, Nancy, (ISRM), paper 1-8.

Barton, N.R. (1972): "A model study of rock-joint deformation" IJRM&MS V9 n5.

Barton, N.R. (1974a): "A review of the shear strength of filled discontinuities in rock" Saertrykk, Fjellsprengningsteknikk Borgmekanikk, (Tapir, Norway) - Brock, Heltzen, and Johannesen, ed. Chapter 19.

Barton, N.R. (1974b): "Estimating the shear strength of rock joints" Proc. 3rd Cong. ISRM, Denver, V2A, p219.

Barton, N.R. and Choubey, V. (1978): "The Shear Strength of Rock Joints in Theory and Practice" Rock Mechanics (Preprint to be published).

Bernaix, J. (1969): "New laboratory methods of studying the mechanical properties of rocks" Int. J. Rock Mech. Min. Sci. Vol 6, pp. 43-90.

Bernaix, J. (1974): "Properties of rock and rock masses" Proc. 3rd Cong. ISRM, Denver, V1A, p9.

Best, B.S. (1970): "An investigation into the use of finite element methods for analysing stress distributions in block jointed rock masses" PhD Thesis, James Cook Univ. of N. Queensland, Townsville, Australia.

Bieniawski, Z.T., H.G. Denkhaus and U.W. Vogler (1969): "Failure of Fractured Rock" Int. J. Rock Mech. Min. Sci. Vol. 6, pp.323-341.

Bray, J.W. (1967): "A study of jointed and fractured rock" Rock Mech. and Eng. Geol., V5, n2 and n3.

Brown, E.T. (1968): "The influence of planar discontinuities on the shear strength of a rock-like material" PhD thesis, James Cook University of N. Queensland, Townsville, Australia.

Brown, E.T. (1970a): "Modes of failure in jointed rock masses" Proc. 2nd Cong. ISRM, Belgrade, V2, paper 3-42.

Brown, E.T. and D.H. Trollope (1970b): "Strength of a model of jointed rock" Journal of the Soil Mechanics and Foundations Division, American Society of Civil Engineers Vol. 96. No SM2, pp 685-704.

Brown, E.T. (1970c): "Strength of models of rock with intermittent joints" J. SM&FD, ASCE, V96, p1935.

Burman, B.C. (1971): "A numerical approach to the mechanics of discontinua" PhD Thesis, James Cook Univ. of N. Queensland, Townsville, Australia.

Byerlee, J.D. (1967): "Frictional characteristics of granite under high confining pressure" J. Geophysical Res. V 72, p3639.

Byrne, R.J. (1974): "Physical and numerical models in rock and soil slope stability" PhD Thesis, James Cook Univ. of N. Queensland, Townsville, Australia.

Chappel, B.A. (1972): "The mechanics of blocky material" PhD thesis, Australian National Univ., Canberra.

Chappel, B.A. (1974): "Numerical and physical experiments with discontinua" Proc. 3rd Cong. ISRM, Denver, V2A, p118.

Coulson, J.H. (1970): "The effect of surface roughness on the shear strength of joints in rocks" Technical Report MRD-270 to U.S. Army Corps of Engineers, Omaha, Nebraska by Dept. of Civil Eng., Univ. of Illinois, Urbana.

Cundall, P. (1971): " A computer model for simulating progressive large scale movements in blocky rock systems" Proc. Int. Symp. on Rock Fracture, Nancy (ISRM), paper 2-8.

Cundall, P.A. (1974): "A computer model for rock mass behavior using interactive graphics" U.S. Army. Corps of Engineers, Technical Report MRD 2-74 (Missouri River Division)

de Rouvray, A. (1970): "Analysis and model studies of underground openings in jointed rock" Pub. Univ. of Calif. Berkeley.

Einstein, H.H., R.A. Nelson, R.W. Bruhn and R.C. Hirschfeld (1970): "Model studies of jointed rock behavior" Proc. 11th Symp. on Rock Mech. (AIME), p83.

Engelder, J.T. (1974): "Coefficients of friction for sandstone sliding on quartz gouge" Proc. 3rd Cong. ISRM, Denver, V2A, p499.

Evdokimov, P.D., and D.D. Sapegin (1970): "A Large scale field shear test on rock" Proc. 2nd Cong. ISRM, Belgrade, V2, paper 3-17.

Fecker, E., and N. Rengers (1971): "Measurement of large scale roughness of rock planes by means of profilograph and geological compass" Proc. Int. Symp. on Rock Fracture, Nancy (ISRM) paper 1-18.

Gale, J. (1975): "A numerical, field and laboratory study of flow in rocks with deformable fractures" PhD Thesis, Univ. of California, Berkeley.

Ghaboussi, J., E.L. Wilson and J. Isenberg (1973): "Finite element for rock joints and interfaces" Journal of Soil Mechanics and Foundation Division, ASCE, v. 99, n. SM10, pp. 833-848.

Goodman, R.E., R.L. Taylor and T. Brekke (1968): "A model for the mechanics of jointed rock" Journal of Soil Mechanics and Foundation Division ASCE, v. 94, n. SM3 pp. 637-658.

Goodman, R.E. (1970): "The Deformability of Joints" in Determination of the in-situ modulus of deformation of rock, (ASTM), STP 1-77, pp. 174-196.

Goodman, R.E. and J. Dubois (1972a): "Duplication of dilatancy in analysis of jointed rocks" J. SM&FD, ASCE V98. No. SM4 p399.

Goodman, R.E., F.E. Heuze and Y. Ohnishi (1972b): "Research on strength-deformability-water pressure relationships for faults in direct shear" Final report to ARPA - contract No. H0210020, University of Calif. Berkeley, April 1972.

Goodman, R.E. and Y. Ohnishi (1973): "Undrained shear testing of jointed rock" Rock Mechanics, V5, p129.

Goodman, R.E., (1974): "The Mechanical Properties of Joints" Proceedings Third Congress, International Society for Rock Mechanics, Denver, V. 1-A, pp.127-140.

Goodman, R.E. and C. St. John: "Static finite element analysis of jointed rock" in Numerical Methods in Geotech. Eng., Christian and Desai, eds. (McGraw-Hill), in press.

Goodman, R.E. (1976): Method of geological engineering in discontinuous rocks West Publishing Co., New York.

Handin, J., and D.W. Stearns (1964): "Sliding friction of rock" Trans. Amer. Geophysical Union, V45, p103 (Abstract - 45th Annual Meeting).

Hayashi, M. (1966): "Strength and dilatancy of brittle jointed mass - the extreme value stochastics and anisotropic failure mechanism" Proc. 1st Cong. ISRM, Lisbon, V1, p295.

Hayahsi, M., and Y. Fugiwari (1968): "A mechanism of anisotropic dilatancy and shear failure of laminately jointed rock masses" Tech. Rep. C67006, Central Res. Inst. of Electric Power Industry, Tokyo.

Hayashi M., and Y. Kitihara (1970): "Anisotropic dilatancy and length of jointed rock masses, and stress distribution in fissured masses" in Rock Mechanics in Japan, V1, p85 Japan Soc. of Civil Engineers.

Heuze, F., R.E. Goodman and A. Bornstein (1971a): "Joint perturbation and no tension solution" Rock Mech. (J. ISRM) V.2, n.5.

Heuze, F.E., R.E. Goodman and A. Bornstein (1971b): "Numerical analyses of deformability tests in jointed rock" Rock Mech. V.3, p13.

Hodgson, R.A. (1961): "Classification of Structures on Joint Surfaces" Am. Jour. Sci., V.259, pp. 493-502.

Hoek, E. and J. Bray (1974): Rock Slope Engineering (IMM, London).

Hofman, H. (1970): "The deformation process of a regularly jointed discontinuum during the excavation of a cut" (in German), Proc. 2nd Cong. ISRM Belgrade, V.3, paper 7-1.

Hoskins, E.R., J.C. Jaeger and K.J. Rosengren (1968): "A medium scale direct friction experiment" IJRM&MS, V5, p143.

Iida R., K. Hojo and J. Harada (1974): "In-situ tests and theoretical studies on the relations between looseness and deformation characteristics of jointed rock masses" Proc. 3rd Cong. of ISRM, Denver, V2B, p719.

Isenberg, J. (1972): "Analytic modelling of rock structure interaction" Report from Agbabian Assoc. to U.S.B. Mines, (AD 749-373).

Iwai, Katsuhiko., (1976): "Fundamental studies of fluid flow through a single fracture" PhD dissertation, Dept. of Civil Engineering, University of Calif., Berkeley.

Jaeger, J.C. (1959): "The frictional properties of joints in rock" Geofisica Pura e Applicata, V.43, p148.

Jaeger, J.C. and K.J. Rosengren (1969): "Friction and sliding of Joints" Proc. Australasian Inst. of Min. and Metallurgy, n229, p93.

Jaeger, J.C. (1970): "The behaviour of closely jointed rock" Proc. 11th Symposium on Rock Mechanics, Berkeley, pp 57-68

Jaeger, J.C. (1971): "Friction of rocks and the stability of rock slopes - Rankine lecture" Geotechnique, V21, p97.

Krsmanovic, D. and Z. Langof (1964): "Large scale laboratory tests of the shear strength of rocky material" Rock Mech. and Eng. Geol. Supplement II, p20.

Krsmanovic, D. and M. Popovic (1966a): "Large scale field tests of the shear strength of limestone" Proc. 1st Cong. ISRM, Lisbon, V1, p773.

Krsmanovic D., M. Tufo and Z. Langof (1966b): "Shear strength of rock masses and possibilities of its reproduction on models" Proc. 1st Congress ISRM, Lisbon, paper 3.52, pp537-542.

Krsmanovic, D. (1971): "On the results of measurement of stresses and strains in the rock mass of a geostatical model of the Grancarevo Dam" Publ. of the Inst. of Geotechnics and Found. Eng., Faculty of Civil Engineering, Sarajevo, Yugoslavia.

Krsmanovic, D. and Z. Langof (1971): "On the results of testing the Geostatical model of the Grancarevo Dam" (see Krsmanovic, D.)

Kutter, H.K. (1971): "Stress distribution in direct shear test samples" Proc. Int. Symp. on Rock Fracture, Nancy, (ISRM), paper 2-6.

Kutter, H.K. (1974): "Results of laboratory direct shear tests on four rock types" Rock Mechanics Research Report number 28, Imperial College, London.

Ladanyi, B. and G. Archambault (1970): "Simulation of shear behaviour of jointed rock mass" Proceedings Eleventh Symposium on Rock Mechanics, AIME, pp. 105-125

Ladanyi, B. and G. Archambault (1972): "Evaluation de la resistance au cisaillement d'un massif rocneus fragmente" Proc. 24th Int. Geol. Cong. Montreal, Section 13, p249.

Lajtai, E.Z. (1969a): "Shear strength of weakness planes in rock" IJRM&MS, V6 p499.

Lajtai, E.Z. (1969b): "Strength of discontinuous rocks in direct shear" Geotechnique 19, No.2, 218-233.

Lane, K.S. and W.J. Heck (1964): "Triaxial testing for strength of rock joints" Proc. 6th Symp. on Rock Mech. Univ. of Missouri, p98.

Locher, H.G. (1968): "Some results of direct shear tests on rock discontinuities" Proc. 2nd Int. Symp on Rock Mech., Madrid (ISRM) V171.

Mahtab, M.A. and R.E. Goodman (1970): "Three dimensional finite element analysis of jointed rock slopes" Proc. 2nd Cong. ISRM, Belgrade, V3, paper 7-12.

Malina, H. (1969): "The numerical determination of stresses and deformations in rock taking into account discontinuities" Proc. 19th Coll. on Geomech. Salzburg.

Martin, G.R. and P.J. Millar (1974): "Joint strength characteristics of a weathered rock" Proc. 3rd Cong. ISRM, Denver, V2A, p263.

Morland, L.W. (1974): "Continuum model of regularly jointed mediums" J. Geoph. Res., V79, p357.

Patton, F.D. (1966a); "Multiple modes of shear failure in rock and related materials". University of Illinois. PhD Thesis.

Patton, F.D. (1966b): "Multiple modes of shear failure in rock" Proc. 1st Cong. ISRM, Lisbon, V1, p509.

Piteau, D.R. (1973): "Characterizing and extrapolating rock joint properties in engineering practice" Rock Mech. Supp. II, p5.

Pratt, H.R., A.D. Black and W.F. Brace (1974a): "Friction and deformation of jointed quartz diorite" Proc. 3rd Cong. ISRM, Denver, V2A, p306.

Pratt, H.R., H.S. Swolfs and A.D. Black (1974b): "Properties of in-situ jointed rock" Final Technical Report No. TR54-57, Terra-Tek, Salt Lake City, Utah, Submitted to Environmental Sciences Division, U.S. Army Research Office, Durham, North Carolina, Contract No. DAH Co4 72 C 0049.

Rengers, N. (1970): "The influence of surface roughness on the friction properties of rock planes" Proc. 2nd Cong. ISRM, Belgrade, V1, paper 1-31.

Ripley, C.F. and K.L. Lee (1961): "Sliding friction tests on sedimentary rock specimens" Trans. 7th Int. Cong. on Large Dams, V.4, p657.

Romero, S.U. (1968): "In-situ direct shear tests on irregular surfaced joints filled with clayey material" Proc. 2nd International Symposium on Rock Mechanics, Madrid, 1968, Vol. 1, pp. 189-194.

Rosenblad, J.L. (1970): "Failure modes of models of jointed rock masses" Proc. 2nd Cong. ISRM, Belgrade, V2, paper 3-11.

Ruiz, M.D., F. Tarran and C.M. Nieble (1968a): "Model studies of stress distribution in direct shear tests" Preliminary Report, Instituto de Pesquisas Tecnologicas, S. Paulo.

Ruiz, M.D., F.P. Camargo, N.F. Midea and C.M. Nieble (1968b): "Some considerations regarding the shear strength of rock masses" Proc. 2nd Int. Symp. on Rock Mech., Madrid, (ISRM) pp.159-169.

St. John, C.M. (1971): "Numerical and observational methods of determining the behaviour of rock slopes in open cast mines" Imperial College, PhD Thesis.

St. John, C. (1972): "Finite element analyses of two and three dimensional jointed structures - computer programmes" Imperial Coll. Rock Mech. Res. Rep. n13 and appendix.

Schneider, H.J. (1974): "Rock friction - a laboratory investigation" Proc. 3rd Cong. ISRM, Denver, V2A, p311.

Snow, D.T. (1968): "Fracture deformation and changes of permeability and storage upon changes of fluid pressure" Quarterly Colorado School of Mines, V.63, n.1, P.201.

Tourenq, C., D. Fourmaintraux and A. Denis (1971): "Propagation des ondes et discontinuites des roches" Proc. Int. Symp. on Rock Fracture, Nancy (ISRM).

Tulinov, R. and L. Molokov (1971): "Role of joint filling material in shear strength of rocks" Proc. Symposium on Rock Fracture, Nancy, France 1971, paper 11-24.

Walker, P.E. (1972): "The shearing behaviour of a jointed rock model" Ground Eng. V5, p24.

Withers, J.H. (1964): "Sliding resistance along discontinuities in rock masses" PhD Thesis, Univ. of Illinois.

Wittke, W., W. Rodatz and M. Wallner (1972): "Three dimensional calculation of the stability of caverns, slopes, and foundations in anisotropic, jointed rock, by means of the finite element method" Deutsche Geotechnik, V1, n1.

Zienkiewiez, O.C., S. Valliappan and I.P. King (1968): "Stress analysis of rock as a no tension material" Geotechnique, V18, p56.

Zienkiewiez, O.C., B. Best, C. Dullage and K.G. Stagg (1970): "Analysis of non-linear problems in rock mechanics with particular reference to jointed rock systems" Proc. 2nd Cong. ISRM, Belgrade, V3, paper 8-14.

Zienkiewiez, O.C. and G.N. Pande (1977): "Time dependent multilaminate model of rocks - a numerical study of deformation and failure of rock masses" International Journal for Numerical and Analytical Method in Geomechanics. Vol. 1, No.4 219-247.

APPENDIX III : INPUT COMMANDS FOR RBM AND EXAMPLE RUN

III-1 INPUT COMMANDS

This list of input cards must be read in conjunction with Section 2.4, which explains the use of the program in more detail. At present the input format is in fixed columns, unlike program SDEM, where free format can be used. Each card image consists of a command word, starting in column one, followed by a number of parameters, in successive 10-column fields, the first field starting in column 11.

DATA SET 1

COMMAND	PARAMETERS	COMMENTS
RESTART		Restart an old run from logical Unit 1. Go to data set 2.
START		Start a new run. This card must be followed by the two following cards:

1) heading card - any characters may be used
2) data card in (4I10,F10.0) format, giving:

NBLOKM = maximum number of blocks to be used
IBOXES = number of boxes in X-direction
JBOXES = number of boxes in y-direction
IBSIZE = size of single box (boxes are square)
TFRAC = fraction of critical time-step

DATA SET 2

COMMAND	PARAMETERS	COMMENTS
CREATE	NC, RHO, IFIX	Create a block with NC corners, density ρ and IFIX≠0 for a fixed block. This card must be followed by other card(s) giving NC corner coordinates: (X(I),Y(I),I=1,NC) in (8F10.0) Format. The coordinates must be given in a clockwise direction. A CREATE command must not be given after a CYCLE Command has been given.

COMMAND	PARAMETERS	COMMENTS
DUMP	LOC1, LOC2	Dump main array from LOC1 to LOC2. If LOC2 <0, dump by boxes, blocks & contacts.
CYCLE	NCYC	Execute NCYC time-steps
STOP		STOP. Saves problem on logical unit 1 unless a serious error has occurred.
PLOT		Plot current mesh.
GRAVITY	GRAVY, GRAVX	y and x accelerations due to gravity. Default values are GRAVY = -9.81 GRAVX = 0.0
STIFFNESS	STIFN, STIFS	Normal and shear contact stiffness. Default = 1×10^8.
DAMPING	$\lambda_{min}, f_{min}, I_\alpha, I_\beta$	λ_{min} } Rayleigh damping terms, f_{min} } defined in Section 2.4.6. If I_α = 1, mass-proportional constant α is set to zero. If I_β = 1, stiffness-proportional constant β is set to zero. If $I_\alpha = I_\beta$=0, normal Rayleigh damping is used.
FRICTION	FRIC	Friction coefficient for all edges.
ZERO		Set all velocities to zero.
RSET	IADR, VAL	Set A(IADR) to VAL.
ISET	IADR, IVAL	Set IA(IADR) to IVAL.
CHECK		Momentum and energy printout
LOAD	FX, FY, NB	Applies x and y forces to block NB centroid.

EXAMPLE RUN OF RBM, WITH INPUT DATA AND OUTPUT LISTING.

THE RUN IS FOR VALIDATION 5A, DESCRIBED IN CHAPTER 2.

INPUT DATA FOR EXAMPLE RUN OF RBM

```
START
VALIDATION 5 ... BLOCK FALLING ONTO PLANE
         2                                                0.1
        10        10       100
         3         1         1
CREATE
     100.0    100.0     900.0     900.0     100.0
         4         1         0
CREATE
     700.0    500.0     700.0     600.0     600.0     600.0     500.0
GRAVITY      -9.81       0.0
STIFFNESS   1.0E7     1.0E7
DAMPING        0.5       5.0         1         0
FRICTION     0.300
PLOT
CYCLE         2500
DUMP            -1
STOP
```

OUTPUT LISTING FOR EXAMPLE RUN OF RBM

```
                                RBM - RIGID BLOCK MODEL.
                                ------------------------

VALIDATIO. 3 ... BLOCK FALLING OUTO FLOOR

                                PRESENT NUMBER OF BLOCKS     2

                                BOXES IN X-DIRECTION        10
                                BOXES IN Y-DIRECTION        10
                                TOTAL NUMBER OF BOXES      100

                                SIZE OF SINGLE BOX         100
                                PROBLEM SIZE IN X-DIRN    1000.
                                PROBLEM SIZE IN Y-DIRN    1000.

                                TIMESTEP MULTIPLIER       0.100

*** CREATE                 4          1.0          1                                                                    ***
( 0.1000E+03,  0.1000E+03 ) (  0.4000E+03,  0.3000E+03 ) (  0.9000E+02,  0.1000E+03 ) (
XC,YC,*ASS,*01 :   6.333E+07   1.667E+02   8.000E+04   3.022E+09
*** CREATE                 4          1.0          0                                                                    ***
(  0.7000E+03,  0.5000E+03 ) (  0.7000E+03,  0.6000E+03 ) (  0.5000E+03,  0.6000E+03 ) (  0.8000E+03,  0.5000E+03 )
XC,YC,*ASS,*01 :   7.000E+02   5.500E+02   1.000E+04   1.667E+07
*** GRAVITY          -9.81        0.0                                                                                  ***
*** OUTPUTS          1.067      1.067                                                                                  ***
*** DAMPING             0.5        5.0          1          0                                                          ***
        MASS DAMPING TERM SET TO ZERO
*** FRICTION           0.300                                                                                          ***
*** PLOT                                                                                                              ***
*** CYCLE              2500
                                TIME INCREMENT =  6.3246E-03
                                DRIFT CORRECTION REQUIRED
*** DUMP                -1                                                                                            ***
-------------------------------------------------------------------------------------------------------------------------
        M1         M2         M3         M4         M5         M6         M7      NBLOCKS      NCYC      NEXPT     NCYCLE
         1          3         64        104        185        187       3000            2      2500        213       2500
        BOX      BLOCK     CENTER
         14         1          1
         15         2          2
         24         1          3
         25         2          5
         26         2          1
         36         2          4
         46         1          2
BLOCK DATA----------
 NB IF NC        XC         YC       THETA       XDOT       YDOT       TDOT         DX         DY      DTHETA       AREA    INERTIA
              XFSUM      YFSUM       MSUM       XLOAD      YLOAD        COS        SIN

  1  3  3  6.33E+02  1.67E+02  0.00E+01  0.00E+01  0.00E+01  0.00E+01  0.00E+01  0.00E+01  0.00E+01  1.00E+04  5.00E+09
           3.79E+07 -2.56E+03 -1.04E+10  0.00E+01  0.00E+01  1.00E+00  0.00E+01 -5.55E+00 -8.67E+01  8.25E+02  2.00E+02
           1.33E+02  2.00E+03  2.67E+02 -8.67E+01  8.00E+02

  2  0  4  5.17E+02  2.58E+02 -1.67E-07 -2.40E+01 -5.99E+00  8.97E-07 -1.52E-01 -3.77E-02  4.41E-09  1.00E+04  1.67E+07
           4.62E+03  9.93E+06  3.52E+04  0.00E+01  0.00E+01 -2.44E-01  9.72E-01 -5.00E+01 -5.00E+01  1.00E+02 -5.00E+01
           5.00E+01  1.00E+02  5.00E+01  5.00E+01  1.00E+02  5.00E+01 -5.00E+01  1.00E+02
CONTACT DATA----------
        NBE        FRE        NPE        NFC        NBC       LINK
          S          N         FN         FS        SIN        COS        XCF        YCF

          1          0          1          1          2        187
  0.00E+01   0.00E+01   5.27E+04  -9.82E+03   2.43E-01   9.70E-01   5.78E+02   2.20E+02

          1          0          1          2          2          0
  0.00E+01   0.00E+01   6.25E+04  -1.87E+04   2.43E-01   9.70E-01   4.81E+02   1.95E+02
-------------------------------------------------------------------------------------------------------------------------
*** STOP                                                                                                              ***
                                TOTAL CYCLES       2500
                                NO. UPDATES         418
                                A RESTART FILE HAS BEEN WRITTEN
```

APPENDIX IV: SUBROUTINE GUIDE TO RBM AND PROGRAM CHANGES

IV-1: SUBROUTINE GUIDE

RBM	Main program - calls SETUP, NEXT and CYCLE.
SETUP	Reads in initial problem definition, sets up data partitions and initialises data. It is called once only from RBM.
NEXT	Reads in commands and takes appropriate action. NEXT returns to RBM only when a number of cycles (time-steps) are to be executed. Before doing this for the first time, BOX is called.
BOX	When all blocks have been created, and some calculation cycles have been requested, BOX is called to perform the initial classification of block corners into boxes.
CYCLE	This is the driver for the main claculation cycle, and calls FORD and MOTION repeatedly for all blocks.
MOTION	Applies the law of motion for a single block. Velocities and displacements are updated from known centroid forces.
FORD	Applies the constitutive laws for a single block. Contact forces are evaluated from known displacements.
REBOX	Re-maps corners into boxes if necessary. REBOX is called from CYCLE, and is triggered when a block crosses an integer boundary.
UPDAT	Contacts are detected by UPDAT, which is called from CYCLE when cumulative displacements have exceeded a given limit.
GLX	Global x-coordinates from local.
GLY	Global y-coordinates from local.
DUMP	Printout of memory or lists - called from NEXT.
BPLOT	Plots a "snapshot" of the blocks (called from NEXT).
CHECK	Momentum and energy calculation (called from NEXT).
FINISH	Orderly termination of the program.

IV-2: PROGRAM CHANGES

Changes made to RBM since pre-release version 1.0 (September 1977). Please refer to listing in Appendix XII for line numbers.

1. Program RBM

i) Line 9: 62 in data statement (was 63).

2. Subroutine SETUP

i) This routine has been completely changed. It now reads just one card, instead of a set. The changes are mainly due to the elimination of co-ordinate scaling and shifting logic.

3. Subroutine NEXT

i) The command "LOAD" has been implemented, necessitating changes to lines 8, 16 and 123 to 127.

ii) Calls to date and time routines have been removed, necessitating changes to the WRITE statement on line 10 and format statement on line 135.

iii) All references to co-ordinate scaling and shifting have been removed, necessitating removal of lines preceding line 40, and modification of lines 44 to 48.

iv) DECODE statement on line 20 has been modified.

v) "AREA" in FORMAT statement 2002 has been changed to "MASS".

vi) Lines 99 to 113 changed to include full Rayleigh damping.

4. Subroutine BOX

i) Line 21: IBOXES replaces JBOXES.

5. Subroutine REBOX

i) Line 20: IBOXES replaces JBOXES.

6. Subroutine MOTION

i) Line 24: ABS inserted before B(5).

7. Subroutine UPDAT

i) Line 8: 0.001 changed to 1.0.

TOLB initialised to -0.01.

ii) Line 79: ICL = I2C + 20 replaces ICL = IA(I2C) + 20.

8. Common block / CBLOCK /

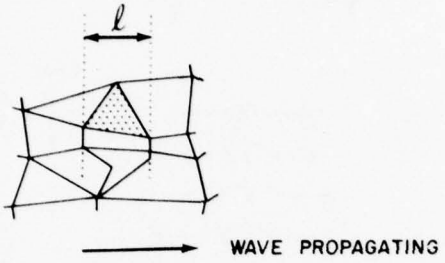
Changes made to reflect:

i) elimination of co-ordinate scaling and shifting;

ii) inclusion of mass damping.

APPENDIX V : DERIVATION OF EFFECTIVE MASSES - PROGRAM SDEM

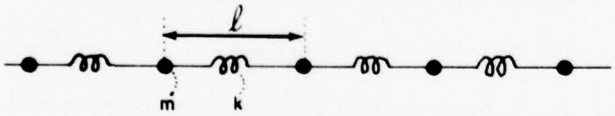
Consider a "continuum" of blocks through which a plane wave is propagating (assume the joints to be very stiff):



Let E* be the modulus appropriate to the type of wave (for example, G for shear waves, or $K + \frac{4}{3}G$ for p-waves). The velocity of propagation in a true continuum will be:

$$\sqrt{\frac{E^*}{\rho}} \quad \text{where } \rho \text{ is the density} \quad \ldots \quad V.1$$

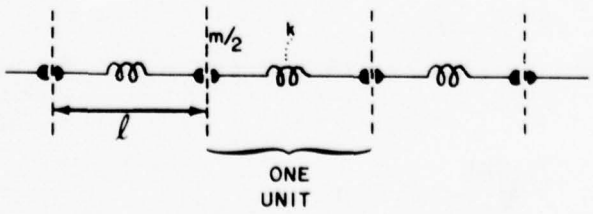
Consider now a line of equal masses and springs:



The velocity of wave propagation in this line is given by

$$C = \ell\sqrt{\frac{k}{m}}$$

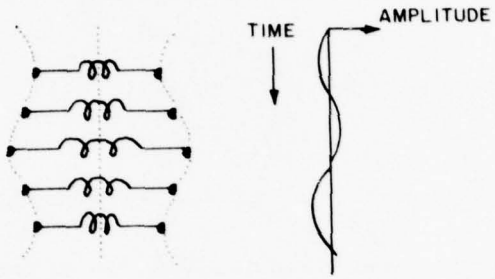
Suppose we now view the line as composed of individual mass/spring units:



Individual units, if taken individually, will oscillate at a natural frequency of

$$\omega_o = \sqrt{\frac{2k}{m/2}} = 2\sqrt{\frac{k}{m}} \quad \text{with}$$

the following mode:



It is now possible to express the velocity of propagation in a mass/spring line in terms of the natural frequency of an individual unit:

$$C = \frac{\ell\omega_o}{2} \qquad \ldots \text{ V.2}$$

If the "continuum" of blocks is now viewed as a mass/spring system, we can establish an equivalence between the two systems. The natural frequency of individual blocks follows from the equations given in Section 3.4. In simplified form these are:

$$\dot{\varepsilon} := \dot{\varepsilon} - \frac{\sigma\Delta t}{m^e}$$

$$\underset{\sigma}{I} := \underset{\sigma}{I} + \dot{\varepsilon}E^*\Delta t$$

The natural frequency of this single degree of freedom system is:

$$\omega_b = \sqrt{\frac{E^*}{m^e}}$$

If these individual blocks are connected together to form a line, we can write down the velocity of propagation in terms of ω_b , using equation V.2.

$$C = \frac{\ell}{2}\sqrt{\frac{E^*}{m^e}}$$

But this must also be equal to the continuum wave speed given in V.1.

$$\frac{\ell}{2}\sqrt{\frac{E^*}{m^e}} = \sqrt{\frac{E^*}{\rho}}$$

$$m^e = \frac{\ell^2 \rho}{4}$$

$$\boxed{m^e = \frac{\ell^2 m}{4A}} \qquad \ldots \text{ V.3.}$$

where m = mass of block

A = area of block

The derivation above makes the assumption that all blocks having the same overall length in the direction of propagation will all have the same effective mass in that direction, and hence the same natural frequency. This seems physically reasonable, since for a plane wave to remain plane, the response time for blocks spanning the same distance should be the same, so that blocks oscillate in phase.

The four effective masses needed in Section 3.4 are:

$$m^{e}_{(i)} = \frac{m}{4A}(\ell_i)^2$$

where ℓ_i are the maximum block widths.

The effective masses derived above must necessarily be approximate, and are not intended for use in wave propagation problems. However, for dynamic problems where the wavelengths are larger than the typical overall problem dimensions, the formulation should be reasonable. In nearly-static problems, of course, the effective masses play no part.

APPENDIX VI : STRESS ROTATION CORRECTION TERMS

VI-1 Program SDEM

If a block, together with the forces acting on it, is rotated, the internal stresses, referred to local axes, are unaffected. However the stresses expressed in global coordinates will change. It is common in lagrangian finite-difference codes to apply stress rotation correction terms to the zone stresses at each time step in order to allow for this apparent stress change due to rotation. The correction terms are derived from the tensor transformation equations used to determine stresses in a new coordinate system (indicated by a bar) when they are known in an old system. These equations are as follows:

$$\bar{\sigma}_{ij} = \sigma_{\alpha\beta} \frac{\partial \bar{x}_i}{\partial x_\alpha} \frac{\partial \bar{x}_j}{\partial x_\beta} = \sigma_{\alpha\beta} J_{i\alpha} J_{j\beta} \qquad \ldots \text{ VI.1}$$

where J_{ij} is defined in section 3.4.

When the angle $\Delta\theta$ between the coordinate systems is small,

$J_{ij} = \delta_{ij} + \dot{R}_{ij} \Delta t$, discarding second order terms in $\Delta\theta$, and noting that $\Delta\theta = \theta\Delta t$, with $\dot{R}_{ij}$ defined in section 3.4.

Using this expression for J_{ij} in VI.1 gives:

$$\bar{\sigma}_{ij} = \sigma_{\alpha\beta}(\delta_{i\alpha} + \dot{R}_{i\alpha}\Delta t)(\delta_{j\beta} + \dot{R}_{j\beta}\Delta t)$$

$$= \sigma_{ij} + (\sigma_{\alpha j}\dot{R}_{i\alpha} + \sigma_{i\beta}\dot{R}_{j\beta})\Delta t \qquad \ldots \text{ VI.2}$$

Format. The coordinates must be given in a clockwise direction. A CREATE command must not be given after a CYCLE Command has been given.

These new stresses correspond to a change in coordinate axes. However, in the case of a zone rotating, the coordinate axes remain fixed, and the zone rotates relative to the axes. Using a negative si in VI-2, and changing the repeated indices, gives the changes in zone stresses due to zone rotation:

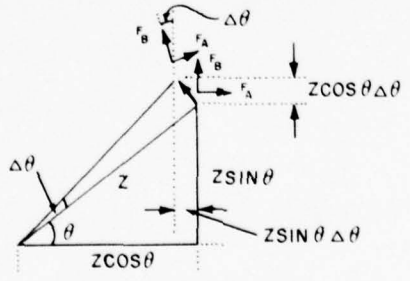
$$\Delta\sigma_{ij} = -(\sigma_{kj}\dot{R}_{ik} + \sigma_{ik}\dot{R}_{jk})\Delta t \qquad \ldots \text{ VI-3}$$

In the program SDEM, the block stresses are derived from the discrete forces acting on the block boundaries, using the equation:

$$\overset{A}{\sigma}_{ij} = \frac{\sum^{c} \overset{c}{F}_i \overset{c}{X}_j}{A} \qquad \ldots \text{ VI-4}$$

The following analysis was carried out to verify that giving a rotation to the discrete block/force system resulted in the same stress rotation correction terms as VI-3, derived for a continuum.

Consider two forces acting on a block that rotates through an angle $\Delta\theta$:



The stress that would be calculated by III-4 before and after the rotation are as follows:

stress	before rotation	after rotation
σ_{11}	F zcosθ	(F cos$\Delta\theta$ - F sin$\Delta\theta$)(zcosθ - zsin$\theta\Delta\theta$)
σ_{12}	F zsinθ	(F cos$\Delta\theta$ - F sin$\Delta\theta$)(zsinθ + zcos$\theta\Delta\theta$)
σ_{21}	F zcosθ	(F cos$\Delta\theta$ + F sin$\Delta\theta$)(zcosθ - zsin$\theta\Delta\theta$)
σ_{22}	F zsinθ	(F cos$\Delta\theta$ + F sin$\Delta\theta$)(zsinθ + zcos$\theta\Delta\theta$)

On multiplying out, and discarding second order terms, the differences between the stresses before and after rotation become:

$$\Delta\sigma_{11} = -(\sigma_{12} + \sigma_{21})\Delta\theta$$

$$\Delta\sigma_{12} = (\sigma_{11} - \sigma_{22})\Delta\theta$$

$$\Delta\sigma_{21} = (\sigma_{11} - \sigma_{22})\Delta\theta$$

$$\Delta\sigma_{22} = (\sigma_{12} + \sigma_{21})\Delta\theta$$

It can be seen that these correction terms are identical with those of VI-3. The terms are used in Section 3.4 in the calculation of internal stresses.

VI.2 Program DBLOCK

The correction of internal stresses due to rotation in program DBLOCK is identical to that described for SDEM (formula VI.3).

However the contact forces require a treatment different from that of the stresses since the correction is required to account for two mechanisms: a) the rotation of the surface normal, and b) the change in surface normals that occurs when a grid-point moves from one edge to another.

Let n_i and $\bar{n}_i$ be the old and new unit normals in the global coordinate system. Let f_i and $\bar{f}_i$ be the old and new components of the contact force in local coordinates (f_1 = shear force, f_2 = normal force). Lef F_i be the components of the contact force in the global coordinate system.

$$f_i = a_{ij} F_j$$

$$\bar{f}_i = \bar{a}_{ij} F_j$$

where $a_{2j} = n_j$, $a_{11} = a_{22}$, $a_{12} = -a_{21}$

$\bar{a}_{2j} = \bar{n}_j$, $\bar{a}_{11} = \bar{a}_{22}$, $\bar{a}_{12} = -\bar{a}_{21}$

Since a_{ij} is orthonormal,

$$a_{ij} \cdot a_{kj} = \delta_{ik}$$

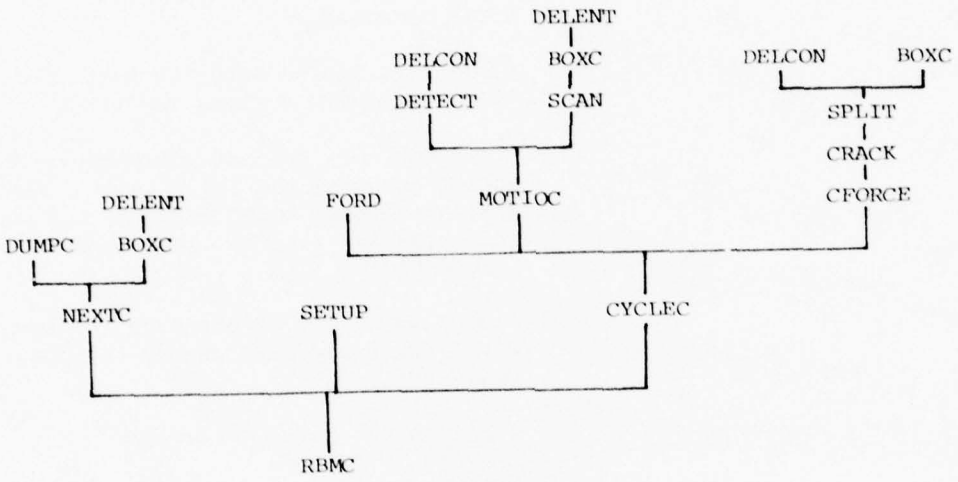
Then:

$$\bar{f}_i = \bar{a}_{ik} a_{jk} f_j$$

APPENDIX VII: SUBROUTINE GUIDE, PROGRAM RBMC

RBMC	Main program
SETUP	Initialization and input of problem size and constants.
NEXT	Processing of input commands. A return is made whenever command CYCLE is given.
BOX	Classifies blocks into the boxes that their edges and corners map into.
CYCLE	CYCLE controls the main calculation cycle; MOTIOC computes the law of motion for a single block; FORDC computes the constitutive law for the contacts around a single block.
CFORCE	Assembles a list of contacts for each block for subsequent use by the crack criterion.
CRACK	Embodies the crack criterion.
SPLIT	Introduces the crack into the selected block, and performs all necessary housekeeping duties.
BPLOT	Plots a "snapshot" of the problem geometry.
SCAN	Determines which boxes each block edge maps into.
DELENT	Deletes obsolete box entries.
DETECT	Detects and updates contacts for the two edges adjacent to a corner.
DELCON	Deletes obsolete contacts.
DUMP	Prints a dump of memory or linked list data.
LIMIT	Checks whether a request for more memory is possible.
EMPTYC	Creates a new contact data group.
EMPTYD	Creates a new "double".
FINISH	Terminates the program.

The main subroutines are called as follows:



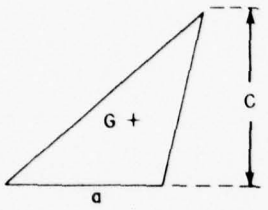
APPENDIX VIII: LIST OF SYMBOLS USED IN CHAPTER 6

The main symbols used in the text are:

$d(m\ n)$	=	distance from m to n
F_i	=	i^{th} component of the force
F_N	=	normal force
F_S	=	tangential force
K_E	=	kinetic energy
K_{NL}	=	normal stiffness during loading
K_{NU}	=	normal stiffness during unloading
K_S	=	shear stiffness
m_N	=	mass of node N
M_M	=	mass of zone M
n_i	=	i^{th} component of the unit normal
$O(\cdot)$	=	'of the order of'
R_{ij}	=	components of the rotation tensor
s	=	arc length
u_i	=	displacement tensor
$\bar{v}_N$	=	velocity vector for node N
v_p	=	P-wave velocity
x,y	=	two-dimensional orthogonal coordinate system
α,β	=	distance ratios
δ_{ij}	=	Kronecker delta
δ_N	=	normal displacement
δ_S	=	tangential displacement
Δ	=	increment
ε_{ij}	=	components of the strain tensor
ε_v	=	volumetric strain
λ,μ	=	Lamé constants
ρ	=	mass density
τ_{ij}	=	components of the stress tensor
τ^e_{ij}	=	elastic part of the stress tensor

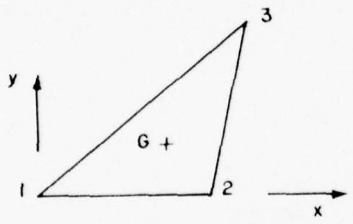
APPENDIX IX : PROPERTIES OF TRIANGLES USED IN PROGRAM DBLOCK

(1) The center of gravity of a triangle is at one third of the distance between any side and the opposite corner.



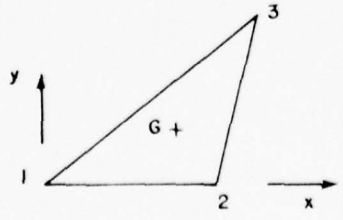
$$y_G = \frac{\int_0^c \frac{c-y}{c} y \, dy}{\frac{ac}{2}} = \frac{c}{3} \quad \text{Q.E.D.}$$

(2) The coordinates of the center of gravity of a triangle are the average of those of its corners.



$$\frac{y_1 + y_2 + y_3}{3} = \frac{y_3}{3} = y_G \quad \text{Q.E.D.}$$

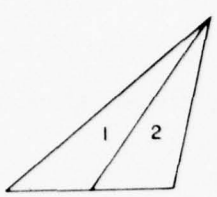
(3) The center of gravity of a triangle does not change if its mass is evenly lumped in its three corners.

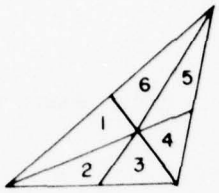


TO PRESERVE THE FIRST MOMENT: $m_3 y_3 = M y_G$

$$\frac{m_3}{M} = \frac{y_G}{y_3} = \frac{1}{3} \quad \text{Q.E.D.}$$

(4) Each median divides a triangle in two triangles of equal area. The three medians divide a triangle in six triangles of equal area.





Obvious since the center of gravity is the intersection of all lines which split the mass of the triangle in two equal parts.

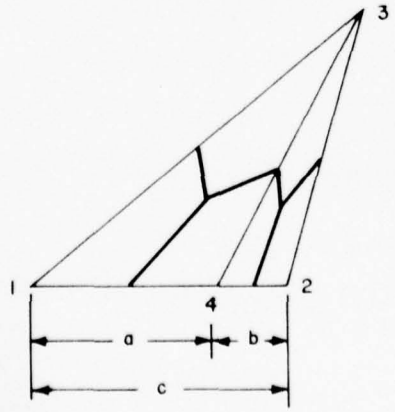
(5) The center of gravity of a triangle is the point of intersection of its medians.

Obvious

(6) The three medians provide a natural lumping of the mass of the triangle in its three corners which does not change the center of gravity.

Obvious

(7) If a new grid point (4) is created and the triangle $\overset{\triangle}{(123)}$ split into two triangles $\overset{\triangle}{(143)}$, $\overset{\triangle}{243)}$ the assigned masses should be:



Let M = mass $\overset{\triangle}{(123)}$ $\frac{a}{c} = \alpha$ $\frac{b}{c} = \beta$

$m_1 = \frac{1}{3}$ mass $\overset{\triangle}{(134)} = \alpha\frac{M}{3}$

$m_2 = \frac{1}{3}$ mass $\overset{\triangle}{(234)} - \beta\frac{M}{3}$

$m_3 = \frac{1}{3}$ mass $\overset{\triangle}{(134)} + \frac{1}{3}$ mass $\overset{\triangle}{(234)} = \frac{M}{3}$

$m_4 = \frac{1}{3}$ mass $\overset{\triangle}{(134)} + \frac{1}{3}$ mass $\overset{\triangle}{(234)} = \frac{M}{3}$

APPENDIX X : INPUT CARDS FOR PROGRAM DBLOCK

The required input stream is as follows:

1.	One card (6I10)	NMAX, MMAX, IMAX, IFLAG, KTOT, ICONT
	if IFLAG = 1, GO TO 6 if IFLAG = 2, GO TO 12	
2.	NMAX cards (4F10.0)	(X(N), Y(N), XD(N), YD(N), N=1,NMAX)
3.	NMAX cards (9F10.0)	(XX(M), YY(M), XY(M), AM(M), LAME1(M), LAME2(M), COHES(M), TANPHI(M), TANPSI(M), M=1,MMAX)
4.	NMAX cards (<IMAX>I10)	((MAN(N,I), I=1,IMAX), N=1,NMAX))
5.	MMAX cards (3I10)	((NAM(M,J), J=1,3), M=1,MMAX))
6.	LCONT cards (3I10, F10.0)	(IMP(L), N2M(L), N3M(L), XNC(L), L=1,LCONT)
7.	One card (5F10.0)	STNL, STNU, DN, STSH, XMU
8.	One card (I10, 4F10.0, 2I10)	NITER, FRAC, GRAV, ARAT, TFRAC, NPRI, NPLOT
9.	Two cards (8I10)	(ITPL(I),I=1,10)
10.	One card (2F10.0)	SCALEX, SCALEY
11.	One card (2F10.0)	RMIN, FMIN
	END OF INPUT STREAM	
12.	One card (I10)	NBLOCK
13.	One card (F10.0)	SCALE

The next two cards are repeated once for each block

14.	One card (I10)	NCORN

NBLOCK times

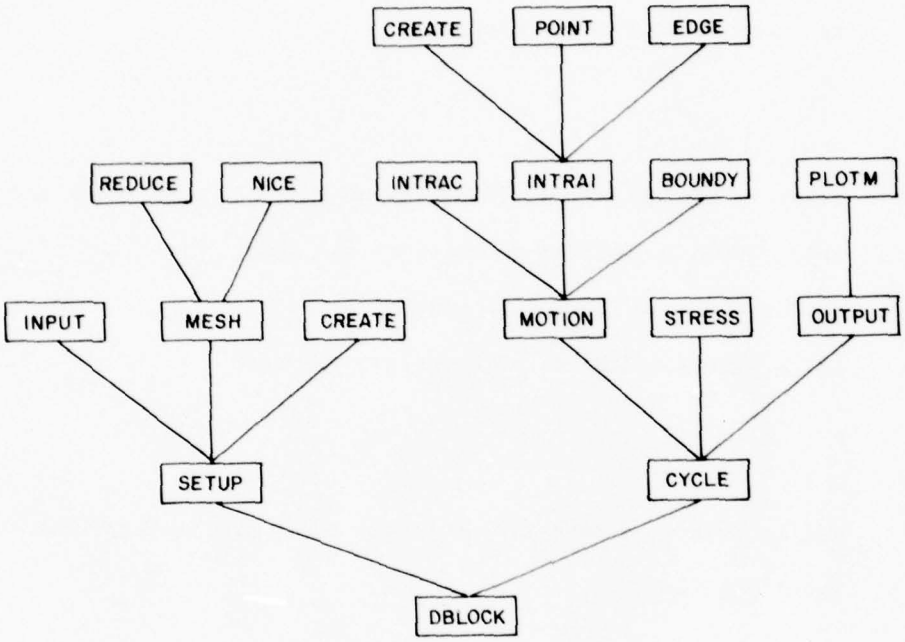
15.	NCORN cards (3I10)	(LIST(1,I), X(I), Y(I), I=1,NCORN)
16.	One card (F10.0)	AMAXL

GO TO 6

The meaning of the FORTRAN variables is given in Appendix XVI.

APPENDIX XI: SUBROUTINE GUIDE - Program DBLOCK

The following tree illustrates the manner in which the different routines are called:



A brief description of each subroutine follows:

DBLOCK

- Is the driving routine for SETUP and CYCLE.

SETUP

- Reads in and prints all the information needed by the program.
- Initializes all variables.
- Assigns grid points masses.
- Computes the time step.
- In problems where more than one block exists SETUP also calls CREATE to create new nodes at the contacts as required.

CYCLE

- Drives the main interaction loop.
- Calls MOTION, STRESS every cycle; OUTPUT when needed.
- Prepares files for the plotter.

MESH

- Reads in the corners defining the geometry of each block and other information needed for creation of the mesh.
- Decomposes each block into triangles.
- Calls REDUCE and NICE, wh. refine the mesh.

REDUCE

- Splits triangles into two until all edges are smaller than a specified length.

NICE

- Rezones each node until the coordinates of all nodes (excluding those on the edge of the block coincide with the average of the coordinates of the surrounding nodes.

INPUT

- Creates a mesh consisting of a regular distribution of triangles.

CREATE

- Creates a new node and zone when a contact requires it.
- Redistributes mass and momentum while preserving their balance.

- Assigns mechanical properties to the new zone.
- Establishes the required logical links for the contact created and modifies those of other contacts which require changes due to the appearance of the new one.

MOTION

- Stresses are added to get the resultant force at each grid point. This provides accelerations; time integration yields velocities.

- Velocities are corrected by either one of two reasons:
  (a) if there is interaction between blocks, INTRAC is called;
  (b) boundary conditions in velocity are applied.

- Time integration of velocities yields displacments.

STRESS

- Computes incremental rotations and strains from velocities for each individual zone.

- Obtains the elastic stress increment, the viscous one (when damping exists) and the modification due to rotation; all 3 are compounded to obtain the new stresses.

OUTPUT

- Prints the main variables at pre-selected iteration numbers.

- Calls PLOTM to create a plot of the mesh at pre-selected iteration numbers.

- Creates files for plotting time histories of variables.

INTRAC

- Computes accelerations and velocities due to forces of interaction across blocks.

INTRA1

- This routine is the most complex in the program. A flow-chart of its functions is presented in Figure XI.1 to help visualize the procedures. As can be seen in the flow-chart, the routine is basically a loop extending to all contacts. For each one, relative displacements are computed, and the appropriate routine is called to calculate interacting forces from those displacements. Besides those functions, INTRA1 also breaks contacts, "tickles" nodes, deletes them or calls CREATE to provide new ones where needed. All the logic for the different types of rezoning is embodied in this routine.

BOUNDY

- Applies the boundary conditions in the form of velocities. These must be explicitly built into BOUNDY by means of FORTRAN statements.

POINT

- Computes forces of interaction from relative displacements across contacts using the laws in Section 6.2.6.1.

EDGE

- Computes forces of interaction from relative displacements across contacts using the laws in Section 6.2.6.2.

PLOTM

- Creates a plot of the complete finite difference mesh.

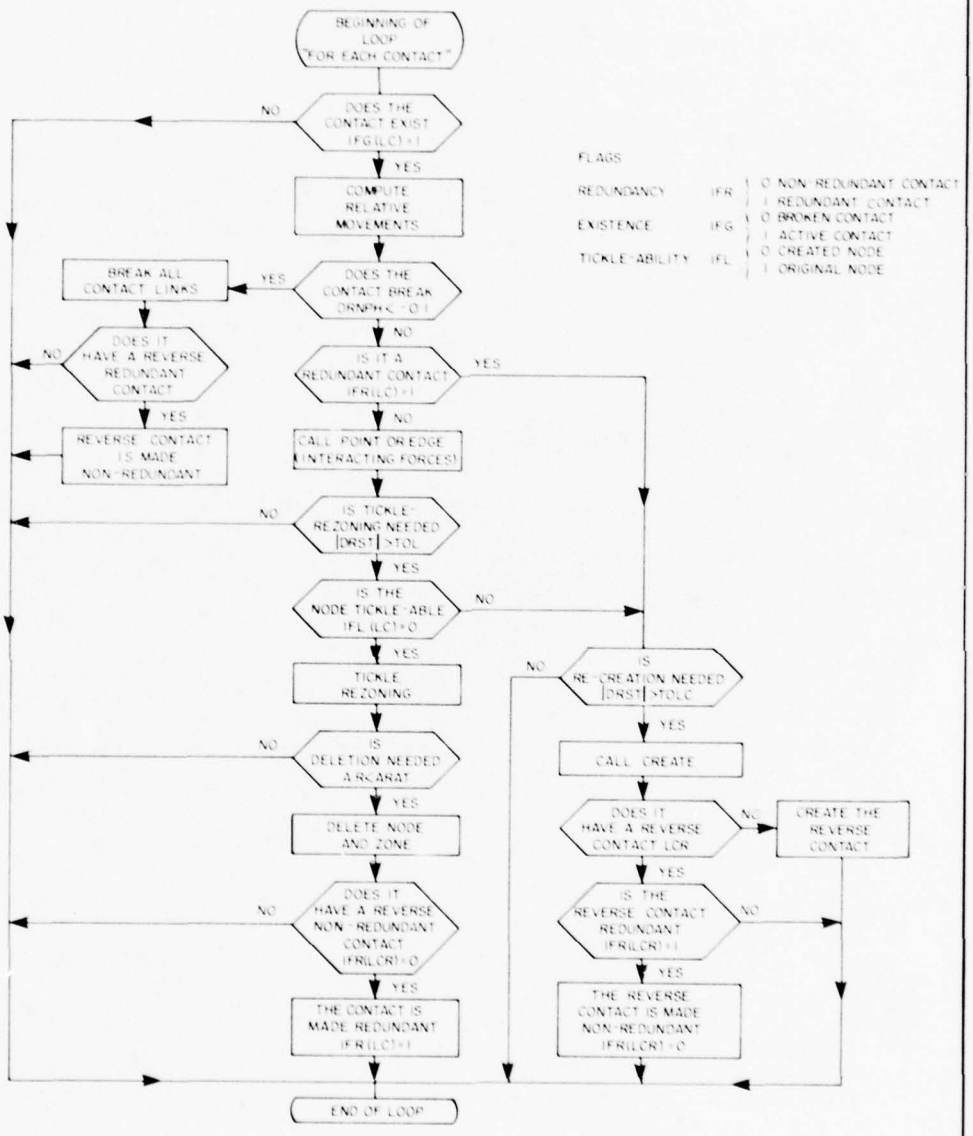


FIG. XI-1: FLOW CHART OF THE ROUTINE INTRA1

DAMES & MOORE

APPENDIX XII: LIST OF VARIABLES AND PROGRAM LISTING FOR RBM

HED (20)	User supplied problem heading.
NBLOKM	Maximum number of blocks allowed. Sets storage according to this value.
NBOXES	Number of boxes (= IBOXES * JBOXES)
M1,M2...M7	Storage boundary markers.
NBLOKS	Number of blocks currently defined.
NCYC	Number of cycles requested on current CYCLE command.
MCYCLE	Current cycle number.
NEMPT	Pointer to start of empty list.
RFLAG	Reboxing flag. Indicates when reboxing is required.
UFLAG	Updating flag. Indicates when updating of coordinates is required.
EFLAG	Error flag. Indicates when an error condition has been detected. (Currently not used)
TFRAC	Fraction of critical timestep.
TDEL	Calculation timestep. TDEL is equal to the critical timestep* TFRAC.
IBOXES	Number of boxes in I-direction.
JBOXES	Number of boxes in J-direction.
BSIZE	Size of one box (boxes are square).
XSIZE	Width of box area (=IBOXES * BSIZE)
YSIZE	Height of box area (=JBOXES * BSIZE)
UDMAX	Maximum velocity detected in the current calculation cycle.
UMOST	Fictitious displacement, formed by summing UDMAX*TDEL and used to trigger the update flag.

STIFN	Normal contact stiffness: Currently is the same for all contacts.
STIFS	Shear contact stiffness. Currently is the same for all contacts.
FRIC	Contact coefficient of friction. Currently is the same for all contacts.
BETA	Stiffness-proportional damping constant.
BDT	BETA/TDEL
ALPHA	Mass-proportional damping constant
CON1 CON2	Constants used in mass-proportional damping.
GRAVX	Acceleration due to gravity in x-direction Units are acceleration units, not G's.
GRAVY	Acceleration due to gravity in y-direction.
LOC 1 LOC 2	Addresses passed to the DUMP routine for indicating the information to be printed.
NUPDAT	Counter of number of updates performed.
LBLOCK	Length of the commm block /CBLOCK/.

PROGRAM LISTING - RBM

```
FORTRAN IV-PLUS V02-51          17:00:21     16-FEB-78          PAGE 1
RBM.LST          /14/TR:BLOCKS/WR

0001            PROGRAM RBM
                C
                C-------------------------------------------------------------
                C------------------- RIGID BLOCK MODEL. ----------------------
                C------------------- FREELY TRANSLATED INTO FORTRAN BY -------
                C------------------- P.J.BERESFORD, FROM AN ORIGINAL PROGRAM -
                C------------------- BY P.A.CUNDALL. ------------------------
                C------------------- DAMES AND MOORE, LONDON. ---------------
                C------------------- VERSION 1.1, JANUARY 1978. -------------
                C-------------------------------------------------------------
                C
0002            COMMON A(3000)
0003            DIMENSION IA(1)
0004            EQUIVALENCE (A,IA)
0005            COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,M5,M6,M7,
                .                NBLOKS,NCYC,MCYCLE,NEMPT,RFLAG,UFLAG,EFLAG,
                .                TFRAC,TDEL,IBOXES,JBOXES,BSIZE,XSIZE,YSIZE,
                .                UDMAX,UMOST,STIFN,STIFS,FRIC,ALPHA,BETA,DET,
                .                CON1,CON2,NVARB,NFRAG,NBR,NPR,
                .                GRAVX,GRAVY,LOC1,LOC2,NUPDAT,LBLOCK
0006            LOGICAL RFLAG,UFLAG,EFLAG
                C
0007            DATA STIFN,STIFS,FRIC,BETA/2*1.0E8,0.0,0.0/
0008            DATA GRAVY,GRAVX/-9.81,0.0/
0009            DATA M7,MCYCLE,NUPDAT,LBLOCK/3000,0,0,62/
0010            DATA RFLAG,UFLAG,EFLAG/.FALSE.,.TRUE.,.FALSE./
                C
0011            CALL PLOTST(.025,'CM')
0012            CALL SETUP
0013         10 CALL NEXT
0014            CALL CYCLE
0015            GOTO 10
                C
0016            END
```

```
FORTRAN IV-PLUS V02-51          17:00:32     16-FEB-76          PAGE 3
RBM.LST          /14/IR:BLOCKS/WR


0001            SUBROUTINE SETUP
                C
0002            COMMON A(3000)
0003            DIMENSION IA(1)
0004            EQUIVALENCE (A,IA)
0005            COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,M5,M6,M7,
                .               NBLOKS,NCYC,MCYCLE,NEMPT,RFLAG,UFLAG,EFLAG,
                .               TFRAC,TDEL,IBOXES,JBOXES,BSIZE,XSIZE,YSIZE,
                .               UDMAX,UMOST,STIFN,STIFS,FRIC,ALPHA,BETA,BDT,
                .               CON1,CON2,NVARB,NFRAG,NBR,NPR,
                .               GRAVX,GRAVY,LOC1,LOC2,NUPDAT,LBLOCK
0006            LOGICAL RFLAG,UFLAG,EFLAG
                C
0007            WRITE(6,2000)
0008            READ(5,1000) WORD
                C
0009            IF(WORD.EQ.4HSTAR) GOTO 20
0010            IF(WORD.EQ.4HREST) GOTO 10
                C
0011            WRITE(6,3000)
0012            STOP
                C
                C--------- 'RESTART' RUN ---
0013          10 READ(1) (HED(I),I=1,LBLOCK)
0014            READ(1) (A(I),I=1,M7)
0015            WRITE(6,2001) HED,MCYCLE
0016            RETURN
                C
                C--------- 'START' RUN ---
0017          20 READ(5,1001) HED,NBLOKM,IBOXES,JBOXES,IBSIZE,TFRAC
                C
0018            NBOXES=IBOXES*JBOXES
0019            BSIZE=IBSIZE
0020            XSIZE=IBOXES*BSIZE
0021            YSIZE=JBOXES*BSIZE
                C
0022            WRITE(6,2002) HED,NBLOKM,IBOXES,JBOXES,NBOXES,
                .               IBSIZE,XSIZE,YSIZE,TFRAC
                C
                C------------------- INITIALISE SOME VARIABLES ---
0023            UMOST=0.0
0024            NBLOKS=0
0025            M1=1
0026            M2=M1+NBLOKM
0027            M3=M2
0028            RETURN
                C
0029           1000 FORMAT(A4)
0030           1001 FORMAT(20A4/4I10,F10.0)
0031           2000 FORMAT(30X,25H RBM - RIGID BLOCK MODEL./
                .          30X,25H ------------------------/)
0032           2001 FORMAT(1X,20A4//30X,30HRESTART RUN. CURRENT CYCLES ...,I10//)
0033           2002 FORMAT(1X,20A4//
                .          30X,24HMAXIMUM NUMBER OF BLOCKS,I5//
                .          30X,24HBOXES IN X-DIRECTION    ,I5/
```

```
FORTRAN IV-PLUS V02-51          17:00:32    16-FEB-78          PAGE 4
R8M.LST          /14/TR:BLOCKS/WR

                .       30X,24HBOXES IN Y-DIRECTION     ,I5/
                .       30X,24HTOTAL NUMBER OF BOXES    ,I5//
                .       30X,24HSIZE OF SINGLE BOX       ,I5/
                .       30X,24HPROBLEM SIZE IN X-DIRN   ,F6.0/
                .       30X,24HPROBLEM SIZE IN Y-DIRN   ,F6.0//
                .       30X,24HTIMESTEP MULTIPLIER      ,F6.3//)
0034      3000 FORMAT(48H *** ERROR : 'START' OR 'RESTART' CARD NOT FOUND)
          C
0035            END
```

```
FORTRAN IV-PLUS V02-51          17:00:56     16-FEB-78          PAGE 6
RBM.LST          /I4/TR:BLOCKS/WR


0001            SUBROUTINE NEXT
                C
0002            COMMON A(3000)
0003            DIMENSION IA(1)
0004            EQUIVALENCE (A,IA)
0005            COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,M5,M6,M7,
                .                NBLOKS,NCYC,MCYCLE,NEMPT,REFLAG,UFLAG,EFLAG,
                .                TFRAC,TDEL,IBOXES,JBOXES,BSIZE,XSIZE,YSIZE,
                .                UDMAX,UMOST,STIFN,STIFS,FRIC,ALPHA,BETA,BDT,
                .                CON1,CON2,NVARB,NFRAG,NBR,NPR,
                .                GRAVX,GRAVY,LOC1,LOC2,NUPDAT,LBLOCK
0006            LOGICAL REFLAG,UFLAG,EFLAG
                C
0007            DIMENSION CARD(20),WORD(20),X(50),Y(50)
                C
0008            DATA WORD /4HCREA,4HDELE,4HDUMP,4HCYCL,4HSTOP,
                .          4HPLOT,4H****,4HRSET,4HISET,4HCHEC,
                .          4HGRAV,4HSTIF,4HDAMP,4HFRIC,4HZERO,
                .          4HLOAD,4HZZZZ,4HZZZZ,4HZZZZ,4HZZZZ/
                C
                C------------------- READ NEXT CARD ---
0009           10 READ(5,1000) CARD
0010              WRITE(6,2000) CARD
0011              DO 20 I=1,20
0012              IF(CARD(1).EQ.WORD(I)) GOTO 30
0013           20 CONTINUE
0014              WRITE(6,3000)
0015              GOTO 10
                C------------------- JUMP TO APPROPRIATE CODE ---
0016           30 GOTO (100,150,200,250,300,
                .       350,400,450,500,550,
                .       600,650,700,750,800,
                .       850, 10, 10, 10, 10), I
                C
0017           40 WRITE(6,3001)
0018              GOTO 10
                C------------------- CREATE A NEW BLOCK ---
0019          100 NBLOKS=NBLOKS+1
0020              DECODE(40,1001,CARD) NC,RHO,JFIX
0021              READ(5,1002) (X(I),Y(I),I=1,NC)
0022              WRITE(6,2001) (X(I),Y(I),I=1,NC)
0023              I2=M3
0024              IA(NBLOKS)=I2
0025              IF(JFIX.NE.0) IA(I2)=1
0026              IA(I2+1)=NC
0027              A(I2+18)=1.0
0028              A(I2+19)=0.0
                C------------------- AREA AND CENTROID OF THIS BLOCK ---
0029              AREA=(X(1)-X(NC))*(Y(1)+Y(NC))
0030              YC=(X(1)-X(NC))*((Y(1)-Y(NC))*(Y(1)+2.0*Y(NC))+3.0*Y(NC)**2)
0031              XC=(Y(1)-Y(NC))*((X(1)-X(NC))*(X(1)+2.0*X(NC))+3.0*X(NC)**2)
0032              DO 110 I=2,NC
0033              AREA=AREA+(X(I)-X(I-1))*(Y(I)+Y(I-1))
0034              YC=YC+(X(I)-X(I-1))*((Y(I)-Y(I-1))*(Y(I)+2.0*Y(I-1))
                .      +3.0*Y(I-1)**2)
```

```
FORTRAN IV-PLUS V02-51          17:00:56     16-FEB-78          PAGE 7
RBM.LST          /I4/TR:BLOCKS/WR

0035            XC=XC+(Y(I)-Y(I-1))*((X(I)-X(I-1))*(X(I)+2.0*X(I-1))
               .      +3.0*X(I-1)**2)
0036          110 CONTINUE
0037            AREA=0.5*AREA
0038            YC=YC/(6.0*AREA)
0039            XC=-XC/(6.0*AREA)
0040            A(I2+2)=XC
0041            A(I2+3)=YC
0042            A(I2+11)=AREA*RHO
          C------------------- LOCAL COORDINATES FOR THIS BLOCK ---
0043            M3=M3+20
0044            A(M3)=X(1)-XC
0045            A(M3+1)=Y(1)-YC
0046            DO 120 I=2,NC
0047            A(M3+3)=X(I)-XC
0048            A(M3+4)=Y(I)-YC
0049            A(M3+2)=SQRT((A(M3+3)-A(M3))**2+(A(M3+4)-A(M3+1))**2)
0050          120 M3=M3+3
0051            A(M3+2)=SQRT((A(I2+20)-A(M3))**2+(A(I2+21)-A(M3+1))**2)
0052            M3=M3+3
          C------------------- MOMENT OF INERTIA ---
0053            RMOI=0.0
0054            IC=I2+20
0055            DO 130 NP=2,NC
0056            AREA=A(IC)*A(IC+1) + (A(IC+3)-A(IC))*(A(IC+4)+A(IC+1))
               .      - A(IC+3)*A(IC+4)
0057            AREA=0.5*AREA
0058            TEMP=A(IC)**2+A(IC+1)**2+A(IC+3)**2+A(IC+4)**2
               .      +A(IC)*A(IC+3)+A(IC+1)*A(IC+4)
0059            RMOI=RMOI+AREA*TEMP/6.0
0060          130 IC=IC+3
0061            AREA=A(IC)*A(IC+1) + (A(I2+20)-A(IC))*(A(I2+21)+A(IC+1))
               .      - A(I2+20)*A(I2+21)
0062            AREA=0.5*AREA
0063            TEMP=A(I2+20)**2+A(I2+21)**2+A(IC)**2+A(IC+1)**2
               .      +A(I2+20)*A(IC)+A(I2+21)*A(IC+1)
0064            RMOI=RMOI+AREA*TEMP/6.0
0065            A(I2+12)=RMOI*RHO
          C
0066            WRITE(6,2002) XC,YC,A(I2+11),A(I2+12)
0067            GOTO 10
          C------------------- DELETE A BLOCK ---
0068          150 GOTO 40
          C------------------- DUMP MEMORY AS REQUESTED ---
0069          200 DECODE(30,1003,CARD) LOC1,LOC2
0070            IF(LOC2.NE.0) GOTO 220
0071            LOC2=LOC1
0072            LOC1=1
0073          220 CALL DUMP
0074            GOTO 10
          C------------------- CYCLE ROUND MOTION AND FORD ---
0075          250 DECODE(20,1001,CARD) NCYC
0076            IF(MCYCLE.EQ.0) CALL BOX
0077            RETURN
          C------------------- STOP COMMAND ---
0078          300 CALL FINISH
```

```
FORTRAN IV-PLUS V02-51          17:00:56     16-FEB-78          PAGE 8
RBM.LST         /I4/TR:BLOCKS/WR

      C------------------- PLOT COMMAND ---
0079    350 CALL BPLOT
0080        GOTO 10
      C------------------- RETURN TO PHASE 1 ---
0081    400 CALL SETUP
0082        GOTO 10
      C------------------- SET REAL DATA ---
0083    450 DECODE(30,1004,CARD) IADR,VAL
0084        IF(IADR.LE.0.OR.IADR.GT.M7) GOTO 480
0085        A(IADR)=VAL
0086        GOTO 10
0087    480 WRITE(6,3002)
0088        GOTO 10
      C------------------- SET INTEGER DATA ---
0089    500 DECODE(30,1003,CARD) IADR,IVAL
0090        IF(IADR.LE.0.OR.IADR.GT.M7) GOTO 480
0091        IA(IADR)=IVAL
0092        GOTO 10
      C------------------- MOMENTUM & ENERGY CHECK ---
0093    550 CALL CHECK
0094        GOTO 10
      C------------------- GRAVITY ---
0095    600 DECODE(30,1005,CARD) GRAVY,GRAVX
0096        GOTO 10
      C------------------- CONTACT STIFFNESSES ---
0097    650 DECODE(30,1005,CARD) STIFN,STIFS
0098        GOTO 10
      C------------------- RAYLEIGH DAMPING ---
0099    700 DECODE(50,1006,CARD) FRAC,FREQ,IF1,IF2
0100        PI2=8.0*ATAN(1.0)
0101        ALPHA=PI2*FRAC*FREQ
0102        BETA=FRAC/(PI2*FREQ)
0103        IF(IF1.EQ.0) GO TO 710
0104        ALPHA=0.0
0105        WRITE(6,3003)
0106    710 IF(IF2.EQ.0) GO TO 720
0107        BETA=0.0
0108        WRITE(6,3004)
0109    720 IF(MCYCLE.EQ.0) GO TO 10
0110        BDT=BETA/TDEL
0111        CON1=1.0-ALPHA*TDEL/2.0
0112        CON2=1.0/(1.0+ALPHA*TDEL/2.0)
0113        GOTO 10
      C------------------- FRICTION COEFFICIENT ---
0114    750 DECODE(20,1005,CARD) FRIC
0115        GOTO 10
      C------------------- ZERO ALL VELOCITIES ---
0116    800 DO 820 NB=1,NBLOKS
0117        I2=IA(NB)
0118        A(I2+5)=0.0
0119        A(I2+6)=0.0
0120        A(I2+7)=0.0
0121    820 CONTINUE
0122        GOTO 10
      C------------------- SET BLOCK LOADS ---
0123    850 DECODE(40,1006,CARD) FX,FY,NB
```

```
FORTRAN IV-PLUS V02-51          17:00:56    16-FEB-78          PAGE 9
RBM.LST          /I4/TR:BLOCKS/WR

0124            I2=IA(NB)
0125            A(I2+16)=FX
0126            A(I2+17)=FY
0127            GOTO 10
              C
0128         1000 FORMAT(20A4)
0129         1001 FORMAT(10X,I10,F10.0,I10)
0130         1002 FORMAT(8F10.0)
0131         1003 FORMAT(10X,2I10)
0132         1004 FORMAT(10X,I10,F10.0)
0133         1005 FORMAT(10X,2F10.0)
0134         1006 FORMAT(10X,2F10.0,2I10)
0135         2000 FORMAT(1X,4H+++ ,20A4,4H +++)
0136         2001 FORMAT(1X,4(1H(,E12.4,1H,,E12.4,3H ) ))
0137         2002 FORMAT(18H XC,YC,MASS,RMOI :,1P4E12.3)
0138         3000 FORMAT(28H !!! ERROR : ILLEGAL COMMAND)
0139         3001 FORMAT(34H !!! ERROR : COMMAND NOT AVAILABLE)
0140         3002 FORMAT(33H !!! ERROR : ADDRESS OUT OF RANGE)
0141         3003 FORMAT(10X,29HMASS DAMPING TERM SET TO ZERO)
0142         3004 FORMAT(10X,34HSTIFFNESS DAMPING TERM SET TO ZERO)
              C
0143            END
```

```
FORTRAN IV-PLUS V02-51          17:03:35     16-FEB-78          PAGE 11
RBM.LST          /I4/TR:BLOCKS/WR


0001            SUBROUTINE BOX
                C
                C------------------- ALL BLOCKS HAVE BEEN CREATED.
                C------------------- BLOCKS CAN NOW BE BOXED -----
                C
0002            COMMON A(3000)
0003            DIMENSION IA(1)
0004            EQUIVALENCE (A,IA)
0005            COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,M5,M6,M7,
                .               NBLOKS,NCYC,MCYCLE,NEMPT,RFLAG,UFLAG,EFLAG,
                .               TFRAC,TDEL,IBOXES,JBOXES,BSIZE,XSIZE,YSIZE,
                .               UDMAX,UMOST,STIFN,STIFS,FRIC,ALPHA,BETA,BDT,
                .               CON1,CON2,NVARB,NFRAG,NBR,NPR,
                .               GRAVX,GRAVY,LOC1,LOC2,NUPDAT,LBLOCK
0006            LOGICAL RFLAG,UFLAG,EFLAG
                C
0007            M4=M3+NBOXES
0008            M5=M4
                C------------------- INITIALISE BOX POINTERS ---
0009            I1=M4-1
0010            DO 10 I=M3,I1
0011          10 IA(I)=0
                C------------------- LOOP ON EACH BLOCK ---
0012            DO 50 NB=1,NBLOKS
0013            I2=IA(NB)
0014            NC=IA(I2+1)
0015            IC=I2+20
                C------------------- LOOP ON EACH CORNER ---
0016            DO 40 NP=1,NC
0017            X=A(I2+2)+A(IC)
0018            Y=A(I2+3)+A(IC+1)
0019            IBOX=MIN0(IFIX(X/BSIZE)+1,IBOXES)
0020            JBOX=MIN0(IFIX(Y/BSIZE),JBOXES-1)
0021            NBOX=JBOX*IBOXES+IBOX
0022            I3=M3+NBOX-1
0023            IA(M5+2)=IA(I3)
0024            IA(I3)=M5
0025            IA(M5)=NP
0026            IA(M5+1)=NB
0027            M5=M5+3
0028            IC=IC+3
0029          40 CONTINUE
                C
0030          50 CONTINUE
                C
0031            M6=M5+NBLOKM
0032            DO 60 I=M5,M6
0033          60 IA(I)=0
                C------------------- CREATE EMPTY LIST TO END OF MEMORY ---
0034            NEMPT=M6
0035            I6=M6+4
0036            DO 80 I=I6,M7,13
0037            J=I
0038            IA(I)=I+9
0039          80 CONTINUE
```

```
FORTRAN IV-PLUS V02-51          17:03:35     16-FEB-78          PAGE 12
RBM.LST          /I4/TR:BLOCKS/WR

0040            IA(J)=0
                C------------------- DETERMINE TIMESTEP ---
0041            TDEL=1.0E20
0042            DO 100 NB=1,NBLOKS
0043            I2=IA(NB)+11
0044            TN=2.0*SQRT(A(I2)/STIFN)
0045            TS=2.0*SQRT(A(I2)/STIFS)
0046            TDEL=AMIN1(TDEL,TN,TS)
0047          100 CONTINUE
0048            TDEL=TDEL*TFRAC
0049            WRITE(6,2000) TDEL
                C------------------- SET UP DAMPING TERMS ---
0050            BDT=BETA/TDEL
0051            CON1=1.0-ALPHA*TDEL/2.0
0052            CON2=1.0/(1.0+ALPHA*TDEL/2.0)
                C
0053            RETURN
                C
0054          2000 FORMAT(30X,17H TIME INCREMENT =,1PE12.4)
                C
0055            END
```

- Applies the boundary conditions in the form of velocities. These must be explicitly built into BOUNDY by means of FORTRAN statements.


```
FORTRAN IV-PLUS V02-51          17:04:30     16-FEB-78          PAGE 14
RBM.LST          /I4/TR:BLOCKS/WR


0001            SUBROUTINE CYCLE
                C
                C------------------- DRIVER FOR ITERATIONS ---
                C
0002            COMMON A(3000)
0003            DIMENSION IA(1)
0004            EQUIVALENCE (A,IA)
0005            COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,M5,M6,M7,
                .               NBLOKS,NCYC,MCYCLE,NEMPT,RFLAG,UFLAG,EFLAG,
                .               TFRAC,TDEL,IBOXES,JBOXES,BSIZE,XSIZE,YSIZE,
                .               UDMAX,UMOST,STIFN,STIFS,FRIC,ALPHA,BETA,BDT,
                .               CON1,CON2,NVARB,NFRAG,NBR,NPR,
                .               GRAVX,GRAVY,LOC1,LOC2,NUPDAT,LBLOCK
0006            LOGICAL RFLAG,UFLAG,EFLAG
                C
0007            DO 100 NCYCLE=1,NCYC
0008            MCYCLE=MCYCLE+1
                C------------------- UPDATE IF NECESSARY ---
0009            IF(UFLAG) CALL UPDAT
                C------------------- SCAN ALL BLOCKS ---
0010            UDMAX=0.0
0011            DO 20 NB=1,NBLOKS
0012            IB=IA(NB)
0013            CALL MOTION(A(IB))
0014            IF(RFLAG) CALL REBOX(NB)
0015          20 CONTINUE
                C------------------- EXIT IF NOTHING MOVED ---
0016            IF(UDMAX.EQ.0.0) GOTO 110
                C------------------- UPDATE CONTACTS ? ---
0017            UMOST=UMOST+UDMAX*TDEL
0018            IF(UMOST.LT.1.0) GOTO 30
                C------------------- YES ---
0019            UFLAG=.TRUE.
                C------------------- SCAN ALL CONTACTS ---
0020          30 DO 50 NB=1,NBLOKS
0021            IBE=IA(NB)
0022            J6=M5+NB-1
0023          40 I6=IA(J6)
0024            IF(I6.EQ.0) GOTO 50
0025            JBC=IA(I6+3)
0026            IBC=IA(JBC)
0027            CALL FORD(A(I6),A(IBC),A(IBE))
0028            J6=I6+4
0029            GOTO 40
0030          50 CONTINUE
                C
                C------------------- END CYCLE LOOP ---
                C
0031          100 CONTINUE
                C
0032          110 CONTINUE
                C
0033            RETURN
                C
0034            END
```

```
FORTRAN IV-PLUS V02-51          17:05:00     16-FEB-78          PAGE 16
RBM.LST          /14/TR:BLOCKS/WR


0001            SUBROUTINE MOTION(B)
                C
                C------------------- LAW OF MOTION FOR A SINGLE BLOCK ---
                C
0002            COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,M5,M6,M7,
                .               NBLOKS,NCYC,MCYCLE,NEMPT,RFLAG,UFLAG,EFLAG,
                .               TFRAC,TDEL,IBOXES,JBOXES,BSIZE,XSIZE,YSIZE,
                .               UDMAX,UMOST,STIFN,STIFS,FRIC,ALPHA,BETA,BDT,
                .               CON1,CON2,NVARB,NFRAG,NBR,NPR,
                .               GRAVX,GRAVY,LOC1,LOC2,NUPDAT,LBLOCK
0003            LOGICAL RFLAG,UFLAG,EFLAG
0004            DIMENSION B(1)
0005            EQUIVALENCE (FIX,JFIX)
                C
0006            FIX=B(1)
                C------------------- IS THIS BLOCK FIXED ? ---
0007            IF(JFIX.NE.0) RETURN
                C------------------- NO ---
                C------------------- VELOCITIES FROM ACCELERATIONS ---
0008            B(6)=(B(6)*CON1+(B(14)/B(12)+GRAVX)*TDEL)*CON2
0009            UDMAX=AMAX1(UDMAX,ABS(B(6)))
0010            B(7)=(B(7)*CON1+(B(15)/B(12)+GRAVY)*TDEL)*CON2
0011            UDMAX=AMAX1(UDMAX,ABS(B(7)))
0012            B(8)=(B(8)*CON1+(B(16)/B(13))*TDEL)*CON2
0013            B(14)=B(17)
0014            B(15)=B(18)
0015            B(16)=0.0
                C------------------- DISPLACEMENTS FROM VELOCITIES ---
0016            IBX=B(3)
0017            IBY=B(4)
0018            B(9)=B(6)*TDEL
0019            B(10)=B(7)*TDEL
0020            B(11)=B(8)*TDEL
0021            B(3)=B(3)+B(9)
0022            B(4)=B(4)+B(10)
0023            B(5)=B(5)+B(11)
                C------------------- DO WE NEED TO REBOX ? ---
0024            IF(IBX.EQ.IFIX(B(3)).AND.IBY.EQ.IFIX(B(4))
                .  .AND.ABS(B(5)).LT.0.01) GOTO 100
                C------------------- YES ---
0025            RFLAG=.TRUE.
0026            B(5)=0.0
                C------------------- UPDATE COS AND SIN FOR THIS BLOCK ---
0027          100 TEMP=B(19)
0028            B(19)=B(19)-B(11)*B(20)
0029            B(20)=B(20)+B(11)*TEMP
                C-------------------------------------------------------
0030            RETURN
                C
0031            END
```

```
FORTRAN IV-PLUS V02-51          17:05:28    16-FEB-78          PAGE 18
RBM.LST         /I4/TR:BLOCKS/WR


0001           SUBROUTINE FORD(C,BC,BE)
               C
               C------------------- FORCE DISPLACEMENT LAW FOR SINGLE CONTACT ---
               C
0002           COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,M5,M6,M7,
              .                NBLOKS,NCYC,MCYCLE,NEMPT,RFLAG,UFLAG,EFLAG,
              .                TFRAC,TDEL,IBOXES,JBOXES,BSIZE,XSIZE,YSIZE,
              .                UDMAX,UMOST,STIFN,STIFS,FRIC,ALPHA,BETA,BDT,
              .                CON1,CON2,NVARB,NFRAG,NBR,NPR,
              .                GRAVX,GRAVY,LOC1,LOC2,NUPDAT,LBLOCK
0003           LOGICAL RFLAG,UFLAG,EFLAG
0004           DIMENSION C(1),BC(1),BE(1)
               C
               C------------------- INCREMENTAL GLOBAL DISPLACEMENTS ---
0005           DUY=BC(10)-BE(10)+BC(11)*(C(12)-BC(3))-BE(11)*(C(12)-BE(3))
0006           DUX=BC( 9)-BE( 9)-BC(11)*(C(13)-BC(4))+BE(11)*(C(13)-BE(4))
               C------------------- NORMAL COORDINATES ---
0007           DUS=DUX*C(11)+DUY*C(10)
0008           DUN=DUY*C(11)-DUX*C(10)
               C------------------- NORMAL FORCE ---
0009           DFN=-DUN*STIFN
0010           C(8)=C(8)+DFN
               C------------------- TEST FOR TENSION ---
0011           IF(C(8).GE.0.0) GOTO 20
0012           C(8)=0.0
0013           C(9)=0.0
0014           DN=0.0
0015           DS=0.0
0016           GOTO 60
               C------------------- SHEAR FORCE ---
0017        20 DFS=DUS*STIFS
0018           C(9)=C(9)+DFS
0019           FRICF=FRIC*C(8)
0020           IF(ABS(C(9)).LE.FRICF) GOTO 40
0021           C(9)=SIGN(FRICF,C(9))
0022           DS=0.0
0023           GOTO 50
               C------------------- DASHPOT FORCES ---
0024        40 DS=BDT*DFS
0025        50 DN=BDT*DFN
               C------------------- GLOBAL CONTACT FORCES ---
0026        60 FYC=(C(9)+DS)*C(10)-(C(8)+DN)*C(11)
0027           FXC=(C(9)+DS)*C(11)+(C(8)+DN)*C(10)
               C------------------- ADD CONTRIBUTION TO BLOCK FORCES ---
0028           BC(14)=BC(14)-FXC
0029           BC(15)=BC(15)-FYC
0030           BC(16)=BC(16)-(FYC*(C(12)-BC(3))-FXC*(C(13)-BC(4)))
0031           BE(14)=BE(14)+FXC
0032           BE(15)=BE(15)+FYC
0033           BE(16)=BE(16)+(FYC*(C(12)-BE(3))-FXC*(C(13)-BE(4)))
               C
0034           RETURN
               C
0035           END
```

```
FORTRAN IV-PLUS V02-51          17:05:59     16-FEB-78          PAGE 20
RBM.LST         /I4/TR:BLOCKS/WR


0001           SUBROUTINE REBOX(NB)
               C
               C------------------- ROUTINE TO REBOX A SINGLE BLOCK ---
               C
0002           COMMON A(3000)
0003           DIMENSION IA(1)
0004           EQUIVALENCE (A,IA)
0005           COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,M5,M6,M7,
              .                NBLOKS,NCYC,MCYCLE,NEMPT,RFLAG,UFLAG,EFLAG,
              .                TFRAC,TDEL,IBOXES,JBOXES,BSIZE,XSIZE,YSIZE,
              .                UDMAX,UMOST,STIFN,STIFS,FRIC,ALPHA,BETA,BDT,
              .                CON1,CON2,NVARB,NFRAG,NBR,NPR,
              .                GRAVX,GRAVY,LOC1,LOC2,NUPDAT,LBLOCK
0006           LOGICAL RFLAG,UFLAG,EFLAG
               C
0007           IB=IA(NB)
0008           NC=IA(IB+1)
               C------------------- SCAN EACH CORNER OF THE BLOCK ---
0009           IC=IB+20
0010           DO 100 NP=1,NC
               C------------------- ARE WE AT EDGE OF DOMAIN ? ---
0011           IF(GLX(A(IB),IC).GT.0.0.AND.GLX(A(IB),IC).LT.XSIZE.AND.
              .   GLY(A(IB),IC).GT.0.0.AND.GLY(A(IB),IC).LT.YSIZE) GOTO 20
               C------------------- YES, SET MASTER FIX FLAG ---
0012           IA(IB)=2
0013           A(IB+5)=0.0
0014           A(IB+6)=0.0
0015           A(IB+7)=0.0
0016           A(IB+8)=0.0
0017           A(IB+9)=0.0
0018           A(IB+10)=0.0
0019           GOTO 900
               C------------------- NO, WHICH BOX SHOULD CORNER BE IN ? ---
0020        20 NBOX=IFIX(GLX(A(IB),IC)/BSIZE)+IFIX(GLY(A(IB),IC)/BSIZE)*IBOXES
               C------------------- SEARCH THIS BOX ---
0021           J4N=M3+NBOX-1
0022        30 I4N=IA(J4N)
0023           IF(I4N.EQ.0) GOTO 40
0024           IF(NB.EQ.IA(I4N+1).AND.NP.EQ.IA(I4N)) GOTO 100
0025           J4N=I4N+2
0026           GOTO 30
               C------------------- SEARCH THE SURROUNDING BOXES ---
0027        40 NXL=MAX0(MOD(NBOX-1,IBOXES),1)
0028           NXU=MIN0(NXL+2,IBOXES)
0029           NYL=MAX0(NBOX/IBOXES,1)
0030           NYU=MIN0(NYL+2,JBOXES)
0031           DO 80 JBOX=NYL,NYU
0032           MBOX=(JBOX-1)*IBOXES+NXL-1
0033           DO 70 IBOX=NXL,NXU
0034           MBOX=MBOX+1
0035           IF(MBOX.EQ.NBOX) GOTO 70
0036           J4M=M3+MBOX-1
0037        50 I4M=IA(J4M)
0038           IF(I4M.EQ.0) GOTO 70
0039           IF(NB.EQ.IA(I4M+1).AND.NP.EQ.IA(I4M)) GOTO 60
```

```
FORTRAN IV-PLUS V02-51          17:05:59     16-FEB-78          PAGE 21
RBM.LST          /I4/TR:BLOCKS/WR

0040            J4M=I4M+2
0041            GOTO 50
        C------------------- SWAP ENTRIES FOR THIS CORNER ---
0042          60 IA(J4M)=IA(I4M+2)
0043            IA(J4N)=I4M
0044            IA(I4M+2)=0
0045            GOTO 100
        C------------------- END OF SURROUNDING BOXES SCAN ---
0046          70 CONTINUE
0047          80 CONTINUE
0048            WRITE(6,3000) NP,NB
0049            EFLAG=.TRUE.
0050            CALL FINISH
        C------------------- END OF CORNERS SCAN ---
0051        100 IC=IC+3
        C
        C------------------- EXIT ---
0052        900 RFLAG=.FALSE.
0053            RETURN
        C
0054       3000 FORMAT(24H ERROR IN REBOX : CORNER,I3,
           .            9H OF BLOCK,I4,20H NOT IN ADJACENT BOX)
        C
0055            END
```

```
FORTRAN IV-PLUS V02-51          17:07:09     16-FEB-78          PAGE 23
RBM.LST          /I4/TR:BLOCKS/WR


0001           SUBROUTINE UPDAT
               C
               C------------------- UPDATE ALL CONTACTS ---
               C
0002           COMMON A(3000)
0003           DIMENSION IA(1)
0004           EQUIVALENCE (A,IA)
0005           COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,M5,M6,M7,
              .                NBLOKS,NCYC,MCYCLE,NEMPT,RFLAG,UFLAG,EFLAG,
              .                TFRAC,TOEL,IBOXES,JBOXES,BSIZE,XSIZE,YSIZE,
              .                UDMAX,UMOST,STIFN,STIFS,FRIC,ALPHA,BETA,BDT,
              .                CON1,CON2,NVARB,NFRAG,NBR,NPR,
              .                GRAVX,GRAVY,LOC1,LOC2,NUPDAT,LBLOCK
0006           LOGICAL RFLAG,UFLAG,EFLAG
               C
0007           LOGICAL FIRST
0008           DATA TOL,TOLB / 1.0 ,-0.01/
               C
               C------------------- SCAN EACH BLOCK ---
0009           DO 500 NB=1,NBLOKS
0010           I2=IA(NB)
0011           NC=IA(I2+1)
               C------------------- SCAN EACH EDGE OF BLOCK ---
0012           IEND2=I2+20
0013           DO 400 NP=1,NC
0014           FIRST=.TRUE.
0015           IEND1=IEND2
0016           IEND2=IEND1+3
0017           IF(NP.EQ.NC) IEND2=I2+20
               C------------------- DETERMINE BOXES TO BE SEARCHED ---
0018           X1=GLX(A(I2),IEND1)
0019           X2=GLX(A(I2),IEND2)
0020           Y1=GLY(A(I2),IEND1)
0021           Y2=GLY(A(I2),IEND2)
0022           TEMP=AMAX1(X1,X2)
0023           X1=AMIN1(X1,X2)
0024           X2=TEMP
0025           TEMP=AMAX1(Y1,Y2)
0026           Y1=AMIN1(Y1,Y2)
0027           Y2=TEMP
0028           NXL=IFIX((X1-TOL)/BSIZE)+1
0029           NXU=MINO(IFIX((X2+TOL)/BSIZE)+1,IBOXES)
0030           NYL=IFIX((Y1-TOL)/BSIZE)+1
0031           NYU=MINO(IFIX((Y2+TOL)/BSIZE)+1,JBOXES)
               C------------------- SEARCH EACH CANDIDATE BOX ---
0032           DO 350 JBOX=NYL,NYU
0033           NBOX=(JBOX-1)*IBOXES+NXL-1
0034           DO 300 IBOX=NXL,NXU
0035           NBOX=NBOX+1
0036           I4=IA(M3+NBOX-1)
               C------------------- END OF BOX LIST ? ---
0037         210 CONTINUE
0038           IF(I4.EQ.0) GOTO 300
               C------------------- NO, IS THIS BLOCK NB ? ---
0039           IF(IA(I4+1).NE.NB) GOTO 220
```

```
FORTRAN IV-PLUS V02-51          17:07:09     16-FEB-78          PAGE 24
RBM.LST          /14/TR:BLOCKS/WR

        C------------------- YES ---
        C------------------- GET NEXT ENTRY IN THIS BOX ---
0040      215 I4=IA(I4+2)
0041          GOTO 210
        C------------------- FIRST TIME THROUGH ? ---
0042      220 IF(.NOT.FIRST) GOTO 230
        C------------------- YES, COMPUTE COS AND SIN FOR THIS EDGE ---
0043          FIRST=.FALSE.
0044          COSA=(GLX(A(I2),IEND2)-GLX(A(I2),IEND1))/A(IEND1+2)
0045          SINA=(GLY(A(I2),IEND2)-GLY(A(I2),IEND1))/A(IEND1+2)
        C------------------- COMPUTE CORNER COORDINATES RELATIVE TO EDGE ---
0046      230 I2C=IA(I4+1)
0047          I2C=IA(I2C)
0048          IC=I2C+IA(I4)*3+17
0049          YT=(GLY(A(I2C),IC)-GLY(A(I2),IEND1))*COSA
             .  -(GLX(A(I2C),IC)-GLX(A(I2),IEND1))*SINA
0050          IF(YT.GT.1.0) GOTO 215
0051          IF(YT.LE.-3.0) GOTO 215
0052          XT=(GLX(A(I2C),IC)-GLX(A(I2),IEND1))*COSA
             .  +(GLY(A(I2C),IC)-GLY(A(I2),IEND1))*SINA
0053          IF(XT.GT.A(IEND1+2)) GOTO 215
0054          IF(XT.LT.0.0) GOTO 215
        C------------------- CONTACT LIST FOR BLOCK NB ---
0055          J6=M5+NB-1
0056          I6=IA(J6)
        C------------------- END OF LIST ? ---
0057      235 CONTINUE
0058          IF(I6.EQ.0) GOTO 250
        C------------------- NO. SAME CORNER AND EDGE ? ---
0059          IF(IA(I4).EQ.IA(I6+2).AND.IA(I4+1).EQ.IA(I6+3)
             .   .AND.NP.EQ.IA(I6+1)) GOTO 240
        C------------------- NO. GET NEXT ENTRY IN CONTACT LIST ---
0060          J6=I6+4
0061          I6=IA(J6)
0062          GOTO 235
        C------------------- CONTACT ALREADY STORED ---
0063      240 IF(YT.GT.-2.0) GOTO 245
        C------------------- APPLY DRIFT CORRECTION ---
0064          WRITE(6,3000)
        C------------------- UPDATE CONTACT DATA ---
0065      245 A(I6+9)=SINA
0066          A(I6+10)=COSA
0067          A(I6+11)=GLX(A(I2C),IC)
0068          A(I6+12)=GLY(A(I2C),IC)
0069          IA(I6)=1
0070          GOTO 215
        C------------------- NEW CONTACT ? ---
0071      250 CONTINUE
0072          IF(YT.GT.0.0) GOTO 215
        C------------------- YES ---
0073          I6=NEMPT
0074          ICR=IC-3
0075          IF(IA(I4).EQ.1) ICR=IC+3*(IA(I2C+1)-1)
0076          YTR=(GLY(A(I2C),ICR)-GLY(A(I2),IEND1))*COSA
             .   -(GLX(A(I2C),ICR)-GLX(A(I2),IEND1))*SINA
0077          IF(YTR.LE.-2.0) GOTO 215
```

```
FORTRAN IV-PLUS V02-51          17:07:09     16-FEB-78          PAGE 25
RBM.LST          /I4/TR:BLOCKS/WR

0078            ICL=IC+3
0079            IF(IA(I4).EQ.IA(I2C+1)) ICL=I2C+20
0080            YTL=(GLY(A(I2C),ICL)-GLY(A(I2),IEND1))*COSA
                .   -(GLX(A(I2C),ICL)-GLX(A(I2),IEND1))*SINA
0081            IF(YTL.LE.-2.0) GOTO 215
              C------------------- ANY SPACE AVAILABLE IN EMPTY LIST ? ---
0082            IF(IA(NEMPT+4).NE.0) GOTO 260
0083            WRITE(6,3001)
0084            CALL FINISH
              C------------------- NEW CONTACT ---
0085          260 A(I6+5)=0.0
0086            A(I6+6)=0.0
0087            A(I6+7)=0.0
0088            A(I6+8)=0.0
0089            IA(I6+1)=NP
0090            IA(I6+2)=IA(I4)
0091            IA(I6+3)=IA(I4+1)
0092            NEMPT=IA(I6+4)
0093            IA(J6)=I6
0094            IA(I6+4)=0
0095            GOTO 245
              C------------------- END OF BOX SCAN ---
0096          300 CONTINUE
0097          350 CONTINUE
              C------------------- END OF EDGE SCAN ---
0098          400 CONTINUE
              C------------------- SCAN CONTACT LIST FOR PRESERVE FLAGS ---
0099            J6=M5+NB-1
0100            I6=IA(J6)
              C------------------- END OF LIST ? ---
0101          410 IF(I6.EQ.0) GOTO 490
              C------------------- NO ---
              C------------------- IS THE PRESERVE FLAG SET ? ---
0102            IF(IA(I6).EQ.0) GOTO 420
              C------------------- YES ---
0103            IA(I6)=0
0104            J6=I6+4
0105            GOTO 430
              C------------------- NO ---
0106          420 IA(J6)=IA(I6+4)
0107            IA(I6+4)=NEMPT
0108            NEMPT=I6
              C------------------- GET NEXT CONTACT ---
0109          430 I6=IA(J6)
0110            GOTO 410
              C------------------- END OF BLOCK SCAN ---
0111          490 CONTINUE
0112          500 CONTINUE
              C
0113            NUPDAT=NUPDAT+1
0114            UFLAG=.FALSE.
0115            UMOST=0.0
0116            RETURN
              C
0117         3000 FORMAT(30X,26H DRIFT CORRECTION REQUIRED)
0118         3001 FORMAT(30X,32H NO MORE MEMORY FOR CONTACT LIST)
```

```
FORTRAN IV-PLUS V02-51          17:07:09     16-FEB-78          PAGE 26
RBM.LST          /I4/TR:BLOCKS/WR

        C
0119            END
```

```
FORTRAN IV-PLUS V02-51          17:08:30     16-FEB-78          PAGE 28
RBM.LST          /14/TR:BLOCKS/WR


0001            FUNCTION GLX(B,IC)
                C
                C------------------- LOCAL-GLOBAL X-COORDINATE TRANSFORMATION ---
                C
                      COMMON A(3000)
0002                  DIMENSION IA(1)
0003                  EQUIVALENCE (A,IA)
0004                  DIMENSION B(1)
0005            C
0006                  GLX=B(3)+A(IC)*B(19)-A(IC+1)*B(20)
0007            C
                      RETURN
0008            C
                      END
```

```
FORTRAN IV-PLUS V02-51          17:08:33     16-FEB-78          PAGE 30
RBM.LST          /14/TR:BLOCKS/WR


0001            FUNCTION GLY(B,IC)
                C
                C------------------- LOCAL-GLOBAL Y-COORDINATE TRANSFORMATION ---
                C
0002            COMMON A(3000)
0003            DIMENSION IA(1)
0004            EQUIVALENCE (A,IA)
0005            DIMENSION B(1)
                C
0006            GLY=B(4)+A(IC)*B(20)+A(IC+1)*B(19)
                C
0007            RETURN
                C
0008            END
```

```
FORTRAN IV-PLUS V02-51          17:08:35     16-FEB-78          PAGE 32
RBM.LST         /14/TR:BLOCKS/WR


0001           SUBROUTINE DUMP
               C
               C------------------- ROUTINE TO PRINT MEMORY ALLOCATION AND CONTENTS
               C
0002           COMMON A(3000)
0003           DIMENSION IA(1)
0004           EQUIVALENCE (A,IA)
0005           COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,M5,M6,M7,
              .                NBLOKS,NCYC,MCYCLE,NEMPT,RFLAG,UFLAG,EFLAG,
              .                TFRAC,TDEL,IBOXES,JBOXES,BSIZE,XSIZE,YSIZE,
              .                UDMAX,UMOST,STIFN,STIFS,FRIC,ALPHA,BETA,BDT,
              .                CON1,CON2,NVARB,NFRAG,NBR,NPR,
              .                GRAVX,GRAVY,LOC1,LOC2,NUPDAT,LBLOCK
0006           LOGICAL RFLAG,UFLAG,EFLAG
               C
0007           IF(LOC2.EQ.0) RETURN
0008           WRITE(6,2000)
0009           WRITE(6,2005) M1,M2,M3,M4,M5,M6,M7,NBLOKS,NCYC,NEMPT,MCYCLE
0010           IF(LOC2.LT.0) GOTO 100
               C------------------- DUMP CONSECUTIVE MEMORY LOCATIONS ---
0011           WRITE(6,2009)
0012           I2=((LOC1-1)/10)*10
0013        10 I1=I2+1
0014           I2=I1+9
0015           WRITE(6,2001) (IA(I),I=I1,I2),I2
0016           IF(I2.LT.LOC2) GOTO 10
0017           GOTO 500
               C
               C------------------- DUMP BY BOX ---
0018       100 WRITE(6,2002)
0019           DO 120 NBOX=1,NBOXES
0020           I3=IA(M3+NBOX-1)
               C------------------- ANY MORE ENTRIES ? ---
0021       110 IF(I3.EQ.0) GOTO 120
               C------------------- NO, PRINT BLOCK,CORNER ---
0022           WRITE(6,2008) NBOX,IA(I3+1),IA(I3)
0023           I3=IA(I3+2)
0024           GOTO 110
0025       120 CONTINUE
               C------------------- CONTENTS OF EACH BLOCK ---
0026           WRITE(6,2006)
0027           DO 130 NB=1,NBLOKS
0028           IF=IA(NB)
0029           IL=IF+19+3*IA(IF+1)
0030           WRITE(6,2003) NB,IA(IF),IA(IF+1),(A(I),I=IF+2,IL)
0031       130 CONTINUE
               C------------------- DUMP CONTACT DATA ---
0032           IF(M5.EQ.0) GOTO 500
0033           WRITE(6,2007)
0034           DO 150 NB=1,NBLOKS
0035           J6=M5+NB-1
0036       140 I6=IA(J6)
0037           IF(I6.EQ.0) GOTO 150
0038           WRITE(6,2004) NB,(IA(I),I=I6,I6+4),
                             (A(I),I=I6+5,I6+12)
```

```
FORTRAN IV-PLUS V02-51          17:08:35     16-FEB-78          PAGE 33
RBM.LST          /I4/TR:BLOCKS/WR

0039            J6=I6+4
0040            GOTO 140
0041          150 CONTINUE
              C------------------
0042          500 WRITE(6,2000)
              C
0043            RETURN
              C
0044         2000 FORMAT(1X,130(1H-))
0045         2001 FORMAT(1X,10012,I6)
0046         2002 FORMAT(10X,3HBOX,7X,5HBLOCK,6X,6HCORNER)
0047         2003 FORMAT(/1X,3I3,1P11E10.2/(10X,1P11E10.2))
0048         2004 FORMAT(/1X,6I10/1X,1P8E10.2)
0049         2005 FORMAT(9X,2HM1,8X,2HM2,8X,2HM3,8X,2HM4,8X,2HM5,8X,2HM6,
                    .       8X,2HM7,4X,6HNBLOKS,6X,4HNCYC,5X,5HNEMPT,4X,6HMCYCLE/
                    .       1X,11I10)
0050         2006 FORMAT(11H BLOCK DATA,10(1H-)/1X,9H NB IF NC,8X,2HXC,8X,2HYC,
                    .       5X,5HTHETA,6X,4HXDOT,6X,4HYDOT,6X,4HTDOT,
                    .       8X,2HDX,8X,2HDY,4X,6HDTHETA,6X,4HAREA,3X,7HINERTIA/
                    .       15X,5HXFSUM,5X,5HYFSUM,6X,4HMSUM,5X,5HXLOAD,5X,5HYLOAD,
                    .       7X,3HCOS,7X,3HSIN)
0051         2007 FORMAT(13H CONTACT DATA,10(1H-)/8X,3HNBE,7X,3HPRE,7X,3HNPE,7X,
                    .       3HNPC,7X,3HNBC,6X,4HLINK/10X,1HS,9X,1HN,8X,2HFN,8X,2HFS,
                    .       7X,3HSIN,7X,3HCOS,7X,3HXCP,7X,3HYCP)
0052         2008 FORMAT(1X,3I12)
0053         2009 FORMAT(1X,30(1H*),19HVALUES ARE IN OCTAL,30(1H*))
              C
0054            END
```

```
FORTRAN IV-PLUS V02-51          17:08:52     16-FEB-78          PAGE 35
RBM.LST          /14/TR:BLOCKS/WR


0001            SUBROUTINE BPLOT
                C
                C------------------- PLOT A SNAPSHOT OF THE GEOMETRY ---
                C
0002            COMMON A(3000)
0003            DIMENSION IA(1)
0004            EQUIVALENCE (A,IA)
0005            COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,M5,M6,M7,
                .                NBLOKS,NCYC,MCYCLE,NEMPT,RFLAG,UFLAG,EFLAG,
                .                TFRAC,TDEL,IBOXES,JBOXES,BSIZE,XSIZE,YSIZE,
                .                UDMAX,UMOST,STIFN,STIFS,FRIC,ALPHA,BETA,BDT,
                .                CON1,CON2,NVARB,NFRAG,NBR,NPR,
                .                GRAVX,GRAVY,LOC1,LOC2,NUPDAT,LBLOCK
0006            LOGICAL RFLAG,UFLAG,EFLAG
                C
0007            SCREEN=10.0
0008            FACT=SCREEN/AMAX1(XSIZE,YSIZE)
                C------------------- PLOT EACH BLOCK ---
0009            DO 100 NB=1,NBLOKS
0010            I2=IA(NB)
0011            NC=IA(I2+1)
0012            XC=FACT*A(I2+2)
0013            YC=FACT*A(I2+3)
                C------------------- FIXED ? ---
0014            IF(IA(I2).EQ.0) GOTO 20
                C------------------- YES, ALREADY PLOTTED ? ---
0015            IF(IA(I2).GT.2) GOTO 100
                C------------------- NO ---
0016            IA(I2)=IA(I2)+2
0017            CALL SYMBOL(XC,YC,0.2,1HF,0.0,1)
0018          20 IC=I2+20
0019            X=FACT*GLX(A(I2),IC)
0020            Y=FACT*GLY(A(I2),IC)
0021            CALL PLOT(X,Y,3)
0022            DO 50 NP=2,NC
0023            IC=IC+3
0024            X=FACT*GLX(A(I2),IC)
0025            Y=FACT*GLY(A(I2),IC)
0026            CALL PLOT(X,Y,2)
0027          50 CONTINUE
0028            X=FACT*GLX(A(I2),I2+20)
0029            Y=FACT*GLY(A(I2),I2+20)
0030            CALL PLOT(X,Y,2)
                C------------------- END OF BLOCK LOOP ---
0031        100 CONTINUE
                C
0032            CALL PLOT(X,Y,3)
0033            CALL PLOT(0.0,0.0,3)
                C
0034            RETURN
                C
0035            END
```

```
FORTRAN IV-PLUS V02-51          17:09:04     16-FEB-78          PAGE 37
RBM.LST          /I4/TR:BLOCKS/WR


0001            SUBROUTINE FINISH
                C
                C------------------- TIDY UP AND STOP ---
                C
0002            COMMON A(3000)
0003            DIMENSION IA(1)
0004            EQUIVALENCE (A,IA)
0005            COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,M5,M6,M7,
                .               NBLOKS,NCYC,MCYCLE,NEMPT,RFLAG,UFLAG,EFLAG,
                .               TFRAC,TDEL,IBOXES,JBOXES,BSIZE,XSIZE,YSIZE,
                .               UDMAX,UMOST,STIFN,STIFS,FRIC,ALPHA,BETA,BDT,
                .               CON1,CON2,NVARB,NFRAG,NBR,NPR,
                .               GRAVX,GRAVY,LOC1,LOC2,NUPDAT,LBLOCK
0006            LOGICAL RFLAG,UFLAG,EFLAG
                C
0007            CALL PLOTND
0008            WRITE(6,2000) MCYCLE,NUPDAT
                C
                C------------------- WRITE RESTART FILE IF NO ERRORS ---
0009            IF(EFLAG) GOTO 100
0010            REWIND 1
0011            WRITE(1) (HED(I),I=1,LBLOCK)
0012            WRITE(1) (A(I),I=1,M7)
0013            WRITE(6,2001)
0014          100 STOP
                C
0015           2000 FORMAT(30X,13H TOTAL CYCLES,I10/
                .          30X,13H NO. UPDATES ,I10)
0016           2001 FORMAT(30X,32H A RESTART FILE HAS BEEN WRITTEN)
                C
0017            END
```

```
FORTRAN IV-PLUS V02-51          17:09:09     16-FEB-76          PAGE 39
RBM.LST          /14/TR:BLOCKS/WR


0001            SUBROUTINE CHECK
                C
                C------------------- MOMENTUM AND ENERGY CHECK ---
                C
0002            COMMON A(3000)
0003            DIMENSION IA(1)
0004            EQUIVALENCE (A,IA)
0005            COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,M5,M6,M7,
                .                NBLOKS,NCYC,MCYCLE,NEMPT,RFLAG,UFLAG,EFLAG,
                .                TFRAC,TDEL,IBOXES,JBOXES,BSIZE,XSIZE,YSIZE,
                .                UDMAX,UMOST,STIFN,STIFS,FRIC,ALPHA,BETA,BDT,
                .                CON1,CON2,NVARB,NFRAG,NBR,NPR,
                .                GRAVX,GRAVY,LOC1,LOC2,NUPDAT,LBLOCK
0006            LOGICAL RFLAG,UFLAG,EFLAG
                C
0007            XMOM=0.0
0008            YMOM=0.0
0009            TMOM=0.0
0010            TARM=0.0
0011            XKE=0.0
0012            YKE=0.0
0013            TKE=0.0
0014            DO 100 NB=1,NBLOKS
0015            I2=IA(NB)
0016            XMOM=XMOM+A(I2+11)*A(I2+5)
0017            YMOM=YMOM+A(I2+11)*A(I2+6)
0018            TMOM=TMOM+A(I2+12)*A(I2+7)
0019            TARM=TARM+A(I2+11)*(A(I2+2)*A(I2+6)-A(I2+3)*A(I2+5))
0020            XKE=XKE+0.5*A(I2+11)*A(I2+5)**2
0021            YKE=YKE+0.5*A(I2+11)*A(I2+6)**2
0022            TKE=TKE+0.5*A(I2+12)*A(I2+7)**2
0023            WRITE(6,2001) XMOM,YMOM,TMOM,TARM,XKE,YKE,TKE
0024          100 CONTINUE
                C
0025            ESUM=XKE+YKE+TKE
0026            TSUM=TMOM+TARM
                C
0027            WRITE(6,2000) XMOM,YMOM,TMOM,TARM,TSUM,XKE,YKE,TKE,ESUM
                C
0028            RETURN
                C
0029           2000 FORMAT(9H MOMENTUM,11X,1HX,11X,1HY,2X,10HROTATIONAL/
                .       9X,1P5E12.4//
                .       9H ENERGY  ,11X,1HX,11X,1HY,2X,10HROTATIONAL,7X,5HTOTAL/
                .       9X,1P4E12.4)
0030           2001 FORMAT(1X,1P7E12.4)
                C
0031            END


PROGRAM SECTIONS

 NAME        SIZE                  ATTRIBUTES

$CODE1   000712   229               RW,I,CON,LCL
```

APPENDIX XIII: LISTING OF PROGRAM SDEM

```
FORTRAN IV-PLUS V02-04G          17:35:23     22-MAR-78           PAGE 1
RBM.FTN          /I4/TR:BLOCKS/WR

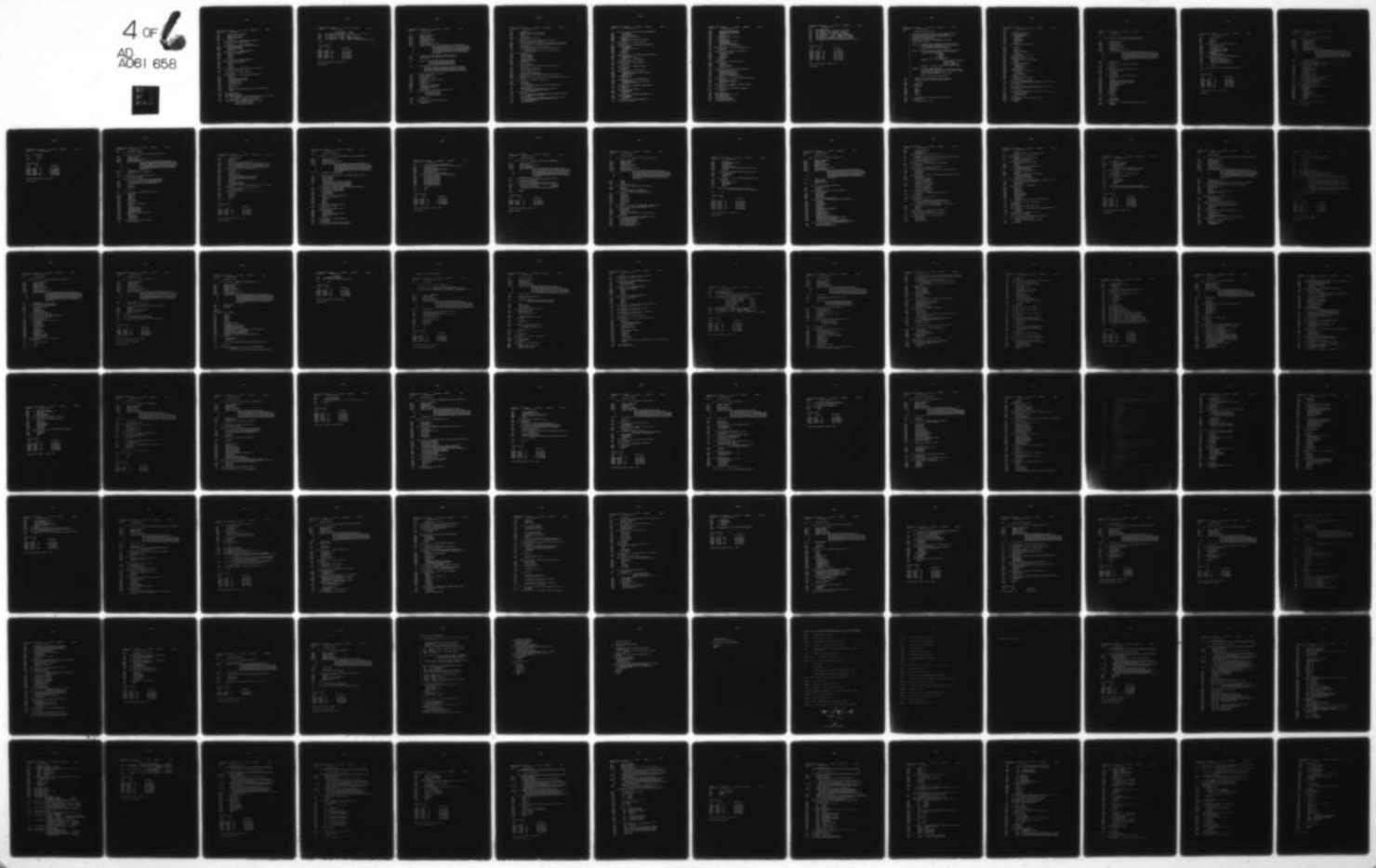
0001            PROGRAM SDEM
        C
        C-------------------------------------------------------------
        C-------------------- RIGID BLOCK MODEL. ---------------------
        C-------------------- FREELY TRANSLATED INTO FORTRAN BY -------
        C-------------------- P.J.BERESFORD, FROM AN ORIGINAL PROGRAM -
        C-------------------- BY P.A.CUNDALL. ------------------------
        C-------------------- DAMES AND MOORE, LONDON. ---------------
        C-------------------- VERSION 2.0, NOVEMBER 1977. ------------
        C-------------------- MODIFIED BY P.A. CUNDALL TO ------------
        C-------------------- INCLUDE LIMITED DEFORMABILITY ----------
        C-------------------------------------------------------------
        C
0002            INCLUDE 'COMMON.FTN'
0003 *          COMMON A(5000)
0004 *          DIMENSION IA(1)
0005 *          EQUIVALENCE (A,IA)
0006            INCLUDE 'CBLOCK.FTN'
0007 *          LOGICAL LOCK
0008 *          LOGICAL RFLAG,DFLAG,EFLAG
0009 *          REAL LAME1,LAME2
0010 *          COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,M5,M6,M7,
     *         .                  NBLOKS,NCYC,MCYCLE,NEMPT,RFLAG,DFLAG,EFLAG,TFRAC,
     *         .                  TDEL,IBOXES,JBOXES,XL,XO,YL,YO,BSIZE,SFACT,XSIZE,
     *         .                  YSIZE,UDMAX,UMOST,STIFN,STIFS,FRIC,BETA,BDT,ALPHB,
     *         .                  CON1,CON2,ALPHA,NVARB,NBR,IBR,NPR,LAME1,LAME2,
     *         .                  G,CON1B,CON2B,GRAVX,GRAVY,LOC1,LOC2,NUPDAT,YIELD,
     *         .                  LOCK,LBLOCK
        C
0011            DATA STIFN,STIFS,FRIC,BETA/2*1.0E8,0.0,0.0/
0012            DATA YIELD /1.0E10/
0013            DATA GRAVY,GRAVX/-9.81,0.0/
0014            DATA M7,MCYCLE,NUPDAT,LBLOCK/3000,0,0,75/
0015            DATA RFLAG,DFLAG,EFLAG/.FALSE.,.TRUE.,.FALSE./
0016            DATA LOCK /.FALSE./
0017            DATA NVARB /24/
        C
0018            CALL PLOTST(.025,'CM')
0019            CALL SETUP
0020         10 CALL NEXT
0021            CALL CYCLE
0022            GOTO 10
        C
0023            END
```

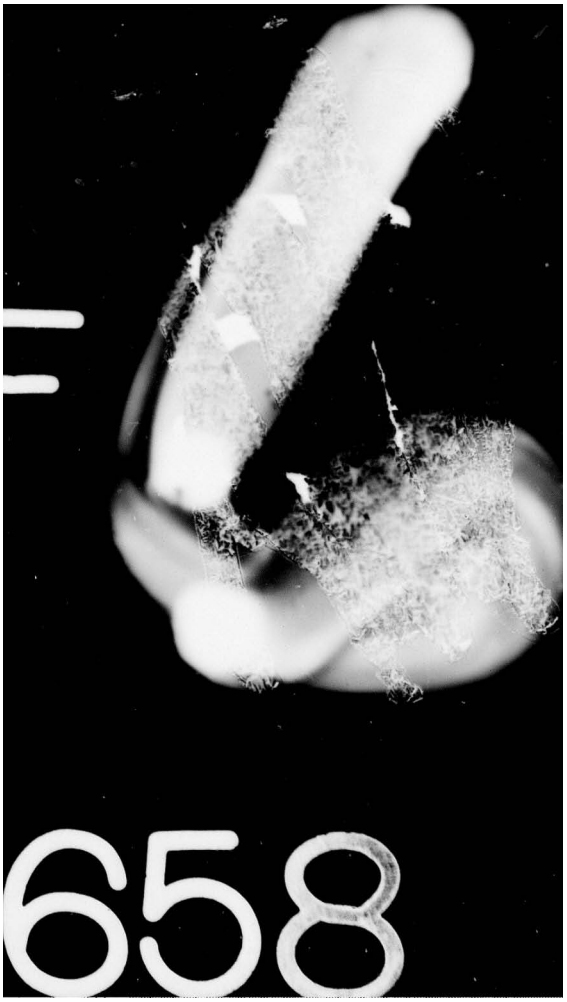
```
FORTRAN IV-PLUS V02-04G          17:35:38     22-MAR-78            PAGE 1
SETUP.FTN        /I4/TR:BLOCKS/WR

0001            SUBROUTINE SETUP
      C
0002            INCLUDE 'COMMON.FTN'
0003 *          COMMON A(5000)
0004 *          DIMENSION IA(1)
0005 *          EQUIVALENCE (A,IA)
0006            INCLUDE 'CBLOCK.FTN'
0007 *          LOGICAL LOCK
0008 *          LOGICAL RFLAG,UFLAG,EFLAG
0009 *          REAL LAME1,LAME2
0010 *          COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,M5,M6,M7,
     *         .                NBLOKS,NCYC,MCYCLE,NEMPT,KFLAG,UFLAG,EFLAG,TFRAC,
     *         .                TDEL,IBOXES,JBOXES,XL,XU,YL,YU,BSIZE,SFACT,XSIZE,
     *         .                YSIZE,UDMAX,UMOST,STIFN,STIFS,FRIC,BETA,BDT,ALPHB,
     *         .                CON1,CON2,ALPHA,NVARB,NBR,IBR,NPR,LAME1,LAME2,
     *         .                G,CON1B,CON2B,GRAVX,GRAVY,LOC1,LOC2,NUPDAT,YIELD,
     *         .                LOCK,LBLOCK
      C
0011            DIMENSION CARD(20),WORD(10)
0012            BYTE DAY(10),TIM(10)
      C
0013            DATA WORD /4HSTAR,4HREST,4HBLOC,4HBOXE,4HXLIM,
               .           4HYLIM,4H****,4HFRAC,4HZZZZ,4HZZZZ/
      C
0014            JOB=0
0015            CALL DATE(DAY)
0016            CALL TIME(TIM)
0017            WRITE(6,2000) DAY,TIM
      C---------------------- READ NEXT CARD ---
0018         10 READ(5,1000) CARD
0019            CALL TIME(TIM)
0020            WRITE(6,2001) CARD,TIM
      C
0021            DO 20 I=1,10
0022            IF(CARD(1).EQ.WORD(I)) GOTO 30
0023         20 CONTINUE
      C
0024            WRITE(6,3000)
0025            GOTO 10
      C
0026         30 IF(I.GT.2.AND.JOB.EQ.0) GOTO 60
0027            GOTO (110,120,130,140,150,
               .      160,900,170, 10, 10), I
      C
0028         50 WRITE(6,3001)
0029            GOTO 10
      C
0030         60 WRITE(6,3002)
0031            GOTO 10
      C---------------------- START OF NEW RUN ---
0032        110 JOB=1
0033            DO 111 I=1,20
0034        111 HED(I)=CARD(I)
0035            GOTO 10
      C---------------------- RESTART RUN ---
0036        120 JOB=2
```

AD-A061 658 DAMES AND MOORE LOS ANGELES CA F/G 8/7
COMPUTER MODELING OF JOINTED ROCK MASSES.(U)
AUG 78 T MAINI, P CUNDALL, J MARTI DACA39-77-C-0004
UNCLASSIFIED WES-TR-N-78-4 NL

4 OF 6
AD A061 658




658

```
FORTRAN IV-PLUS V02-04G          17:35:38     22-MAR-78           PAGE 2
SETUP.FTN       /I4/TR:BLOCKS/WR

0037           READ(1) (HED(I),I=1,LBLOCK)
0038           READ(1) (A(I),I=1,M7)
0039           WRITE(6,2003) NCYCLE
0040           WRITE(6,2004) HED
0041           RETURN
               C------------------- MAXIMUM NUMBER OF BLOCKS ---
0042         130 DECODE(20,1001,CARD) NBLOKM
0043           GOTO 10
               C------------------- NUMBER OF BOXES ---
0044         140 DECODE(40,1001,CARD) IBOXES,JBOXES,IBSIZE
0045           BSIZE=FLOAT(IBSIZE)
0046           NBOXES=IBOXES*JBOXES
0047           GOTO 10
               C------------------- PROBLEM X-LIMITS ---
0048         150 DECODE(30,1002,CARD) XL,XU
0049           IF(XL.LT.XU) GOTO 10
0050           TEMP=XL
0051           XL=XU
0052           XU=TEMP
0053           GOTO 10
               C------------------- PROBLEM Y-LIMITS ---
0054         160 DECODE(30,1002,CARD) YL,YU
0055           IF(YL.LT.YU) GOTO 10
0056           TEMP=YL
0057           YL=YU
0058           YU=TEMP
0059           GOTO 10
               C------------------- FRACTION OF CRITICAL TIMESTEP ---
0060         170 DECODE(20,1002,CARD) TFRAC
0061           GOTO 10
               C------------------- SETUP FINISHED ---
0062         900 CONTINUE
               C------------------- INITIALISE SOME VARIABLES ---
0063           UMOST=0.0
0064           NBLOKS=0
0065           XSIZE=FLOAT(IBOXES)*BSIZE
0066           YSIZE=FLOAT(JBOXES)*BSIZE
0067           XF=XSIZE/(XU-XL)
0068           YF=YSIZE/(YU-YL)
0069           SFACT=AMIN1(XF,YF)
0070           M1=1
0071           M2=M1+NBLOKM
0072           M3=M2
0073           WRITE(6,2002) NBLOKM,NBOXES,XL,XU,YL,YU,BSIZE,SFACT
0074           RETURN
               C
0075          1000 FORMAT(20A4)
0076          1001 FORMAT(10X,3I10)
0077          1002 FORMAT(10X,5F10.0)
0078          2000 FORMAT(30X28HSDEM - SIMPLY-DEFORMABLE DEM,53X,10A1,1X,10A1/
                  .       30X28H----------------------------/)
0079          2001 FORMAT(1X,4H+++ ,20A4,4H +++,32X,10A1)
0080          2002 FORMAT(/30X,25H MAXIMUM NUMBER OF BLOCKS,I5/
                  .       30X,25H NUMBER OF BOXES         ,I5/
                  .       30X,10H X-LIMITS ,2F10.2/
                  .       30X,10H Y-LIMITS ,2F10.2/
```

```
FORTRAN IV-PLUS V02-04G          17:35:38     22-MAR-78            PAGE 3
SETUP.FTN        /I4/TR:BLOCKS/WR

              .          30X,10H BSIZE     ,F10.2/
              .          30X,10H SFACT     ,F10.2)
0081      2003 FORMAT(30X,31H RESTART RUN. CURRENT CYCLES ..,I10)
0082      2004 FORMAT(30X,9HHEADING: ,20A4)
0083      3000 FORMAT(28H !!! ERROR : ILLEGAL COMMAND)
0084      3001 FORMAT(34H !!! ERROR : COMMAND NOT AVAILABLE)
0085      3002 FORMAT(48H !!! ERROR : 'START' OR 'RESTART' CARD NOT FOUND)
         C
0086            END


PROGRAM SECTIONS

 NAME        SIZE                  ATTRIBUTES

$CODE1   002220    584              RW,I,CON,LCL
$PDATA   000026     11              RW,D,CON,LCL
$IDATA   000740    240              RW,D,CON,LCL
$VARS    000244     82              RW,D,CON,LCL
$TEMPS   000004      2              RW,D,CON,LCL
.$$$$.   047040  10000              RW,D,OVR,GBL
CBLOCK   000460    152              RW,D,OVR,GBL


TOTAL SPACE ALLOCATED = 053176 11071

SETUP,O=SETUP
```

```
FORTRAN IV-PLUS V02-04G          17:36:23     22-MAR-78            PAGE 1
NEXT.FTN         /14/TR:BLOCKS/WR

0001           SUBROUTINE NEXT
       C
0002           INCLUDE 'COMMON.FTN'
0003 *         COMMON A(5000)
0004 *         DIMENSION IA(1)
0005 *         EQUIVALENCE (A,IA)
0006           INCLUDE 'CBLOCK.FTN'
0007 *         LOGICAL LOCK
0008 *         LOGICAL RFLAG,UFLAG,EFLAG
0009 *         REAL LAME1,LAME2
0010 *         COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,M5,M6,M7,
      *        .                NBLOKS,NCYC,MCYCLE,NEMPT,RFLAG,UFLAG,EFLAG,TFRAC,
      *        .                TDEL,IBOXES,JBOXES,XL,XU,YL,YU,BSIZE,SFACT,XSIZE,
      *        .                YSIZE,UDMAX,UMOST,STIFN,STIFS,FRIC,BETA,BDT,ALPHB,
      *        .                CON1,CON2,ALPHA,NVARB,NBR,IBR,NPR,LAME1,LAME2,
      *        .                G,CON1B,CON2B,GRAVX,GRAVY,LOC1,LOC2,NUPDAT,YIELD,
      *        .                LOCK,LBLOCK
       C
0011           COMMON /PAR/ LINE(80),RVAR(10),IVAR(10),AVAR(10)
0012           DIMENSION CARD(20),WORD(20),X(50),Y(50),ICODE(20)
0013           BYTE TIM(10)
       C
0014           DATA WORD /4HCREA,4HDELE,4HDUMP,4HCYCL,4HSTOP,
              .           4HPLOT,4H****,4HRSET,4HISET,4HCHEC,
              .           4HGRAV,4HSTIF,4HDAMP,4HFRIC,4HZERO,
              .           4HLOAD,4HBDAM,4HELAS,4HLOCK,4HPLAS/
0015           DATA ICODE /
              .     0000000121,0000000000,0000000011,0000000001,0000000000,
              .     0000000000,0000000000,0000000021,0000000011,0000000000,
              .     0000000022,0000000022,0000001122,0000000002,0000000000,
              .     0000000122,0000000022,0000000022,0000000003,0000000022/
       C
       C------------------- READ NEXT CARD ---
0016       10 READ(5,1000) CARD
0017          CALL TIME(TIM)
0018          WRITE(6,2000) CARD,TIM
0019          DO 20 I=1,20
0020          IF(CARD(1).EQ.WORD(I)) GOTO 30
0021       20 CONTINUE
0022          WRITE(6,3000)
0023          GOTO 10
0024       30 DECODE(80,1007,CARD) LINE
0025          CALL PARSE(ICODE(I),IERR)
0026          IF(IERR.EQ.0) GOTO 35
0027          CALL FINISH
       C------------------- JUMP TO APPROPRIATE CODE ---
0028       35 GOTO (100,150,200,250,300,
              .       350,400,450,500,550,
              .       600,650,700,750,800,
              .       850,900,950,970,980), I
       C
0029       40 WRITE(6,3001)
0030          GOTO 10
       C------------------- CREATE A NEW BLOCK ---
0031      100 NBLOKS=NBLOKS+1
0032          NC=IVAR(1)
```

```
FORTRAN IV-PLUS V02-046          17:36:23     22-MAR-78           PAGE 2
NEXT.FTN         /14/TR:BLOCKS/WR

0033            RHO=RVAR(2)
0034            READ(5,1002) (X(I),Y(I),I=1,NC)
0035            WRITE(6,2001) (X(I),Y(I),I=1,NC)
0036            I2=M3
0037            IA(NBLOKS)=I2
0038            IF(IVAR(3).NE.0) IA(I2)=1
0039            IA(I2+1)=NC
          C------------------- AREA AND CENTROID OF THIS BLOCK ---
0040            AREA=(X(1)-X(NC))*(Y(1)+Y(NC))
0041            YC=(X(1)-X(NC))*((Y(1)-Y(NC))*(Y(1)+2.0*Y(NC))+3.0*Y(NC)**2)
0042            XC=(Y(1)-Y(NC))*((X(1)-X(NC))*(X(1)+2.0*X(NC))+3.0*X(NC)**2)
0043            DO 110 I=2,NC
0044            AREA=AREA+(X(I)-X(I-1))*(Y(1)+Y(I-1))
0045            YC=YC+(X(I)-X(I-1))*((Y(1)-Y(I-1))*(Y(1)+2.0*Y(I-1))
               .      +3.0*Y(I-1)**2)
0046            XC=XC+(Y(I)-Y(I-1))*((X(1)-X(I-1))*(X(1)+2.0*X(I-1))
               .      +3.0*X(I-1)**2)
0047          110 CONTINUE
0048            AREA=0.5*AREA
0049            YC=YC/(6.0*AREA)
0050            XC=-XC/(6.0*AREA)
0051            YC=(YC-YL)*SFACT
0052            XC=(XC-XL)*SFACT
0053            AREA=AREA*SFACT*SFACT
0054            A(I2+2)=XC
0055            A(I2+3)=YC
0056            A(I2+7)=AREA*RHO
          C------------------- LOCAL COORDINATES FOR THIS BLOCK ---
0057            M3=M3+NVARB
0058            A(M3)=(X(1)-XL)*SFACT-XC
0059            A(M3+1)=(Y(1)-YL)*SFACT-YC
0060            DO 120 I=2,NC
0061            A(M3+3)=(X(I)-XL)*SFACT-XC
0062            A(M3+4)=(Y(I)-YL)*SFACT-YC
0063            A(M3+2)=SQRT((A(M3+3)-A(M3))**2+(A(M3+4)-A(M3+1))**2)
0064          120 M3=M3+3
0065            A(M3+2)=SQRT((A(I2+NVARB)-A(M3))**2+(A(I2+NVARB+1)-A(M3+1))**2)
0066            M3=M3+3
          C------------------- MOMENT OF INERTIA ---
0067            RMOI=0.0
0068            IC=I2+NVARB
0069            DO 130 NP=2,NC
0070            AREA=A(IC)*A(IC+1) + (A(IC+3)-A(IC))*(A(IC+4)+A(IC+1))
               .      - A(IC+3)*A(IC+4)
0071            AREA=0.5*AREA
0072            TEMP=A(IC)**2+A(IC+1)**2+A(IC+3)**2+A(IC+4)**2
               .      +A(IC)*A(IC+3)+A(IC+1)*A(IC+4)
0073            RMOI=RMOI+AREA*TEMP/6.0
0074          130 IC=IC+3
0075            AREA=A(IC)*A(IC+1)+(A(I2+NVARB)-A(IC))*(A(I2+NVARB+1)+A(IC+1))
               .      - A(I2+NVARB)*A(I2+NVARB+1)
0076            AREA=0.5*AREA
0077            TEMP=A(I2+NVARB)**2+A(I2+NVARB+1)**2+A(IC)**2+A(IC+1)**2
               .      +A(I2+NVARB)*A(IC)+A(I2+NVARB+1)*A(IC+1)
0078            RMOI=RMOI+AREA*TEMP/6.0
0079            A(I2+8)=RMOI*RHO
```

```
FORTRAN IV-PLUS V02-04G          17:36:23     22-MAR-78          PAGE 3
NEXT.FTN         /I4/TR:BLOCKS/WR

        C
        C------------------- BACK TO GLOBAL CO-ORDINATES ---
0080            IC=I2+NVARB
0081            DO 140 NP=1,NC
0082            A(IC)=A(IC)+A(I2+2)
0083            A(IC+1)=A(IC+1)+A(I2+3)
0084          140 IC=IC+3
0085            WRITE(6,2002) XC,YC,A(I2+7),A(I2+8)
0086            GOTO 10
        C------------------- DELETE A BLOCK ---
0087          150 GOTO 40
        C------------------- DUMP MEMORY AS REQUESTED ---
0088          200 LOC1=IVAR(1)
0089            LOC2=IVAR(2)
0090            IF(LOC2.NE.0) GOTO 220
0091            LOC2=LOC1
0092            LOC1=1
0093          220 CALL DUMP
0094            GOTO 10
        C------------------- CYCLE ROUND MOTION AND FORD ---
0095          250 NCYC=IVAR(1)
0096            IF(MCYCLE.EQ.0) CALL BOX
0097            RETURN
        C------------------- STOP COMMAND ---
0098          300 CALL FINISH
        C------------------- PLOT COMMAND ---
0099          350 CALL BPLOT
0100            GOTO 10
        C------------------- RETURN TO PHASE 1 ---
0101          400 CALL SETUP
0102            GOTO 10
        C------------------- SET REAL DATA ---
0103          450 IADR=IVAR(1)
0104            IF(IADR.LE.0.OR.IADR.GT.M7) GOTO 480
0105            A(IADR)=RVAR(2)
0106            GOTO 10
0107          480 WRITE(6,3002)
0108            GOTO 10
        C------------------- SET INTEGER DATA ---
0109          500 IADR=IVAR(1)
0110            IF(IADR.LE.0.OR.IADR.GT.M7) GOTO 480
0111            IA(IADR)=IVAR(2)
0112            GOTO 10
        C------------------- MOMENTUM & ENERGY CHECK ---
0113          550 CALL CHECK
0114            GOTO 10
        C------------------- GRAVITY ---
0115          600 GRAVY=RVAR(1)
0116            GRAVX=RVAR(2)
0117            GOTO 10
        C------------------- CONTACT STIFFNESSES ---
0118          650 STIFN=RVAR(1)
0119            STIFS=RVAR(2)
0120            GOTO 10
        C------------------- RAYLEIGH DAMPING ---
0121          700 PI2=8.0*ATAN(1.0)
```

```
FORTRAN IV-PLUS V02-04G          17:36:23     22-MAR-78          PAGE 4
NEXT.FTN         /I4/TR:BLOCKS/WR

0122            ALPHA=PI2*RVAR(1)*RVAR(2)
0123            BETA=RVAR(1)/(PI2*RVAR(2))
0124            IF(IVAR(3).EQ.0) GO TO 710
0125            ALPHA=0.0
0126            WRITE(6,3003)
0127          710 IF(IVAR(4).EQ.0) GO TO 720
0128            BETA=0.0
0129            WRITE(6,3004)
0130          720 IF(MCYCLE.EQ.0) GO TO 10
0131            BDT=BETA/TDEL
0132            CON1=1.0-ALPHA*TDEL/2.0
0133            CON2=1.0/(1.0+ALPHA*TDEL/2.0)
0134            GOTO 10
              C------------------- FRICTION COEFFICIENT ---
0135          750 FRIC=RVAR(1)
0136            GOTO 10
              C------------------- ZERO ALL VELOCITIES ---
0137          800 DO 820 NB=1,NBLOKS
0138            I2=IA(NB)
0139            A(I2+5)=0.0
0140            A(I2+6)=0.0
0141            A(I2+7)=0.0
0142          820 CONTINUE
0143            GOTO 10
              C------------------- SET BLOCK LOADS ---
0144          850 NB=IVAR(3)
0145            I2=IA(NB)
0146            A(I2+16)=RVAR(1)
0147            A(I2+17)=RVAR(2)
0148            GOTO 10
              C------------------- INTERNAL BLOCK DAMPING ---
0149          900 PI2=8.0*ATAN(1.0)
0150            ALPHB=PI2*RVAR(1)*RVAR(2)
0151            IF(MCYCLE.EQ.0) GOTO 10
0152            CON1B=1.0-ALPHB*TDEL/2.0
0153            CON2B=1.0/(1.0+ALPHB*TDEL/2.0)
0154            GOTO 10
              C------------------- ELASTIC PROPERTIES FOR BLOCKS ---
0155          950 G=RVAR(2)
0156            LAME1=RVAR(1)+4.0*G/3.0
0157            LAME2=RVAR(1)-2.0*G/3.0
0158            GOTO 10
              C------------------- LOCKED OR UNLOCKED JOINTS ---
0159          970 LOCK=AVAR(1).EQ.4HON
0160            GOTO 10
              C-------------------- PLASTICITY CONSTANTS ---
0161          980 YIELD=RVAR(1)
0162            GOTO 10
              C
0163         1000 FORMAT(20A4)
0164         1001 FORMAT(10X,I10,F10.0,I10)
0165         1002 FORMAT(8F10.0)
0166         1003 FORMAT(10X,2I10)
0167         1004 FORMAT(10X,I10,F10.0)
0168         1005 FORMAT(10X,2F10.0)
0169         1006 FORMAT(10X,2F10.0,2I10)
```

```
FORTRAN IV-PLUS V02-04G          17:36:23     22-MAR-78            PAGE 5
NEXT.FTN         /I4/TR:BLOCKS/WR

0170      1007 FORMAT(80A1)
0171      2000 FORMAT(1X,4H+++ ,20A4,4H +++,32X,10A1)
0172      2001 FORMAT(1X,4(1H(,E12.4,1H,,E12.4,3H ) ))
0173      2002 FORMAT(18H XC,YC,MASS,RMOI :,1P4E12.3)
0174      3000 FORMAT(28H !!! ERROR : ILLEGAL COMMAND)
0175      3001 FORMAT(34H !!! ERROR : COMMAND NOT AVAILABLE)
0176      3002 FORMAT(33H !!! ERROR : ADDRESS OUT OF RANGE)
0177      3003 FORMAT(10X,29HMASS DAMPING TERM SET TO ZERO)
0178      3004 FORMAT(10X,34HSTIFFNESS DAMPING TERM SET TO ZERO)
         C
0179            END


PROGRAM SECTIONS

 NAME        SIZE                    ATTRIBUTES

$CODE1   005472  1437                RW,I,CON,LCL
$PDATA   000056    23                RW,D,CON,LCL
$IDATA   000456   151                RW,D,CON,LCL
$VARS    001306   355                RW,D,CON,LCL
$TEMPS   000034    14                RW,D,CON,LCL
.$$$$.   047040 10000                RW,D,OVR,GBL
CBLOCK   000460   152                RW,D,OVR,GBL
PAR      000670   220                RW,D,OVR,GBL


TOTAL SPACE ALLOCATED = 060200 12352

NEXT,O=NEXT
```

```
FORTRAN IV-PLUS V02-04G          17:37:47     22-MAR-78           PAGE 1
PARSE.FTN       /I4/TR:BLOCKS/WR

0001           SUBROUTINE PARSE(ICODE,IERR)
               C
               C    ROUTINE TO PARSE INPUT LINE ACCORDING TO EXPECTED NUMBER
               C    AND TYPE OF ARGUMENTS FOLLOWING A COMMAND WORD.
               C    THE COMMAND WORD IS DISCARDED (IT IS UP TO THE USER TO
               C    INTERPRET THE COMMAND WORD).
               C    WRITTEN BY P.A. CUNDALL, JULY 4, 1977.
               C    MODIFIED NOV 4, 1977 BY P.A. CUNDALL TO USE DECIMAL
               C    CODES INSTEAD OF OCTAL.
               C
               C    NOTES:  1)  ARGUMENTS CAN BE SEPARATED BY BLANKS OR COMMAS
               C    ------  2)  MULTIPLE CONTIGUOUS SEPARATORS ARE TREATED AS ONE
               C            3)  MAXIMUM NUMBER OF ARGUMENTS IS 10
               C            4)  LAST ARGUMENT MUST NOT BE ALPHANUMERIC
               C
               C    INPUT:  1) ICODE (32 BIT WORD). THIS IS PACKED WITH UP TO TEN
               C    ------       NUMBERS DENOTING THE EXPECTED FORM AND NUMBER
               C                OF ARGUMENTS. THE CODES ARE:
               C                                  0 = NOTHING
               C                                  1 = INTEGER
               C                                  2 = REAL
               C                                  3 = ALPHANUMERIC
               C                EXAMPLE: ICODE = 1321  MEANS -
               C                --------                    FIRST PARAMETER IS INTEGER
               C                                            SECOND IS REAL
               C                                            THIRD IS ALPHANUMERIC
               C                                            LAST IS INTEGER
               C            2) LINE(80) ... THIS IS THE INPUT LINE IN 80A1 FORMAT
               C
               C    OUTPUT: 1) RVAR(10),IVAR(10),AVAR(10)
               C    -------    THE INPUT PARAMETERS APPEAR IN THE APPROPRIATE ARRAY.
               C               FOR EXAMPLE, AN INTEGER VARIABLE AS THE THIRD PARAMETER
               C               WOULD TURN UP IN IVAR(3).
               C               UNUSED VARIABLES APPEAR AS ZERO OR BLANK
               C
               C            2) IERR = 4 INDICATES AN ERROR RETURN
               C
0002           COMMON /PAR/ LINE(80),RVAR(10),IVAR(10),AVAR(10)
0003           DIMENSION MLINE(20),LPOINT(20),WORD(5),LCODE(10)
               C
0004           DO 5 I=1,10
0005           RVAR(I)=0.0
0006           IVAR(I)=0
0007         5 AVAR(I)=4H                                              4
               C
0008           IERR=0
0009           NARG=0
0010           LMAX=80
0011           NGAP=0
0012           L=1
0013           GO TO 15
               C
0014        10 IF(LINE(L).NE.1H,.AND.LINE(L).NE.1H ) GO TO 20
0015           NGAP=NGAP+1
0016        15 L=L+1
0017           IF(L.GT.LMAX) GO TO 100
```

```
FORTRAN IV-PLUS V02-04G          17:37:47     22-MAR-78            PAGE 2
PARSE.FTN        /I4/TR:BLOCKS/WR

0018            GO TO 10
0019         20 IF(NGAP.EQ.0) GO TO 15
0020            DO 30 LL=L,LMAX
0021         30 LINE(LL-NGAP)=LINE(LL)
0022            LMAX=LMAX-NGAP
0023            L=L-NGAP
0024            NARG=NARG+1
0025            LPOINT(NARG)=L
0026            NGAP=0
0027            GO TO 15
0028        100 LPOINT(NARG+1)=LMAX-NGAP+1
        C
0029            IC=ICODE
0030            IARG=0
0031            DO 110 I=1,10
0032            ICN=IC/10
0033            LCODE(I)=IC-ICN*10
0034            IC=ICN
0035            IF(LCODE(I).EQ.0) GOTO 120
0036        110 IARG=IARG+1
0037        120 IF(IARG.GE.NARG) GO TO 130
0038        121 WRITE(6,700)
0039        700 FORMAT(30X,17HILLEGAL PARAMETER)
0040            IERR=4
0041            RETURN
0042        130 IF(NARG.EQ.0) RETURN
0043            DO 200 I=1,NARG
0044            L=LPOINT(I)
0045            NSIZE=LPOINT(I+1)-L
0046            DO 140 N=1,20
0047        140 MLINE(N)=1H
0048            IF(LCODE(I).EQ.3) GO TO 170
0049            L1=21-NSIZE
0050            DO 150 N=L1,20
0051            MLINE(N)=LINE(L)
0052        150 L=L+1
0053            ENCODE(20,151,WORD) MLINE
0054        151 FORMAT(20A1)
0055            GO TO (155,165), LCODE(I)
        C---------------INTEGER---------------------------------------
0056        155 DECODE(20,156,WORD,ERR=121) IVAR(I)
0057        156 FORMAT(I20)
0058            GO TO 200
        C---------------REAL------------------------------------------
0059        165 DECODE(20,166,WORD,ERR=121) RVAR(I)
0060        166 FORMAT(F20.0)
0061            GO TO 200
        C---------------ALPHANUMERIC----------------------------------
0062        170 NSIZE=MIN0(4,NSIZE)
0063            DO 180 N=1,NSIZE
0064            MLINE(N)=LINE(L)
0065        180 L=L+1
0066            ENCODE(4,181,AVAR(I)) (MLINE(J),J=1,4)
0067        181 FORMAT(4A1)
0068        200 CONTINUE
0069            RETURN
```

```
FORTRAN IV-PLUS V02-04G          17:38:13     22-MAR-78           PAGE 1
BOX.FTN          /14/TR:BLOCKS/WR

0001           SUBROUTINE BOX
               C
               C------------------- ALL BLOCKS HAVE BEEN CREATED,
               C------------------- BLOCKS CAN NOW BE BOXED -----
               C
0002           INCLUDE 'COMMON.FTN'
0003 *         COMMON A(5000)
0004 *         DIMENSION IA(1)
0005 *         EQUIVALENCE (A,IA)
0006           INCLUDE 'CBLOCK.FTN'
0007 *         LOGICAL LOCK
0008 *         LOGICAL RFLAG,UFLAG,EFLAG
0009 *         REAL LAME1,LAME2
0010 *         COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,M5,M6,M7,
     *        .                NBLOKS,NCYC,MCYCLE,NEMPT,RFLAG,UFLAG,EFLAG,TFRAC,
     *        .                TDEL,IBOXES,JBOXES,XL,XU,YL,YU,BSIZE,SFACT,XSIZE,
     *        .                YSIZE,UDMAX,UMOST,STIFN,STIFS,FRIC,BETA,BDT,ALPHB,
     *        .                CON1,CON2,ALPHA,NVARB,NBR,IBR,NPR,LAME1,LAME2,
     *        .                G,CON1B,CON2B,GRAVX,GRAVY,LOC1,LOC2,NUPDAT,YIELD,
     *        .                LOCK,LBLOCK
               C
0011           M4=M3+NBOXES
0012           M5=M4
               C------------------- INITIALISE BOX POINTERS ---
0013           I1=M4-1
0014           DO 10 I=M3,I1
0015        10 IA(I)=0
               C------------------- LOOP ON EACH BLOCK ---
0016           DO 50 NB=1,NBLOKS
0017           I2=IA(NB)
0018           NC=IA(I2+1)
0019           IC=I2+NVARB
               C------------------- LOOP ON EACH CORNER ---
0020           DO 40 NP=1,NC
0021           X=A(IC)
0022           Y=A(IC+1)
0023           IBOX=MIN0(IFIX(X/BSIZE)+1,IBOXES)
0024           JBOX=MIN0(IFIX(Y/BSIZE),JBOXES-1)
0025           NBOX=JBOX*IBOXES+IBOX
0026           I3=M3+NBOX-1
0027           IA(M5+2)=IA(I3)
0028           IA(I3)=M5
0029           IA(M5)=NP
0030           IA(M5+1)=NB
0031           M5=M5+3
0032           IC=IC+3
0033        40 CONTINUE
               C
0034        50 CONTINUE
               C
0035           M6=M5+NBLOKM
0036           DO 60 I=M5,M6
0037        60 IA(I)=0
               C------------------- CREATE EMPTY LIST TO END OF MEMORY ---
0038           NEMPT=M6
0039           I6=M6+4
```

```
FORTRAN IV-PLUS V02-04G          17:38:13     22-MAR-78            PAGE 2
BOX.FTN          /14/TR:BLOCKS/WR

0040            DO 80 I=I6,M7,13
0041            J=I
0042            IA(I)=I+9
0043         80 CONTINUE
0044            IA(J)=0
        C------------------- DETERMINE TIMESTEP ---
0045            TDEL=1.0E20
0046            DO 100 NB=1,NBLOKS
0047            I2=IA(NB)+7
0048            TN=2.0*SQRT(A(I2)/ST1FN)
0049            TS=2.0*SQRT(A(I2)/ST1FS)
0050            TDEL=AMIN1(TDEL,TN,TS)
0051        100 CONTINUE
0052            TDEL=TDEL*TFRAC
0053            WRITE(6,2000) TDEL
        C------------------- SET UP DAMPING TERMS ---
0054            BDT=BETA/TDEL
0055            CON1=1.0-ALPHA*TDEL/2.0
0056            CON2=1.0/(1.0+ALPHA*TDEL/2.0)
0057            CON1B=1.0-ALPHB*TDEL/2.0
0058            CON2B=1.0/(1.0+ALPHB*TDEL/2.0)
        C
0059            RETURN
        C
0060       2000 FORMAT(30X,17H TIME INCREMENT =,1PE12.4)
        C
0061            END


PROGRAM SECTIONS

 NAME        SIZE                    ATTRIBUTES

$CODE1   002120    552               RW,I,CON,LCL
$PDATA   000004      2               RW,D,CON,LCL
$IDATA   000060     24               RW,D,CON,LCL
$VARS    000104     34               RW,D,CON,LCL
$TEMPS   000006      3               RW,D,CON,LCL
.$$$$.   047040  10000               RW,D,OVR,GBL
CBLOCK   000460    152               RW,D,OVR,GBL


TOTAL SPACE ALLOCATED = 052036 10767

BOX,O=BOX
```

```
FORTRAN IV-PLUS V02-04G          17:38:41     22-MAR-78            PAGE 1
CYCLE.FTN        /I4/TR:BLOCKS/WR

0001            SUBROUTINE CYCLE
                C
                C------------------- DRIVER FOR ITERATIONS ---
                C
0002            INCLUDE 'COMMON.FTN'
0003 *          COMMON A(5000)
0004 *          DIMENSION IA(1)
0005 *          EQUIVALENCE (A,IA)
0006            INCLUDE 'CBLOCK.FTN'
0007 *          LOGICAL LOCK
0008 *          LOGICAL RFLAG,UFLAG,EFLAG
0009 *          REAL LAME1,LAME2
0010 *          COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,M5,M6,M7,
     *         .                NBLOKS,NCYC,MCYCLE,NEMPT,RFLAG,UFLAG,EFLAG,TFRAC,
     *         .                TDEL,IBOXES,JBOXES,XL,XU,YL,YU,BSIZE,SFACT,XSIZE,
     *         .                YSIZE,UDMAX,UMOST,STIFN,STIFS,FRIC,BETA,BDT,ALPHB,
     *         .                CON1,CON2,ALPHA,NVARB,NBR,IBR,NPR,LAME1,LAME2,
     *         .                G,CON1B,CON2B,GRAVX,GRAVY,LOC1,LOC2,NUPDAT,YIELD,
     *         .                LOCK,LBLOCK
                C
0011            DO 100 NCYCLE=1,NCYC
0012            MCYCLE=MCYCLE+1
                C------------------- UPDATE IF NECESSARY ---
0013            IF(UFLAG) CALL UPDAT
                C------------------- SCAN ALL BLOCKS ---
0014            UDMAX=0.0
0015            DO 20 NBR=1,NBLOKS
0016            IBR=IA(NBR)
0017            CALL MOTION(A(IBR))
0018         20 CONTINUE
0019            DO 25 NB=1,NBLOKS
0020            IB=IA(NB)
0021         25 CALL STRESS(A(IB))
                C------------------- EXIT IF NOTHING MOVED ---
0022            IF(UDMAX.EQ.0.0) GOTO 110
                C------------------- UPDATE CONTACTS ? ---
0023            UMOST=UMOST+UDMAX*TDEL
0024            IF(UMOST.LT.1.0) GOTO 30
                C------------------- YES ---
0025            UFLAG=.TRUE.
                C------------------- SCAN ALL CONTACTS ---
0026         30 DO 50 NB=1,NBLOKS
0027            IBE=IA(NB)
0028            J6=M5+NB-1
0029         40 I6=IA(J6)
0030            IF(I6.EQ.0) GOTO 50
0031            JBC=IA(I6+3)
0032            IBC=IA(JBC)
0033            CALL FORD(A(I6),A(IBC),A(IBE))
0034            J6=I6+4
0035            GOTO 40
0036         50 CONTINUE
                C
                C------------------- END CYCLE LOOP ---
                C
0037        100 CONTINUE
```

```
FORTRAN IV-PLUS V02-04G          17:38:41     22-MAR-78          PAGE 2
CYCLE.FTN        /I4/TR:BLOCKS/WR

        C
0038      110 CONTINUE
        C
0039            RETURN
        C
0040            END


PROGRAM SECTIONS

 NAME        SIZE                  ATTRIBUTES

$CODE1   001212    325             RW,I,CON,LCL
$IDATA   000022      9             RW,D,CON,LCL
$VARS    000040     16             RW,D,CON,LCL
$TEMPS   000010      4             RW,D,CON,LCL
.$$$$.   047040  10000             RW,D,OVR,GBL
CBLOCK   000460    152             RW,D,OVR,GBL


TOTAL SPACE ALLOCATED = 051024 10506

CYCLE,O=CYCLE
```

```
FORTRAN IV-PLUS V02-04G          17:39:02     22-MAR-78            PAGE 1
MOTION.FTN       /14/TR:BLOCKS/WR

0001            SUBROUTINE MOTION(B)
                C
                C------------------- LAW OF MOTION FOR A SINGLE BLOCK ---
                C
0002            INCLUDE 'CBLOCK.FTN'
0003  *         LOGICAL LOCK
0004  *         LOGICAL RFLAG,UFLAG,EFLAG
0005  *         REAL LAME1,LAME2
0006  *         COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,M5,M6,M7,
      *        .                NBLOKS,NCYC,MCYCLE,NEMPT,RFLAG,UFLAG,EFLAG,TFRAC,
      *        .                TDEL,IBOXES,JBOXES,XL,XU,YL,YU,BSIZE,SFACT,XSIZE,
      *        .                YSIZE,UDMAX,UMOST,STIFN,STIFS,FRIC,BETA,BDT,ALPHB,
      *        .                CON1,CON2,ALPHA,NVARB,NBR,IBR,NPR,LAME1,LAME2,
      *        .                G,CON1B,CON2B,GRAVX,GRAVY,LOC1,LOC2,NUPDAT,YIELD,
      *        .                LOCK,LBLOCK
0007            DIMENSION B(1)
0008            EQUIVALENCE (FIX,JFIX),(ANC,NC)
                C
0009            FIX=B(1)
                C------------------- IS THIS BLOCK FIXED ? ---
0010            IF(JFIX.NE.0) RETURN
                C------------------- NO ---
                C------------------- VELOCITIES FROM ACCELERATIONS ---
0011            B(5)=(B(5)*CON1+(B(10)/B(8)+GRAVX)*TDEL)*CON2
0012            B(6)=(B(6)*CON1+(B(11)/B(8)+GRAVY)*TDEL)*CON2
0013            B(7)=(B(7)*CON1+(B(12)/B(9))*TDEL)*CON2
0014            B(10)=0.0
0015            B(11)=0.0
0016            B(12)=0.0
                C
0017            ANC=B(2)
                C
                C------------------- DETERMINE AREA & WIDTH ---
0018            XMIN=XSIZE
0019            XMAX=0.0
0020            YMIN=YSIZE
0021            YMAX=0.0
0022            AREA=0.0
0023            K1=NVARB+1
0024            DO 50 NP=1,NC
0025            K2=K1+3
0026            IF(NP.EQ.NC) K2=NVARB+1
0027            X=B(K1)
0028            Y=B(K1+1)
0029            XMIN=AMIN1(XMIN,X)
0030            XMAX=AMAX1(XMAX,X)
0031            YMIN=AMIN1(YMIN,Y)
0032            YMAX=AMAX1(YMAX,Y)
0033            AREA=AREA+(B(K2)-X)*(B(K2+1)+Y)
0034         50 K1=K2
0035            AREA=AREA*2.0
0036            XW=XMAX-XMIN
0037            YW=YMAX-YMIN
0038            AM=TDEL/B(8)
                C------------------- EFFECTIVE MASSES ---
0039            EM11=AM/(XW*XW)
```

```
FORTRAN IV-PLUS V02-04G          17:39:02     22-MAR-78           PAGE 2
MOTION.FTN       /14/TR:BLOCKS/WR

0040            EM12=AM/(YW*YW)
0041            EM21=EM11
0042            EM22=EM12
      C------------------- NEW STRAIN RATES ---
0043            B(13)=(B(13)*CON1B+(4.0*B(21)-B(17)*AREA)*EM11)*CON2B
0044            B(14)=(B(14)*CON1B+(4.0*B(22)-B(18)*AREA)*EM12)*CON2B
0045            B(15)=(B(15)*CON1B+(4.0*B(23)-B(19)*AREA)*EM21)*CON2B
0046            B(16)=(B(16)*CON1B+(4.0*B(24)-B(20)*AREA)*EM22)*CON2B
      C------------------- UPDATE BLOCK CORNERS ---
0047            IC=NVARB+1
0048            DO 150 NPR=1,NC
0049            IC1=IC+1
0050            XARM=B(IC)-B(3)
0051            YARM=B(IC1)-B(4)
0052            XDC=B(5)+B(13)*XARM+(B(14)-B(7))*YARM
0053            YDC=B(6)+B(16)*YARM+(B(15)+B(7))*XARM
0054            UDMAX=AMAX1(UDMAX,ABS(XDC),ABS(YDC))
0055            IBX=B(IC)
0056            IBY=B(IC1)
0057            B(IC)=B(IC)+XDC*TDEL
0058            B(IC1)=B(IC1)+YDC*TDEL
0059            IF(IBX.NE.IFIX(B(IC)).OR.IBY.NE.IFIX(B(IC1))) CALL REBOX
0060      150 IC=IC+3
      C------------------- RIGID BODY DISPLACEMENTS ---
0061            B(3)=B(3)+B(5)*TDEL
0062            B(4)=B(4)+B(6)*TDEL
      C-------------------------------------------------------
0063            B(21)=0.0
0064            B(22)=0.0
0065            B(23)=0.0
0066            B(24)=0.0
0067            RETURN
      C
0068            END


PROGRAM SECTIONS

  NAME        SIZE                  ATTRIBUTES

$CODE1   002240    592              RW,I,CON,LCL
$IDATA   000054     22              RW,D,CON,LCL
$VARS    000154     54              RW,D,CON,LCL
$TEMPS   000012      5              RW,D,CON,LCL
CBLOCK   000460    152              RW,D,OVR,GBL


TOTAL SPACE ALLOCATED = 003162   825

MOTION,O=MOTION
```

```
FORTRAN IV-PLUS V02-04G          17:39:43     22-MAR-78            PAGE 1
FORD.FTN         /I4/TR:BLOCKS/WR

0001           SUBROUTINE FORD(C,BC,BE)
       C
       C------------------- FORCE DISPLACEMENT LAW FOR SINGLE CONTACT ---
       C
0002           INCLUDE 'CBLOCK.FTN'
0003  *        LOGICAL LOCK
0004  *        LOGICAL RFLAG,UFLAG,EFLAG
0005  *        REAL LAME1,LAME2
0006  *        COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,M5,M6,M7,
      *       .                NBLOKS,NCYC,MCYCLE,NEMPT,RFLAG,UFLAG,EFLAG,TFRAC,
      *       .                TDEL,IBOXES,JBOXES,XL,XU,YL,YU,BSIZE,SFACT,XSIZE,
      *       .                YSIZE,UDMAX,UMOST,STIFN,STIFS,FRIC,BETA,BDT,ALPHB,
      *       .                CON1,CON2,ALPHA,NVARB,NBR,IBR,NPR,LAME1,LAME2,
      *       .                G,CON1B,CON2B,GRAVX,GRAVY,LOC1,LOC2,NUPDAT,YIELD,
      *       .                LOCK,LBLOCK
0007           DIMENSION C(1),BC(1),BE(1)
       C
       C------------------- RELATIVE X & Y VELOCITIES ACROSS CONTACT ---
0008           XCC=C(12)-BC(3)
0009           YCC=C(13)-BC(4)
0010           XCE=C(12)-BE(3)
0011           YCE=C(13)-BE(4)
0012           XD=BC(13)*XCC+(BC(14)-BC(7))*YCC+BC(5)
              .  -BE(13)*XCE-(BE(14)-BE(7))*YCE-BE(5)
0013           YD=BC(16)*YCC+(BC(15)+BC(7))*XCC+BC(6)
              .  -BE(16)*YCE-(BE(15)+BE(7))*XCE-BE(6)
       C------------------- SHEAR & NORMAL DISPLACEMENT INCREMENTS ---
0014           DUS=(XD*C(11)+YD*C(10))*TDEL
0015           DUN=(YD*C(11)-XD*C(10))*TDEL
       C------------------- NORMAL FORCE ---
0016           DFN=-DUN*STIFN
0017           C(8)=C(8)+DFN
0018           IF(LOCK) GOTO 10
       C------------------- TEST FOR TENSION ---
0019           IF(C(8).GE.0.0) GOTO 15
0020           C(8)=0.0
0021           C(9)=0.0
0022           DN=0.0
0023           DS=0.0
0024           GOTO 60
       C------------------- LOCKED JOINT ---
0025        10 FRICF=ABS(FRIC*C(8))
0026           GOTO 20
       C------------------- SHEAR FORCE ---
0027        15 FRICF=FRIC*C(8)
0028        20 DFS=DUS*STIFS
0029           C(9)=C(9)+DFS
0030           IF(ABS(C(9)).LE.FRICF) GOTO 40
0031           C(9)=SIGN(FRICF,C(9))
0032           DS=0.0
0033           GOTO 50
       C------------------- DASHPOT FORCES ---
0034        40 DS=BDT*DFS
0035        50 DN=BDT*DFN
       C------------------- GLOBAL CONTACT FORCES ---
0036        60 FYC=(C(9)+DS)*C(10)-(C(8)+DN)*C(11)
```

```
FORTRAN IV-PLUS V02-04G          17:39:43     22-MAR-78            PAGE 2
FORD.FTN         /14/TR:BLOCKS/WR

0037            FXC=(C(9)+DS)*C(11)+(C(8)+DN)*C(10)
      C-------------------- ADD CONTRIBUTION TO BLOCK FORCES ---
0038            BC(10)=BC(10)-FXC
0039            BC(11)=BC(11)-FYC
0040            BC(12)=BC(12)-(FYC*XCC-FXC*YCC)
0041            BE(10)=BE(10)+FXC
0042            BE(11)=BE(11)+FYC
0043            BE(12)=BE(12)+(FYC*XCE-FXC*YCE)
      C-------------------- APPLIED STRESS IN BLOCKS ---
0044            BC(21)=BC(21)-FXC*XCC
0045            BC(22)=BC(22)-FXC*YCC
0046            BC(23)=BC(23)-FYC*XCC
0047            BC(24)=BC(24)-FYC*YCC
0048            BE(21)=BE(21)+FXC*XCE
0049            BE(22)=BE(22)+FXC*YCE
0050            BE(23)=BE(23)+FYC*XCE
0051            BE(24)=BE(24)+FYC*YCE
      C
0052            RETURN
      C
0053            END


PROGRAM SECTIONS

 NAME        SIZE                  ATTRIBUTES

$CODE1   001444    402             RW,I,CON,LCL
$IDATA   000044     18             RW,D,CON,LCL
$VARS    000074     30             RW,D,CON,LCL
$TEMPS   000010      4             RW,D,CON,LCL
CBLOCK   000460    152             RW,D,OVR,GBL


TOTAL SPACE ALLOCATED = 002274   606

FORD,O=FORD
```

```
FORTRAN IV-PLUS V02-04G          17:40:12     22-MAR-78           PAGE 1
STRESS.FTN       /I4/TR:BLOCKS/WR

0001            SUBROUTINE STRESS(B)
                C
                C------------------ INTERNAL STRESSES FROM STRAINRATES ---
                C
0002            INCLUDE 'CBLOCK.FTN'
0003 *          LOGICAL LOCK
0004 *          LOGICAL RFLAG,UFLAG,EFLAG
0005 *          REAL LAME1,LAME2
0006 *          COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,M5,M6,M7,
     *         .                NMLOKS,NCYC,MCYCLE,NEMPT,RFLAG,UFLAG,EFLAG,TFRAC
     *         .                TDEL,IBOXES,JBOXES,XL,XU,YL,YU,BSIZE,SFACT,XSIZE,
     *         .                YSIZE,UDMAX,UMOST,STIFN,STIFS,FRIC,BETA,BDT,ALPHB,
     *         .                CON1,CON2,ALPHA,NVARB,NBR,IBR,NPR,LAME1,LAME2,
     *         .                G,CON1B,CON2B,GRAVX,GRAVY,LOC1,LOC2,NUPDAT,YIELD,
     *         .                LOCK,LBLOCK
0007            DIMENSION B(1)
                C------------------ STRESS ROTATION CORRECTION TERMS ---
0008            S22C=(B(18)+B(19))*B(7)
0009            S12C=(B(17)-B(20))*B(7)
                C------------------ NEW STRESSES -----------------------
0010            B(17)=B(17)+(B(13)*LAME1+B(16)*LAME2-S22C)*TDEL
0011            B(18)=B(18)+(B(14)*2.0*G                  +S12C)*TDEL
0012            B(19)=B(19)+(B(15)*2.0*G                  +S12C)*TDEL
0013            B(20)=B(20)+(B(16)*LAME1+B(13)*LAME2+S22C)*TDEL
                C
0014            RETURN
0015            END


PROGRAM SECTIONS

 NAME        SIZE                  ATTRIBUTES

$CODE1   000274     94             RW,I,CON,LCL
$IDATA   000012      5             RW,D,CON,LCL
$VARS    000010      4             RW,D,CON,LCL
CBLOCK   000460    152             RW,D,OVR,GBL


TOTAL SPACE ALLOCATED = 000776   255

STRESS,O=STRESS
```

```
FORTRAN IV-PLUS V02-04G          17:40:21     22-MAR-78           PAGE 1
REBOX.FTN        /I4/TR:BLOCKS/WR

0001            SUBROUTINE REBOX
                C
                C------------------- ROUTINE TO REBOX A SINGLE BLOCK ---
                C
0002            INCLUDE 'COMMON.FTN'
0003 *          COMMON A(5000)
0004 *          DIMENSION IA(1)
0005 *          EQUIVALENCE (A,IA)
0006            INCLUDE 'CBLOCK.FTN'
0007 *          LOGICAL LOCK
0008 *          LOGICAL RFLAG,UFLAG,EFLAG
0009 *          REAL LAME1,LAME2
0010 *          COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,M5,M6,M7,
     *         .                NBLOKS,NCYC,MCYCLE,NEMPT,RFLAG,UFLAG,EFLAG,TFRAC,
     *         .                TDEL,IBOXES,JBOXES,XL,XU,YL,YU,BSIZE,SFACT,XSIZE,
     *         .                YSIZE,UDMAX,UMOST,STIFN,STIFS,FRIC,BETA,BDT,ALPHB,
     *         .                CON1,CON2,ALPHA,NVARB,NBR,IBR,NPR,LAME1,LAME2,
     *         .                G,CON1B,CON2B,GRAVX,GRAVY,LOC1,LOC2,NUPDAT,YIELD,
     *         .                LOCK,LBLOCK
                C
0011            NB=NBR
0012            IB=IBR
0013            NP=NPR
0014            IC=IB+NVARB+3*(NP-1)
                C------------------- ARE WE AT EDGE OF DOMAIN ? ---
0015            XE=A(IC)
0016            YE=A(IC+1)
0017            IF(XE.GT.0.0.AND.XE.LT.XSIZE.AND.
                .   YE.GT.0.0.AND.YE.LT.YSIZE) GOTO 20
                C------------------- YES, SET MASTER FIX FLAG ---
0018            IA(IB)=2
0019            A(IB+4)=0.0
0020            A(IB+5)=0.0
0021            A(IB+6)=0.0
0022            A(IB+12)=0.0
0023            A(IB+13)=0.0
0024            A(IB+14)=0.0
0025            A(IB+15)=0.0
0026            RETURN
                C------------------- NO, WHICH BOX SHOULD CORNER BE IN ? ---
0027          20 NBOX=IFIX(XE/BSIZE) + IFIX(YE/BSIZE)*IBOXES + 1
                C------------------- SEARCH THIS BOX ---
0028            J4N=M3+NBOX-1
0029          30 I4N=IA(J4N)
0030            IF(I4N.EQ.0) GOTO 40
0031            IF(NB.EQ.IA(I4N+1).AND.NP.EQ.IA(I4N)) RETURN
0032            J4N=I4N+2
0033            GOTO 30
                C------------------- SEARCH THE SURROUNDING BOXES ---
0034          40 NXL=MAX0(MOD(NBOX-1,IBOXES),1)
0035            NXU=MIN0(NXL+2,IBOXES)
0036            NYL=MAX0(NBOX/IBOXES,1)
0037            NYU=MIN0(NYL+2,JBOXES)
0038            DO 80 JBOX=NYL,NYU
0039            MBOX=(JBOX-1)*IBOXES+NXL-1
0040            DO 70 IBOX=NXL,NXU
```

```
FORTRAN IV-PLUS V02-04G          17:40:21     22-MAR-78            PAGE 2
REBOX.FTN        /I4/TR:BLOCKS/WR

0041            MBOX=MBOX+1
0042            IF(MBOX.EQ.NBOX) GOTO 70
0043            J4M=M3+MBOX-1
0044         50 I4M=IA(J4M)
0045            IF(I4M.EQ.0) GOTO 70
0046            IF(NB.EQ.IA(I4M+1).AND.NP.EQ.IA(I4M)) GOTO 60
0047            J4M=I4M+2
0048            GOTO 50
        C------------------- SWAP ENTRIES FOR THIS CORNER ---
0049         60 IA(J4M)=IA(I4M+2)
0050            IA(J4N)=I4M
0051            IA(I4M+2)=0
0052            RETURN
        C------------------- END OF SURROUNDING BOXES SCAN ---
0053         70 CONTINUE
0054         80 CONTINUE
0055            WRITE(6,3000) NP,NB
0056            EFLAG=.TRUE.
0057            CALL FINISH
0058            RETURN
        C
0059       3000 FORMAT(24H ERROR IN REBOX : CORNER,I3,
          .          9H OF BLOCK,I4,20H NOT IN ADJACENT BOX)
        C
0060            END


PROGRAM SECTIONS

 NAME        SIZE                  ATTRIBUTES

$CODE1   002010    516              RW,I,CON,LCL
$PDATA   000010      4              RW,D,CON,LCL
$IDATA   000140     48              RW,D,CON,LCL
$VARS    000110     36              RW,D,CON,LCL
$TEMPS   000010      4              RW,D,CON,LCL
.$$$$.   047040  10000              RW,D,OVR,GBL
CBLOCK   000460    152              RW,D,OVR,GBL


TOTAL SPACE ALLOCATED = 052020 10760

REBOX,O=REBOX
```

```
FORTRAN IV-PLUS V02-04G          17:40:53     22-MAR-78            PAGE 1
UPDAT.FTN       /14/TR:BLOCKS/WR

0001            SUBROUTINE UPDAT
                C
                C------------------- UPDATE ALL CONTACTS ---
                C
0002            INCLUDE 'COMMON.FTN'
0003 *          COMMON A(5000)
0004 *          DIMENSION IA(1)
0005 *          EQUIVALENCE (A,IA)
0006            INCLUDE 'CBLOCK.FTN'
0007 *          LOGICAL LOCK
0008 *          LOGICAL RFLAG,UFLAG,EFLAG
0009 *          REAL LAME1,LAME2
0010 *          COMMON /CBLOCK/ MED(20),NBLOKM,NBOXES,M1,M2,M3,M4,M5,M6,M7,
                *               .                  NBLOKS,NCYC,MCYCLE,NEMPT,RFLAG,UFLAG,EFLAG,TFRAC,
                *               .                  TDEL,IBOXES,JBOXES,XL,XU,YL,YU,BSIZE,SFACT,XSIZE,
                *               .                  YSIZE,UDMAX,UMOST,STIFN,STIFS,FRIC,BETA,BDT,ALPHB,
                *               .                  CON1,CON2,ALPHA,NVARB,NBK,IBR,NPR,LAME1,LAME2,
                *               .                  G,CON1B,CON2B,GRAVX,GRAVY,LOC1,LOC2,NUPDAT,YIELD,
                *               .                  LOCK,LBLOCK
0011            LOGICAL FIRST
                D       LOGICAL DEBUG
                D       DATA DEBUG/.TRUE./
0012            DATA TOL /1.0/
                C
                C------------------- SCAN EACH BLOCK ---
0013            DO 500 NB=1,NBLOKS
0014            I2=IA(NB)
0015            NC=IA(I2+1)
                C------------------- SCAN EACH EDGE OF BLOCK ---
0016            IEND2=I2+NVARB
0017            DO 400 NP=1,NC
0018            FIRST=.TRUE.
0019            IEND1=IEND2
0020            IEND2=IEND1+3
0021            IF(NP.EQ.NC) IEND2=I2+NVARB
                C------------------- DETERMINE BOXES TO BE SEARCHED ---
0022            X1=A(IEND1)
0023            X2=A(IEND2)
0024            Y1=A(IEND1+1)
0025            Y2=A(IEND2+1)
0026            XDIF=X2-X1
0027            YDIF=Y2-Y1
0028            Z=SQRT(XDIF*XDIF+YDIF*YDIF)
0029            A(IEND1+2)=Z
0030            XX1=AMIN1(X1,X2)
0031            XX2=AMAX1(X1,X2)
0032            YY1=AMIN1(Y1,Y2)
0033            YY2=AMAX1(Y1,Y2)
0034            NXL=IFIX((XX1-TOL)/BSIZE)+1
0035            NXU=MIN0(IFIX((XX2+TOL)/BSIZE)+1,IBOXES)
0036            NYL=IFIX((YY1-TOL)/BSIZE)+1
0037            NYU=MIN0(IFIX((YY2+TOL)/BSIZE)+1,JBOXES)
                D       IF(DEBUG) PRINT*,' NPE,NXYLU',NP,NXL,NXU,NYL,NYU
                C------------------- SEARCH EACH CANDIDATE BOX ---
0038            DO 350 JBOX=NYL,NYU
0039            NBOX=(JBOX-1)*IBOXES+NXL-1
```

```
FORTRAN IV-PLUS V02-046          17:40:53     22-MAR-78           PAGE 2
UPDAT.FTN       /14/TR:BLOCKS/WR

0040           DO 300 IBOX=NXL,NXU
0041           NBOX=NBOX+1
0042           I4=IA(M3+NBOX-1)
               C------------------- END OF BOX LIST ? ---
0043         210 CONTINUE
               D      IF(DEBUG) PRINT*,' NBOX,I4,NPC,NBC',NBOX,I4,IA(I4),IA(I4+1)
0044           IF(I4.EQ.0) GOTO 300
               C------------------- NO, IS THIS BLOCK NB ? ---
0045           IF(IA(I4+1).NE.NB) GOTO 220
               C------------------- YES ---
               C------------------- GET NEXT ENTRY IN THIS BOX ---
0046         215 I4=IA(I4+2)
0047           GOTO 210
               C------------------- FIRST TIME THROUGH ? ---
0048         220 IF(.NOT.FIRST) GOTO 230
               C------------------- YES, COMPUTE COS, SIN  FOR THIS EDGE ---
0049           FIRST=.FALSE.
0050           COSA=XDIF/Z
0051           SINA=YDIF/Z
               C------------------- COMPUTE CORNER COORDINATES RELATIVE TO EDGE ---
0052         230 I2C=IA(I4+1)
0053           I2C=IA(I2C)
0054           IC=I2C+IA(I4)*3+NVARB-3
0055           YT=(A(IC+1)-Y1)*COSA
              .  -(A(IC)  -X1)*SINA
               D      IF(DEBUG) PRINT*,' YT',YT
0056           IF(YT.GT.1.0) GOTO 215
0057           IF(YT.LE.-3.0) GOTO 215
0058           XT=(A(IC)  -X1)*COSA
              .  +(A(IC+1)-Y1)*SINA
               D      IF(DEBUG) PRINT*,' XT',XT
0059           IF(XT.GT.2) GOTO 215
0060           IF(XT.LT.0.0) GOTO 215
               C------------------- CONTACT LIST FOR BLOCK NB ---
0061           J6=M5+NB-1
0062           I6=IA(J6)
               C------------------- END OF LIST ? ---
0063         235 CONTINUE
0064           IF(I6.EQ.0) GOTO 250
               C------------------- NO, SAME CORNER AND EDGE ? ---
0065           IF(IA(I4).EQ.IA(I6+2).AND.IA(I4+1).EQ.IA(I6+3)
              .    .AND.NP.EQ.IA(I6+1)) GOTO 240
               C------------------- NO, GET NEXT ENTRY IN CONTACT LIST ---
               D       IF(DEBUG) PRINT*,' GET NEXT ENTRY'
0066           J6=I6+4
0067           I6=IA(J6)
0068           GOTO 235
               C------------------- CONTACT ALREADY STORED ---
0069         240 IF(YT.GT.-2.0) GOTO 245
               C------------------- APPLY DRIFT CORRECTION ---
0070           WRITE(6,3000)
               C------------------- UPDATE CONTACT DATA ---
0071         245 A(I6+9)=SINA
0072           A(I6+10)=COSA
0073           A(I6+11)=A(IC)
0074           A(I6+12)=A(IC+1)
```

```
FORTRAN IV-PLUS V02-04G          17:40:53     22-MAR-78            PAGE 3
UPDAT.FTN        /14/TR:BLOCKS/WR

0075            IA(I6)=1
        D       IF(DEBUG) PRINT*,' UPDATE CONTACT DATA'
0076            GOTO 215
        C------------------- NEW CONTACT ? ---
0077          250 CONTINUE
        D       IF(DEBUG) PRINT*,' GOT TO 250!'
0078            IF(YT.GT.0.0) GOTO 215
        C------------------- YES ---
        D       IF(DEBUG) PRINT*,' NEW CONTACT'
0079            I6=NEMPT
0080            ICR=IC-3
0081            IF(IA(I4).EQ.1) ICR=IC+3*(IA(I2C+1)-1)
0082            YTR=(A(ICR+1)-Y1)*COSA
               .   -(A(ICR)  -X1)*SINA
        D       IF(DEBUG) PRINT*,' YTR',YTR
0083            IF(YTR.LE.-2.0) GOTO 215
0084            ICL=IC+3
0085            IF(IA(I4).EQ.IA(I2C+1)) ICL=I2C+NVARB
0086            YTL=(A(ICL+1)-Y1)*COSA
               .   -(A(ICL)  -X1)*SINA
        D       IF(DEBUG) PRINT*,' YTL',YTL
0087            IF(YTL.LE.-2.0) GOTO 215
        C------------------- ANY SPACE AVAILABLE IN EMPTY LIST ? ---
0088            IF(IA(NEMPT+4).NE.0) GOTO 260
0089            WRITE(6,3001)
0090            CALL FINISH
        C------------------- NEW CONTACT ---
0091          260 A(I6+5)=0.0
0092            A(I6+6)=0.0
0093            A(I6+7)=0.0
0094            A(I6+8)=0.0
0095            IA(I6+1)=NP
0096            IA(I6+2)=IA(I4)
0097            IA(I6+3)=IA(I4+1)
0098            NEMPT=IA(I6+4)
0099            IA(J6)=I6
0100            IA(I6+4)=0
        D       IF(DEBUG) PRINT*,' REALLY IS A NEW CONTACT!!!'
0101            GOTO 245
        C------------------- END OF BOX SCAN ---
0102          300 CONTINUE
0103          350 CONTINUE
        C------------------- END OF EDGE SCAN ---
0104          400 CONTINUE
        C------------------- SCAN CONTACT LIST FOR PRESERVE FLAGS ---
0105            J6=M5+NB-1
0106            I6=IA(J6)
        C------------------- END OF LIST ? ---
0107          410 IF(I6.EQ.0) GOTO 490
        C------------------- NO ---
        C------------------- IS THE PRESERVE FLAG SET ? ---
0108            IF(IA(I6).EQ.0) GOTO 420
        C------------------- YES ---
        D       IF(DEBUG) PRINT*,' PRESERVE FLAG SET'
0109            IA(I6)=0
0110          415 J6=I6+4
```

```
FORTRAN IV-PLUS V02-04G          17:40:53     22-MAR-78            PAGE 4
UPDAT.FTN       /I4/TR:BLOCKS/WR

0111            GOTO 430
               C------------------- NO ---
0112          420 IF(LOCK) GOTO 415
0113              IA(J6)=IA(I6+4)
0114              IA(I6+4)=NEMPT
0115              NEMPT=I6
               D      IF(DEBUG) PRINT*,' PRESERVE FLAG NOT SET'
               C------------------- GET NEXT CONTACT ---
0116          430 I6=IA(J6)
0117              GOTO 410
               C------------------- END OF BLOCK SCAN ---
0118          490 CONTINUE
               D      IF(DEBUG) DEBUG=.FALSE.
0119          500 CONTINUE
               C
0120              NUPDAT=NUPDAT+1
0121              UFLAG=.FALSE.
0122              UMOST=0.0
0123              RETURN
               C
0124         3000 FORMAT(30X,26H DRIFT CORRECTION REQUIRED)
0125         3001 FORMAT(30X,32H NO MORE MEMORY FOR CONTACT LIST)
               C
0126              END


PROGRAM SECTIONS

 NAME        SIZE                  ATTRIBUTES

$CODE1  004236  1103               RW,I,CON,LCL
$PDATA  000004     2               RW,D,CON,LCL
$IDATA  000140    48               RW,D,CON,LCL
$VARS   000234    78               RW,D,CON,LCL
$TEMPS  000022     9               RW,D,CON,LCL
.$$$$.  047040 10000               RW,D,OVR,GBL
CBLOCK  000460   152               RW,D,OVR,GBL


TOTAL SPACE ALLOCATED = 054400 11392

UPDAT,O=UPDAT
```

```
FORTRAN IV-PLUS V02-04G          17:41:56     22-MAR-78           PAGE 1
DUMP.FTN         /I4/TR:BLOCKS/WR

0001            SUBROUTINE DUMP
                C
                C------------------- ROUTINE TO PRINT MEMORY ALLOCATION AND CONTENTS ---
                C
0002            INCLUDE 'COMMON.FTN'
0003  *         COMMON A(5000)
0004  *         DIMENSION IA(1)
0005  *         EQUIVALENCE (A,IA)
0006            INCLUDE 'CBLOCK.FTN'
0007  *         LOGICAL LOCK
0008  *         LOGICAL RFLAG,UFLAG,EFLAG
0009  *         REAL LAME1,LAME2
0010  *         COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,M5,M6,M7,
      *        .                NBLOKS,NCYC,MCYCLE,NEMPT,RFLAG,UFLAG,EFLAG,TFRAC,
      *        .                TDEL,IBOXES,JBOXES,XL,XU,YL,YU,BSIZE,SFACT,XSIZE,
      *        .                YSIZE,UDMAX,UMOST,STIFN,STIFS,FRIC,BETA,BDT,ALPHB,
      *        .                CON1,CON2,ALPHA,NVARB,NBR,IBR,NPR,LAME1,LAME2,
      *        .                G,CON1B,CON2B,GRAVX,GRAVY,LOC1,LOC2,NUPDAT,YIELD,
      *        .                LOCK,LBLOCK
                C
0011            IF(LOC2.EQ.0) RETURN
0012            WRITE(6,2000)
0013            WRITE(6,2005) M1,M2,M3,M4,M5,M6,M7,NBLOKS,NCYC,NEMPT,MCYCLE
0014            IF(LOC2.LT.0) GOTO 100
                C------------------- DUMP CONSECUTIVE MEMORY LOCATIONS ---
0015            WRITE(6,2009)
0016            I2=((LOC1-1)/10)*10
0017          10 I1=I2+1
0018            I2=I1+9
0019            WRITE(6,2001) (IA(I),I=I1,I2),I2
0020            IF(I2.LT.LOC2) GOTO 10
0021            GOTO 500
                C
                C------------------- DUMP BY BOX ---
0022          100 WRITE(6,2002)
0023            DO 120 NBOX=1,NBOXES
0024            I3=IA(M3+NBOX-1)
                C------------------- ANY MORE ENTRIES ? ---
0025          110 IF(I3.EQ.0) GOTO 120
                C------------------- NO, PRINT BLOCK,CORNER ---
0026            WRITE(6,2008) NBOX,IA(I3+1),IA(I3)
0027            I3=IA(I3+2)
0028            GOTO 110
0029          120 CONTINUE
                C------------------- CONTENTS OF EACH BLOCK ---
0030            WRITE(6,2006)
0031            DO 130 NB=1,NBLOKS
0032            IF=IA(NB)
0033            IL=IF+NVARB-1+3*IA(IF+1)
0034            WRITE(6,2003) NB,IA(IF),IA(IF+1),(A(I),I=IF+2,IL)
0035          130 CONTINUE
                C------------------- DUMP CONTACT DATA ---
0036            IF(M5.EQ.0) GOTO 500
0037            WRITE(6,2007)
0038            DO 150 NB=1,NBLOKS
0039            J6=M5+NB-1
```

```
FORTRAN IV-PLUS V02-04G          17:41:56     22-MAR-78           PAGE 2
DUMP.FTN         /I4/TR:BLOCKS/WR

0040       140 I6=IA(J6)
0041           IF(I6.EQ.0) GOTO 150
0042           WRITE(6,2004) NB,(IA(I),I=I6,I6+4),
                             (A(I),I=I6+5,I6+12)
0043           J6=I6+4
0044           GOTO 140
0045       150 CONTINUE
       C-------------------
0046       500 WRITE(6,2000)
       C
0047           RETURN
       C
0048      2000 FORMAT(1X,130(1H-))
0049      2001 FORMAT(1X,10012,I6)
0050      2002 FORMAT(10X,3HBOX,7X,5HBLOCK,6X,6HCORNER)
0051      2003 FORMAT(/1X,3I3,1P11E10.2/(10X,1P11E10.2))
0052      2004 FORMAT(/1X,6I10/1X,1P8E10.2)
0053      2005 FORMAT(9X,2HM1,8X,2HM2,8X,2HM3,8X,2HM4,8X,2HM5,8X,2HM6,
                .       8X,2HM7,4X,6HNBLOKS,6X,4HNCYC,5X,5HNEMPT,4X,6HMCYCLE/
                .       1X,11I10)
0054      2006 FORMAT(11H BLOCK DATA,10(1H-)/1X,9H NB IF NC,8X,2HXC,8X,2HYC,
                .       6X,4HXDOT,6X,4HYDOT,6X,4HIDOT,6X,4HMASS,
                .       7X,3HMOI,5X,5HXFSUM,5X,5HYFSUM,6X,4HMSUM,6X,4HED11
                .      /16X,4HED12,6X,4HED21,6X,4HED22,6X,4HSI11,6X,4HSI12,
                .       6X,4HSI21,6X,4HSI22,6X,4HSA11,6X,4HSA12,6X,4HSA21,
                .       6X,4HSA22)
0055      2007 FORMAT(13H CONTACT DATA,10(1H-)/8X,3HNBE,7X,3HPRE,7X,3HNPE,7X,
                .       3HNPC,7X,3HNBC,6X,4HLINK/10X,1HS,9X,1HN,8X,2HFN,8X,2HFS,
                .       7X,3HSIN,7X,3HCOS,7X,3HXCP,7X,3HYCP)
0056      2008 FORMAT(1X,3I12)
0057      2009 FORMAT(1X,30(1H*),19HVALUES ARE IN OCTAL,30(1H*))
       C
0058           END


PROGRAM SECTIONS

 NAME        SIZE                  ATTRIBUTES

$CODE1   002272    605             RW,I,CON,LCL
$PDATA   000010      4             RW,D,CON,LCL
$IDATA   001026    267             RW,D,CON,LCL
$VARS    000050     20             RW,D,CON,LCL
$TEMPS   000016      7             RW,D,CON,LCL
.$$$$.   047040  10000             RW,D,OVR,GBL
CBLOCK   000460    152             RW,D,OVR,GBL


TOTAL SPACE ALLOCATED = 053136 11055

NO FPP INSTRUCTIONS GENERATED

DUMP,O=DUMP
```

```
FORTRAN IV-PLUS V02-04G          17:42:37     22-MAR-78           PAGE 1
BPLOT.FTN       /I4/TR:BLOCKS/WR

0001            SUBROUTINE BPLOT
                C
                C------------------- PLOT A SNAPSHOT OF THE GEOMETRY ---
                C
0002            INCLUDE 'COMMON.FTN'
0003  *         COMMON A(5000)
0004  *         DIMENSION IA(1)
0005  *         EQUIVALENCE (A,IA)
0006            INCLUDE 'CBLOCK.FTN'
0007  *         LOGICAL LOCK
0008  *         LOGICAL RFLAG,UFLAG,EFLAG
0009  *         REAL LAME1,LAME2
0010  *         COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,M5,M6,M7,
      *        .                NBLOKS,NCYC,MCYCLE,NEMPT,RFLAG,UFLAG,EFLAG,TFRAC,
      *        .                TDEL,IBOXES,JBOXES,XL,XU,YL,YU,BSIZE,SFACT,XSIZE,
      *        .                YSIZE,UDMAX,UMOST,STIFN,STIFS,FRIC,BETA,BDT,ALPHB,
      *        .                CON1,CON2,ALPHA,NVARB,NBR,IBR,NPR,LAME1,LAME2,
      *        .                G,CON1B,CON2B,GRAVX,GRAVY,LOC1,LOC2,NUPDAT,YIELD,
      *        .                LOCK,LBLOCK
                C
0011            SCREEN=10.0
0012            FACT=SCREEN/AMAX1(XSIZE,YSIZE)
                C------------------- PLOT EACH BLOCK ---
0013            DO 100 NB=1,NBLOKS
0014            I2=IA(NB)
0015            NC=IA(I2+1)
0016            XC=FACT*A(I2+2)
0017            YC=FACT*A(I2+3)
                C------------------- FIXED ? ---
0018            IF(IA(I2).EQ.0) GOTO 20
                C------------------- YES, ALREADY PLOTTED ? ---
0019            IF(IA(I2).GT.2) GOTO 100
                C------------------- NO ---
0020            IA(I2)=IA(I2)+2
0021            CALL SYMBOL(XC,YC,0.2,1HF,0.0,1)
0022          20 IC=I2+NVARB
0023            X=FACT*A(IC)
0024            Y=FACT*A(IC+1)
0025            CALL PLOT(X,Y,3)
0026            DO 50 NP=2,NC
0027            IC=IC+3
0028            X=FACT*A(IC)
0029            Y=FACT*A(IC+1)
0030            CALL PLOT(X,Y,2)
0031          50 CONTINUE
0032            X=FACT*A(I2+NVARB)
0033            Y=FACT*A(I2+NVARB+1)
0034            CALL PLOT(X,Y,2)
                C------------------- END OF BLOCK LOOP ---
0035         100 CONTINUE
                C
0036            CALL PLOT(0.0,0.0,3)
                C
0037            RETURN
                C
0038            END
```

```
FORTRAN IV-PLUS V02-04G          17:43:00     22-MAR-78           PAGE 1
FINISH.FTN       /14/TR:BLOCKS/WR

0001            SUBROUTINE FINISH
                C
                C------------------- TIDY UP AND STOP ---
                C
0002            INCLUDE 'COMMON.FTN'
0003 *          COMMON A(5000)
0004 *          DIMENSION IA(1)
0005 *          EQUIVALENCE (A,IA)
0006            INCLUDE 'CBLOCK.FTN'
0007 *          LOGICAL LOCK
0008 *          LOGICAL RFLAG,UFLAG,EFLAG
0009 *          REAL LAME1,LAME2
0010 *          COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,M5,M6,M7,
     *         .                NBLOKS,NCYC,MCYCLE,NEMPT,RFLAG,UFLAG,EFLAG,TFRAC,
     *         .                TDEL,IBOXES,JBOXES,XL,XU,YL,YU,BSIZE,SFACT,XSIZE,
     *         .                YSIZE,UDMAX,UMOST,STIFN,STIFS,FRIC,BETA,BDT,ALPHB,
     *         .                CON1,CON2,ALPHA,NVARB,NBR,IBR,NPR,LAME1,LAME2,
     *         .                G,CON1B,CON2B,GRAVX,GRAVY,LOC1,LOC2,NUPDAT,YIELD,
     *         .                LOCK,LBLOCK
                C
0011            CALL PLOTND
0012            WRITE(6,2000) MCYCLE,NUPDAT
                C
                C------------------- WRITE RESTART FILE IF NO ERRORS ---
0013            IF(EFLAG) GOTO 100
0014            REWIND 1
0015            WRITE(1) (HED(I),I=1,LBLOCK)
0016            WRITE(1) (A(I),I=1,M7)
0017            WRITE(6,2001)
0018          100 STOP
                C
0019           2000 FORMAT(30X,13H TOTAL CYCLES,I10/
                .       30X,13H NO. UPDATES ,I10)
0020           2001 FORMAT(30X,32H A RESTART FILE HAS BEEN WRITTEN)
                C
0021            END


PROGRAM SECTIONS

 NAME        SIZE                  ATTRIBUTES

$CODE1  000364   122                RW,I,CON,LCL
$IDATA  000120    40                RW,D,CON,LCL
$VARS   000004     2                RW,D,CON,LCL
.$$$$.  047040 10000                RW,D,OVR,GBL
CBLOCK  000460   152                RW,D,OVR,GBL


TOTAL SPACE ALLOCATED = 050230 10316

NO FPP INSTRUCTIONS GENERATED

FINISH,O=FINISH
```

```
FORTRAN IV-PLUS V02-04G          17:43:17     22-MAR-78           PAGE 1
CHECK.FTN       /I4/TR:BLOCKS/WR

0001           SUBROUTINE CHECK
               C
               C------------------- MOMENTUM AND ENERGY CHECK ---
               C
0002           INCLUDE 'COMMON.FTN'
0003 *         COMMON A(5000)
0004 *         DIMENSION IA(1)
0005 *         EQUIVALENCE (A,IA)
0006           INCLUDE 'CBLOCK.FTN'
0007 *         LOGICAL LOCK
0008 *         LOGICAL RFLAG,UFLAG,EFLAG
0009 *         REAL LAME1,LAME2
0010 *         COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,M5,M6,M7,
     *        .                NBLOKS,NCYC,MCYCLE,NEMPT,RFLAG,UFLAG,EFLAG,TFRAC,
     *        .                TDEL,IBOXES,JBOXES,XL,XU,YL,YU,BSIZE,SFACT,XSIZE,
     *        .                YSIZE,UDMAX,UMOST,STIFN,STIFS,FRIC,BETA,BDT,ALPHB,
     *        .                CON1,CON2,ALPHA,NVARB,NBR,IBR,NPR,LAME1,LAME2,
     *        .                G,CON1B,CON2B,GRAVX,GRAVY,LOC1,LOC2,NUPDAT,YIELD,
     *        .                LOCK,LBLOCK
               C
0011           WRITE(6,3000)
0012           GO TO 200
0013           XMOM=0.0

(XMOM=0.0)
ERROR  26
W NO PATH TO THIS STATEMENT

0014           YMOM=0.0
0015           TMOM=0.0
0016           TARM=0.0
0017           XKE=0.0
0018           YKE=0.0
0019           TKE=0.0
0020           DO 100 NB=1,NBLOKS
0021           I2=IA(NB)
0022           XMOM=XMOM+A(I2+11)*A(I2+5)
0023           YMOM=YMOM+A(I2+11)*A(I2+6)
0024           TMOM=TMOM+A(I2+12)*A(I2+7)
0025           TARM=TARM+A(I2+11)*(A(I2+2)*A(I2+6)-A(I2+3)*A(I2+5))
0026           XKE=XKE+0.5*A(I2+11)*A(I2+5)**2
0027           YKE=YKE+0.5*A(I2+11)*A(I2+6)**2
0028           TKE=TKE+0.5*A(I2+12)*A(I2+7)**2
0029           WRITE(6,2001) XMOM,YMOM,TMOM,TARM,XKE,YKE,TKE
0030         100 CONTINUE
               C
0031           ESUM=XKE+YKE+TKE
0032           TSUM=TMOM+TARM
               C
0033           WRITE(6,2000) XMOM,YMOM,TMOM,TARM,TSUM,XKE,YKE,TKE,ESUM
               C
0034         200 RETURN
               C
0035        2000 FORMAT(9H MOMENTUM,11X,1HX,11X,1HY,2X,10HROTATIONAL/
                   .       9X,1P5E12.4//
                   .       9H ENERGY  ,11X,1HX,11X,1HY,2X,10HROTATIONAL,7X,5HTOTAL/
```

```
FORTRAN IV-PLUS V02-04G          17:43:17     22-MAR-78          PAGE 2
CHECK.FTN        /I4/TR:BLOCKS/WR

                .          9X,1P4E12.4)
0036       2001 FORMAT(1X,1P7E12.4)
0037       3000 FORMAT(30X,17HNOT AVAILABLE YET)
          C
0038             END


PROGRAM SECTIONS

 NAME         SIZE                    ATTRIBUTES

$CODE1   000674    222                RW,I,CON,LCL
$IDATA   000176     63                RW,D,CON,LCL
$VARS    000054     22                RW,D,CON,LCL
.$$$$.   047040  10000                RW,D,OVR,GBL
CBLOCK   000460    152                RW,D,OVR,GBL


TOTAL SPACE ALLOCATED = 050666 10459

CHECK,O=CHECK
```

APPENDIX XIV: LISTING OF PROGRAM RBMC

```
FORTRAN IV-PLUS V02-04G          14:34:43     29-MAR-78            PAGE 1
RBMC.FTN          /14/TR:BLOCKS/WR

0001            PROGRAM RBMC
                C
                C------------------ RIGID BLOCK MODEL ---
                C
                C------------------ ORIGINAL PROGRAM BY P.A.CUNDALL
                C------------------ TRANSLATION INTO FORTRAN BY P.J.BERESFORD
                C------------------ INTRODUCTION OF CRACKING BY N.C.LAST,MARCH 1978
                C
                C
                C
0002            INCLUDE 'COMMON.FTN'
0003 *          COMMON A(3000)
0004 *          DIMENSION IA(1)
0005 *          EQUIVALENCE (A,IA)
0006            INCLUDE 'CBLOCK.FTN'
0007 *          COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,
     *         .                NBLOKS,NCYC,MCYCLE,NEMPT,RHO,EFLAG,TFRAC,
     *         .                TDEL,IBOXES,JBOXES,XL,XU,YL,YU,BSIZE,SFACT,XSIZE,
     *         .                YSIZE,UDMAX,UMOST,STIFN,STIFS,FRIC,BETA,BDT,TMAX,
     *         .                CON1,CON2,ALPHA,GRAVX,GRAVY,LOC1,LOC2,NUPDAT,NB1,
     *         .                NVARB,NFRAG,NPR,NEMPTC,NEMPTD,NEMPTG,IEND1,LBLOCK
0008 *          LOGICAL EFLAG
                C
0009            DATA STIFN,STIFS,FRIC,BETA/2*1.0E8,0.0,0.0/
0010            DATA GRAVY,GRAVX/-9.81,0.0/
0011            DATA M4,MCYCLE,NUPDAT,LBLOCK/3000,0,0,68/
0012            DATA EFLAG/.FALSE./
0013            DATA NVARB,NFRAG/14,10/
                C
0014            CALL PLOTST(.025,'CM')
0015            CALL SETUPC
0016         10 CALL NEXTC
0017            CALL CYCLEC
0018            GO TO 10
                C
0019            END


PROGRAM SECTIONS

 NAME        SIZE                  ATTRIBUTES

$CODE1  000076     31              RW,I,CON,LCL
$PDATA  000010      4              RW,D,CON,LCL
$IDATA  000010      4              RW,D,CON,LCL
.$$$$.  027340   6000              RW,D,OVR,GBL
CBLOCK  000424    138              RW,D,OVR,GBL


TOTAL SPACE ALLOCATED = 030102  6177

NO FPP INSTRUCTIONS GENERATED
```

```
FORTRAN IV-PLUS V02-04G          14:34:48     29-MAR-78            PAGE 2
RBMC.FTN         /14/TR:BLOCKS/WR

0001           SUBROUTINE SETUPC
       C
0002           INCLUDE 'COMMON.FTN'
0003  *        COMMON A(3000)
0004  *        DIMENSION IA(1)
0005  *        EQUIVALENCE (A,IA)
0006           INCLUDE 'CBLOCK.FTN'
0007  *        COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,
      *       .                NBLOKS,NCYC,MCYCLE,NEMPT,RHO,EFLAG,TFRAC,
      *       .                TDEL,IBOXES,JBOXES,XL,XU,YL,YU,BSIZE,SFACT,XSIZE,
      *       .                YSIZE,UDMAX,UMOST,STIFN,STIFS,FRIC,BETA,BDT,TMAX,
      *       .                CON1,CON2,ALPHA,GRAVX,GRAVY,LOC1,LOC2,NUPDAT,NB1,
      *       .                NVARB,NFRAG,NPR,NEMPTC,NEMPTD,NEMPTG,IEND1,LBLOCK
0008  *        LOGICAL EFLAG
       C
0009           DIMENSION CARD(20),WORD(10)
0010           BYTE DAY(10),TIM(10)
       C
0011           DATA WORD /4HSTAR,4HREST,4HBLOC,4HBOXE,4HXLIM,
              .          4HYLIM,4H****,4HFRAC,4HDENS,4HISTR/
       C
0012           JOB=0
0013           CALL DATE(DAY)
0014           CALL TIME(TIM)
0015           WRITE(6,2000) DAY,TIM
       C------------------- READ NEXT CARD ---
0016        10 READ(5,1000) CARD
0017           CALL TIME(TIM)
0018           WRITE(6,2001) CARD,TIM
       C
0019           DO 20 I=1,10
0020           IF(CARD(1).EQ.WORD(I)) GOTO 30
0021        20 CONTINUE
       C
0022           WRITE(6,3000)
0023           GOTO 10
       C
0024        30 IF(I.GT.2.AND.JOB.EQ.0) GOTO 60
0025           GOTO (110,120,130,140,150,
              .      160,900,170, 180, 190), I
       C
0026        50 WRITE(6,3001)
0027           GOTO 10
       C
0028        60 WRITE(6,3002)
0029           GOTO 10
       C------------------- START OF NEW RUN ---
0030       110 JOB=1
0031           DO 111 I=1,20
0032       111 HED(I)=CARD(I)
0033           GOTO 10
       C------------------- RESTART RUN ---
0034       120 JOB=2
0035           READ(1) (HED(I),I=1,LBLOCK)
0036           READ(1) (A(I),I=1,M4)
0037           WRITE(6,2003) MCYCLE
```

```
FORTRAN IV-PLUS V02-04G          14:34:48     29-MAR-78           PAGE 3
RBMC.FTN         /14/TR:BLOCKS/wR

0038            RETURN
              C------------------- MAXIMUM NUMBER OF BLOCKS ---
0039          130 DECODE(20,1001,CARD) NBLOKM
0040              GOTO 10
              C------------------- NUMBER OF BOXES ---
0041          140 DECODE(40,1001,CARD) IBOXES,JBOXES,IBSIZE
0042              BSIZE=FLOAT(IBSIZE)
0043              NBOXES=IBOXES*JBOXES
0044              GOTO 10
              C------------------- PROBLEM X-LIMITS ---
0045          150 DECODE(30,1002,CARD) XL,XU
0046              IF(XL.LT.XU) GOTO 10
0047              TEMP=XL
0048              XL=XU
0049              XU=TEMP
0050              GOTO 10
              C------------------- PROBLEM Y-LIMITS ---
0051          160 DECODE(30,1002,CARD) YL,YU
0052              IF(YL.LT.YU) GOTO 10
0053              TEMP=YL
0054              YL=YU
0055              YU=TEMP
0056              GOTO 10
              C------------------- FRACTION OF CRITICAL TIMESTEP ---
0057          170 DECODE(20,1002,CARD) TFRAC
0058              GOTO 10
              C------------------- DENSITY OF BLOCKS ---
0059          180 DECODE(20,1003,CARD) RHO
0060              GO TO 10
              C------------------- MAXIMUM TENSILE STRENGTH ---
0061          190 DECODE(20,1003,CARD) TMAX
0062              GO TO 10
              C------------------- SETUP FINISHED ---
0063          900 CONTINUE
              C------------------- INITIALISE SOME VARIABLES ---
0064              UMOST=0.0
0065              NBLOKS=0
0066              XSIZE=FLOAT(IBOXES)*BSIZE
0067              YSIZE=FLOAT(JBOXES)*BSIZE
0068              XF=XSIZE/(XU-XL)
0069              YF=YSIZE/(YU-YL)
0070              SFACT=AMIN1(XF,YF)
0071              M1=1
0072              M2=M1+(NFRAG+1)*NBLOKM
0073              M3=M2+NBOXES
0074              NEMPTG=M3
0075              NEMPTC=0
0076              NEMPTD=0
0077              I1=M2+NBOXES-1
0078              DO 5 I=M2,I1
0079            5 IA(I)=0
0080              WRITE(6,2002) NBLOKM,RHO,TMAX,NBOXES,XL,XU,YL,YU,BSIZE,SFACT
0081              RETURN
              C
0082         1000 FORMAT(20A4)
0083         1001 FORMAT(10X,3I10)
```

```
FORTRAN IV-PLUS V02-04G          14:34:48     29-MAR-78           PAGE 4
RBMC.FTN         /14/TR:BLOCKS/WR

0084      1002 FORMAT(10X,5F10.0)
0085      1003 FORMAT(10X,F10.0)
0086      2000 FORMAT(30X,26H RBMC - RIGID BLOCK MODEL.,55X,10A1,1X,10A1/
                .        30X,26H ------------------------- / )
0087      2001 FORMAT(1X,4H+++ ,20A4,4H +++,32X,10A1)
0088      2002 FORMAT(/30X,25H MAXIMUM NUMBER OF BLOCKS,15/
                .        30X,25H DENSITY OF BLOCKS        ,F10.2/
                .        30X,25H TENSILE STRENGTH         ,F10.2/
                .        30X,25H NUMBER OF BOXES          ,15/
                .        30X,25H X-LIMITS                 ,2F10.2/
                .        30X,25H Y LIMITS                 ,2F10.2/
                .        30X,25H BSIZE                    ,F10.2/
                .        30X,25H SFACT                    ,F10.2)
0089      2003 FORMAT(30X,31H RESTART RUN. CURRENT CYCLES ..,I10)
0090      3000 FORMAT(28H !!! ERROR : ILLEGAL COMMAND)
0091      3001 FORMAT(34H !!! ERROR : COMMAND NOT AVAILABLE)
0092      3002 FORMAT(48H !!! ERROR : 'START' OR 'RESTART' CARD NOT FOUND)
          C
0093           END


PROGRAM SECTIONS

 NAME        SIZE                  ATTRIBUTES

$CODE1   002534    686              RW,I,CON,LCL
$PDATA   000026     11              RW,D,CON,LCL
$IDATA   001106    291              RW,D,CON,LCL
$VARS    000250     84              RW,D,CON,LCL
$TEMPS   000004      2              RW,D,CON,LCL
.$$$$.   027340   6000              RW,D,OVR,GBL
CBLOCK   000424    138              RW,D,OVR,GBL


TOTAL SPACE ALLOCATED = 034130  7212
```

```
FORTRAN IV-PLUS V02-04G          14:35:09     29-MAR-78           PAGE 5
RBMC.FTN         /14/TR:BLOCKS/WR

0001            SUBROUTINE NEXTC
                C
0002            INCLUDE 'COMMON.FTN'
0003 *          COMMON A(3000)
0004 *          DIMENSION IA(1)
0005 *          EQUIVALENCE (A,IA)
0006            INCLUDE 'CBLOCK.FTN'
0007 *          COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,
                *               .                 NBLOKS,NCYC,MCYCLE,NEMPT,RHO,EFLAG,TFRAC,
                *               .                 TDEL,IBOXES,JBOXES,XL,XU,YL,YU,BSIZE,SFACT,XSIZE,
                *               .                 YSIZE,UDMAX,UMOST,STIFN,STIFS,FRIC,BETA,BDT,TMAX,
                *               .                 CON1,CON2,ALPHA,GRAVX,GRAVY,LOC1,LOC2,NUPDAT,NB1,
                *               .                 NVARB,NFRAG,NPR,NEMPTC,NEMPTD,NEMPTG,IEND1,LBLOCK
0008 *          LOGICAL EFLAG
                C
0009            DIMENSION CARD(20),WORD(20),X(50),Y(50)
0010            BYTE TIM(10)
                C
0011            DATA WORD /4HCREA,4HDELE,4HDUMP,4HCYCL,4HSTOP,
                .            4HPLOT,4H****,4HRSET,4HISET,4HZZZZ,
                .            4HGRAV,4HSTIF,4HDAMP,4HFRIC,4HZERO,
                .            4HZZZZ,4HZZZZ,4HZZZZ,4HZZZZ,4HZZZZ/
                C
                C------------------- READ NEXT CARD ---
0012          10 READ(5,1001) CARD
0013             CALL TIME(TIM)
0014             WRITE(6,2000) CARD,TIM
0015             DO 20 I=1,20
0016             IF(CARD(1).EQ.WORD(I)) GOTO 30
0017          20 CONTINUE
0018             WRITE(6,3000)
0019             GOTO 10
                C------------------- JUMP TO APPROPRIATE CODE ---
0020          30 GOTO (100,150,200,250,300,
                .       350,400,450,500,10,
                .       600,650,700,750,800,
                .       10, 10, 10, 10, 10), I
                C
0021          40 WRITE(6,3001)
0022             GOTO 10
                C------------------- CREATE A NEW BLOCK ---
0023         100 NBLOKS=NBLOKS+1
0024             DECODE(30,1003,CARD) NC,JFIX
0025             CALL LIMIT(NVARB+3*NC)
0026             READ(5,1002) (X(I),Y(I),I=1,NC)
0027             WRITE(6,2001) (X(I),Y(I),I=1,NC)
0028             NEMPT=NEMPTG
0029             I2=NEMPT
0030             NEMPT=I2+NVARB
0031             IA(NBLOKS)=I2
0032             IA(I2+12)=NEMPT
0033             IA(I2+13)=0
0034             IF(JFIX.NE.0) IA(I2)=1
0035             IA(I2+1)=NC
                C------------------- AREA AND CENTROID OF THIS BLOCK ---
0036             AREA=(X(1)-X(NC))*(Y(1)+Y(NC))
```

```
FORTRAN IV-PLUS V02-04G          14:35:09     29-MAR-78           PAGE 6
RBMC.FTN         /14/TR:BLOCKS/WR

0037            YC=(X(1)-X(NC))*((Y(1)-Y(NC))*(Y(1)+2.0*Y(NC))+3.0*Y(NC)**2)
0038            XC=(Y(1)-Y(NC))*((X(1)-X(NC))*(X(1)+2.0*X(NC))+3.0*X(NC)**2)
0039            DO 110 I=2,NC
0040            AREA=AREA+(X(I)-X(I-1))*(Y(I)+Y(I-1))
0041            YC=YC+(X(I)-X(I-1))*((Y(I)-Y(I-1))*(Y(I)+2.0*Y(I-1))
               .      +3.0*Y(I-1)**2)
0042            XC=XC+(Y(I)-Y(I-1))*((X(I)-X(I-1))*(X(I)+2.0*X(I-1))
               .      +3.0*X(I-1)**2)
0043          110 CONTINUE
0044            AREA=0.5*AREA
0045            YC=YC/(6.0*AREA)
0046            XC=-XC/(6.0*AREA)
0047            YC=(YC-YL)*SFACT
0048            XC=(XC-XL)*SFACT
0049            AREA=AREA*SFACT*SFACT
0050            A(12+2)=XC
0051            A(12+3)=YC
0052            A(12+7)=AREA*RHO
              C------------------- LOCAL COORDINATES FOR THIS BLOCK ---
0053            DO 120 I=1,NC
0054            A(NEMPT)=(X(I)-XL)*SFACT-XC
0055            A(NEMPT+1)=(Y(I)-YL)*SFACT-YC
0056            IA(NEMPT+2)=NEMPT+3
0057          120 NEMPT=NEMPT+3
0058            IA(NEMPT-1)=IA(12+12)
              C------------------- MOMENT OF INERTIA ---
0059            RMOI=0.0
0060            IC=IA(12+12)
0061            DO 130 NP=1,NC
0062            IC1=IA(IC+2)
0063            AREA=A(IC)*A(IC+1) + (A(IC1)-A(IC))*(A(IC1+1)+A(IC+1))
               .      - A(IC1)*A(IC1+1)
0064            AREA=0.5*AREA
0065            TEMP=A(IC)**2+A(IC+1)**2+A(IC1)**2+A(IC1+1)**2
               .      +A(IC)*A(IC1)+A(IC+1)*A(IC1+1)
0066            RMOI=RMOI+AREA*TEMP/6.0
0067          130 IC=IC1
0068            A(12+8)=RMOI*RHO
              C
0069            WRITE(6,2002) A(12+2),A(12+3),A(12+7),A(12+8)
              C
              C------------------- GLOBAL COORDINATES ---
0070            IC=IA(12+12)
0071            DO 140 I=1,NC
0072            A(IC)=A(IC)+A(12+2)
0073            A(IC+1)=A(IC+1)+A(12+3)
0074          140 IC=IA(IC+2)
0075            NEMPTG=NEMPT
0076            GOTO 10
              C------------------- DELETE A BLOCK ---
0077          150 GOTO 40
              C------------------- DUMP MEMORY AS REQUESTED ---
0078          200 DECODE(30,1003,CARD) LOC1,LOC2
0079            IF(LOC2.NE.0) GOTO 220
0080            LOC2=LOC1
0081            LOC1=1
```

```
FORTRAN IV-PLUS V02-04G          14:35:09     29-MAR-78          PAGE 7
RBMC.FTN          /I4/TR:BLOCKS/RR

0082       220 CALL DUMPC
0083           GOTO 10
           C------------------- CYCLE ROUND MOTION AND FORD ---
0084       250 DECODE(20,1003,CARD) NCYC
0085           IF(MCYCLE.NE.0) GO TO 280
           C------------------- DETERMINE TIMESTEP ---
0086           TDEL=1.0E20
0087           DO 260 NB1=1,NBLOKS
0088           I2=IA(NB1)
0089           TN=2.0*SQRT(A(I2+7)/STIFN)
0090           TS=2.0*SQRT(A(I2+7)/STIFS)
0091           TDEL=AMIN1(TDEL,TN,TS)
0092       260 CONTINUE
0093           TDEL=TDEL*TFRAC
0094           WRITE(6,3005) TDEL
0095           BDT=BETA/TDEL
0096           CON1=1.0-ALPHA*TDEL/2.0
0097           CON2=1.0/(1.0+ALPHA*TDEL/2.0)
           C------------------- BOXING OF BLOCKS ---
0098           DO 270 NB1=1,NBLOKS
0099           CALL BOXC
0100       270 CONTINUE
0101       280 RETURN
           C------------------- STOP COMMAND ---
0102       300 CALL FINISH
           C------------------- PLOT COMMAND ---
0103       350 CALL BPLOTC
0104           GOTO 10
           C------------------- RETURN TO PHASE 1 ---
0105       400 CALL SETUPC
0106           GOTO 10
           C------------------- SET REAL DATA ---
0107       450 DECODE(30,1004,CARD) IADR,VAL
0108           IF(IADR.LE.0.OR.IADR.GT.M4) GOTO 480
0109           A(IADR)=VAL
0110           GOTO 10
0111       480 WRITE(6,3002)
0112           GOTO 10
           C------------------- SET INTEGER DATA ---
0113       500 DECODE(30,1003,CARD) IADR,IVAL
0114           IF(IADR.LE.0.OR.IADR.GT.M7) GOTO 480
0115           IA(IADR)=IVAL
0116           GOTO 10
           C------------------- GRAVITY ---
0117       600 DECODE(30,1005,CARD) GRAVY,GRAVX
0118           GOTO 10
           C------------------- CONTACT STIFFNESSES ---
0119       650 DECODE(30,1005,CARD) STIFN,STIFS
0120           GO TO 10
           C------------------- RAYLEIGH DAMPING ---
0121       700 DECODE(50,1006,CARD) FRAC,FREQ,IF1,IF2
0122           PI2=8.0*ATAN(1.0)
0123           ALPHA=PI2*FRAC*FREQ
0124           BETA=FRAC/(PI2*FREQ)
0125           IF(IF1.EQ.0) GO TO 720
0126           ALPHA=0.0
```

```
FORTRAN IV-PLUS V02-04G          14:35:09     29-MAR-78           PAGE 8
RBMC.FTN          /I4/IR:BLOCKS/WR

0127          . WRITE(6,3003)
0128        720 IF(IF2.EQ.0) GO TO 10
0129            BETA=0.0
0130            WRITE(6,3004)
0131            GO TO 10
              C------------------- FRICTION COEFFICIENT ---
0132        750 DECODE(20,1005,CARD) FRIC
0133            GOTO 10
              C------------------- ZERO ALL VELOCITIES ---
0134        800 DO 820 NB=1,NBLOKS
0135            I2=IA(NB)
0136            A(I2+4)=0.0
0137            A(I2+5)=0.0
0138            A(I2+6)=0.0
0139        820 CONTINUE
0140            GOTO 10
              C
0141       1001 FORMAT(20A4)
0142       1002 FORMAT(8F10.0)
0143       1003 FORMAT(10X,2I10)
0144       1004 FORMAT(10X,I10,F10.0)
0145       1005 FORMAT(10X,2F10.0)
0146       1006 FORMAT(10X,2F10.0,2I10)
0147       2000 FORMAT(1X,4H+++ ,20A4,4H +++,32X,10A1)
0148       2001 FORMAT(1X,4(1H(,E12.4,1H,,E12.4,3H ) ))
0149       2002 FORMAT(18H XC,YC,MASS,RMOI :,1P4E12.3)
0150       3000 FORMAT(28H !!! ERROR : ILLEGAL COMMAND)
0151       3001 FORMAT(34H !!! ERROR : COMMAND NOT AVAILABLE)
0152       3002 FORMAT(33H !!! ERROR : ADDRESS OUT OF RANGE)
0153       3003 FORMAT(10X,29HMASS DAMPING TERM SET TO ZERO)
0154       3005 FORMAT(X/,30X,17H TIME INCREMENT =,1PE12.4)
0155       3004 FORMAT(10X,34HSTIFFNESS DAMPING TERM SET TO ZERO)
              C
0156            END


PROGRAM SECTIONS

 NAME        SIZE                    ATTRIBUTES

$CODE1   005314   1382               RW,I,CON,LCL
$PDATA   000056     23               RW,D,CON,LCL
$IDATA   000546    179               RW,D,CON,LCL
$VARS    001232    333               RW,D,CON,LCL
$TEMPS   000030     12               RW,D,CON,LCL
.$$$$.   027340   6000               RW,D,OVR,GBL
CBLOCK   000424    138               RW,D,OVR,GBL


TOTAL SPACE ALLOCATED = 037406  8067
```

```
FORTRAN IV-PLUS V02-04G          14:36:00     29-MAR-78           PAGE 9
RBMC.FTN         /14/TR:BLOCKS/WR

0001           SUBROUTINE BOXC
               C
               C------------------- ROUTINE TO BOX BLOCKS ---
               C
0002           INCLUDE 'COMMON.FTN'
0003  *        COMMON A(3000)
0004  *        DIMENSION IA(1)
0005  *        EQUIVALENCE (A,IA)
0006           INCLUDE 'CBLOCK.FTN'
0007  *        COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,
      *       .                NBLOKS,NCYC,MCYCLE,NEMPT,RHO,EFLAG,TFRAC,
      *       .                TDEL,IBOXES,JBOXES,XL,XU,YL,YU,BSIZE,SFACT,XSIZE,
      *       .                YSIZE,UDMAX,UMOST,STIFN,STIFS,FRIC,BETA,BDT,TMAX,
      *       .                CON1,CON2,ALPHA,GRAVX,GRAVY,LOC1,LOC2,NUPDAT,NB1,
      *       .                NVARB,NFRAG,NPR,NEMPTC,NEMPTD,NEMPTG,IEND1,LBLOCK
0008  *        LOGICAL EFLAG
0009           DATA TOL/1.0/
               C
0010           I2=IA(NB1)
0011           NC=IA(I2+1)
               C------------------- DETERMINE CORNER COORDINATES ---
0012           IEND1=IA(I2+12)
0013           X1=A(IEND1)
0014           Y1=A(IEND1+1)
0015           IEND2=IA(IEND1+2)
0016           X2=A(IEND2)
0017           Y2=A(IEND2+1)
0018           XSTEP=1.0
0019           YSTEP=1.0
0020           NP=0
0021           I2N=0
               C------------------- START BOXING ---
0022         150 NP=NP+1
0023           NS=1
0024           XTOL=0.0
0025           YTOL=0.0
               C------------------- DETERMINE SCANNING DIRECTION ---
0026           IF(ABS(X2-X1).GT.ABS(Y2-Y1)) NS=0
               C------------------- DETERMINE X AND Y INCREMENTS ---
0027           IF(X1.GT.X2) XSTEP=-XSTEP
0028           IF(Y1.GT.Y2) YSTEP=-YSTEP
               C------------------- DETERMINE TOLERANCES ---
0029           IF(NS.EQ.0) XTOL=TOL*XSTEP
0030           IF(NS.EQ.1) YTOL=TOL*YSTEP
               C------------------- DETERMINE LIMITS OF EDGE ---
0031           IBOX1=MIN0(IFIX((X1-XTOL)/BSIZE)+1,IBOXES)
0032           JBOX1=MIN0(IFIX((Y1-YTOL)/BSIZE),JBOXES-1)
0033           IBOXN=MIN0(IFIX((X2+XTOL)/BSIZE)+1,IBOXES)
0034           JBOXN=MIN0(IFIX((Y2+YTOL)/BSIZE),JBOXES-1)
0035           NBOX1=JBOX1*IBOXES+IBOX1
0036           NBOXN=JBOXN*IBOXES+IBOXN
               C------------------- SET BOX POINTERS ---
0037           I21=M2+NBOX1-1
0038           I22=M2+NBOXN-1
0039           IF(I2N.EQ.I22) GO TO 750
0040           I2N=0
```

```
FORTRAN IV-PLUS V02-04G          14:36:00     29-MAR-78            PAGE 10
RBMC.FTN         /I4/TR:BLOCKS/WR

0041            GO TO 450
              C------------------- DETERMINE DIRECTION OF BOX INCREMENT ---
0042          200 IF(IBOX1.EQ.IBOXN) GO TO 650
0043              IF(JBOX1.EQ.JBOXN) GO TO 700
0044              IF(I2N.EQ.0) GO TO 210
0045              IF(NS.EQ.0) GO TO 650
0046              GO TO 700
              C------------------- SET UP EQUATION OF THE LINE ---
0047          210 TEMP1=(Y2-Y1)/(X2-X1)
0048              TEMP2=Y1-X1*TEMP1
0049              IF(NS.EQ.0) GO TO 250
              C------------------- DETERMINE COORDINATES OF INTERSECTION
              C                    WITH BOX GRID ---
0050              YN=FLOAT(JBOX1)*BSIZE
0051              IF(Y1.GT.Y2) YN=YN+BSIZE
0052              GO TO 300
0053          250 XN=FLOAT(IBOX1-1)*BSIZE
0054              IF(X1.GT.X2) XN=XN+BSIZE
0055          300 IF(NS.EQ.0) GO TO 350
0056              YN=YN+YSTEP*BSIZE
0057              XN=(YN-TEMP2)/TEMP1
0058              GO TO 400
0059          350 XN=XN+XSTEP*BSIZE
0060              YN=TEMP1*XN+TEMP2
              C------------------- DETERMINE THE BOX IN WHICH THE
              C                    INTERSECTION OCCURS ---
0061          400 JBOXN=MIN0(IFIX((YN+YTOL)/BSIZE),JBOXES-1)
0062              IBOXN=MIN0(IFIX((XN+XTOL)/BSIZE)+1,IBOXES)
0063              NBOXN=JBOXN*IBOXES+IBOXN
              C------------------- SET BOX POINTER ---
0064              I2N=M2+NBOXN-1
0065              GO TO 200
              C------------------- IS BLOCK ALREADY ENTERED ?---
0066          450 IF(IA(I21).EQ.0) GO TO 550
0067              I23=IA(I21)
0068          500 IF(ABS(IA(I23)).EQ.NB1) GO TO 600
0069              IF(IA(I23+1).EQ.0) GO TO 550
0070              I23=IA(I23+1)
0071              GO TO 500
              C------------------- NO, PLACE ENTRY IN EMPTY 'DOUBLES' LIST ---
0072          550 CALL EMPTYD
0073              IA(NEMPT+1)=IA(I21)
0074              IA(I21)=NEMPT
0075              IA(NEMPT)=-NB1
              C------------------- YES ---
0076          600 IF(IA(I23).EQ.NB1) IA(I23)=-IA(I23)
              C------------------- IS ALL THE EDGE BOXED ?---
0077              IF(I21.EQ.I22) GO TO 750
0078              IF(I21.EQ.I2N) GO TO 300
              C------------------- NO, INCREMENT BOX NUMBER ---
0079              GO TO 200
              C------------------- INCREMENT IN Y DIRECTION ---
0080          650 I21=I21+IFIX(YSTEP)*IBOXES
0081              JBOX1=JBOX1+IFIX(YSTEP)
0082              GO TO 450
              C------------------- INCREMENT IN X DIRECTION ---
```

```
FORTRAN IV-PLUS V02-04G          14:36:00     29-MAR-78            PAGE 11
RBMC.FTN          /I4/TR:BLOCKS/WR

0083       700 I21=I21+IFIX(XSTEP)
0084           IBOX1=IBOX1+IFIX(XSTEP)
0085           GO TO 450
0086       750 XSTEP=ABS(XSTEP)
0087           YSTEP=ABS(YSTEP)
           C------------------- IS THE BLOCK BOXED ?---
0088           IF(NP.EQ.NC) GO TO 800
           C------------------- NO, BOX NEXT EDGE ---
0089           I2N=I22
0090           X1=A(IEND2)
0091           Y1=A(IEND2+1)
0092           IEND2=IA(IEND2+2)
0093           X2=A(IEND2)
0094           Y2=A(IEND2+1)
0095           GO TO 150
           C------------------- YES, DELETE OBSOLETE ENTRIES ---
0096       800 CONTINUE
0097           CALL DELENT
0098           RETURN
0099           END


PROGRAM SECTIONS

 NAME        SIZE                    ATTRIBUTES

$CODE1   003302   865                RW,I,CON,LCL
$IDATA   000046    19                RW,D,CON,LCL
$VARS    000160    56                RW,D,CON,LCL
$TEMPS   000004     2                RW,D,CON,LCL
.$$$$.   027340  6000                RW,D,OVR,GBL
CBLOCK   000424   138                RW,D,OVR,GBL


TOTAL SPACE ALLOCATED = 033520  7080
```

```
FORTRAN IV-PLUS V02-04G          14:36:31     29-MAR-78           PAGE 12
RBMC.FTN         /14/TR:BLOCKS/WR

0001            SUBROUTINE CYCLEC
                C
                C------------------- DRIVER FOR ITERATIONS ---
                C
0002            INCLUDE 'COMMON.FTN'
0003 *          COMMON A(3000)
0004 *          DIMENSION IA(1)
0005 *          EQUIVALENCE (A,IA)
0006            INCLUDE 'CBLOCK.FTN'
0007 *          COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,
                *              .                NBLOKS,NCYC,MCYCLE,NEMP1,RHO,EFLAG,TFRAC,
                *              .                TDEL,IBOXES,JBOXES,XL,XU,YL,YU,BSIZE,SFACT,XSIZE,
                *              .                YSIZE,UDMAX,UMOST,STIFN,STIFS,FRIC,BETA,BDT,TMAX,
                *              .                CON1,CON2,ALPHA,GRAVX,GRAVY,LOC1,LOC2,NUPDAT,NB1,
                *              .                NVARB,NFRAG,NPR,NEMPTC,NEMPTD,NEMPTG,IEND1,LBLOCK
0008 *          LOGICAL EFLAG
                C
0009            DO 100 NCYCLE=1,NCYC
0010            MCYCLE=MCYCLE+1
                C------------------- SCAN ALL BLOCKS ---
0011            UDMAX=0.0
0012            DO 20 NB1=1,NBLOKS
0013            IBR=IA(NB1)
0014            CALL MOTIOC(A(IBR))
0015         20 CONTINUE
                C------------------- EXIT IF NOTHING MOVED ---
0016            IF(UDMAX.EQ.0) RETURN
                C------------------- CALL FORD ---
0017            DO 50 NB1=1,NBLOKS
0018         50 CALL FORDC
                C
                C------------------- CHECK FOR CRACKING ---
0019            DO 60 NB1=1,NBLOKS
0020            I2=IA(NB1)
                C------------------- FIXED BLOCKS NOT CONSIDERED ---
0021            IF(IA(I2).NE.0) GO TO 60
0022            CALL CFORCE
0023         60 CONTINUE
                C
                C------------------- END CYCLE LOOP ---
                C
0024        100 CONTINUE
                C
0025            RETURN
                C
0026            END


PROGRAM SECTIONS

 NAME        SIZE                  ATTRIBUTES

$CODE1   000614    198            RW,I,CON,LCL
$IDATA   000006      3            RW,D,CON,LCL
$VARS    000014      6            RW,D,CON,LCL
$TEMPS   000010      4            RW,D,CON,LCL
```

```
FORTRAN IV-PLUS V02-04G          14:36:39     29-MAR-78           PAGE 14
RBMC.FTN         /14/TR:BLOCKS/WR

0001           SUBROUTINE MOTIOC(B)
               C
               C------------------- LAW OF MOTION FOR A SINGLE BLOCK ---
               C
0002           INCLUDE 'COMMON.FTN'
0003 *         COMMON A(3000)
0004 *         DIMENSION IA(1)
0005 *         EQUIVALENCE (A,IA)
0006           INCLUDE 'CBLOCK.FTN'
0007 *         COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,
     *        .                NBLOKS,NCYC,MCYCLE,NEMPT,RHO,EFLAG,TFRAC,
     *        .                TDEL,IBOXES,JBOXES,XL,XU,YL,YU,BSIZE,SFACT,XSIZE,
     *        .                YSIZE,UDMAX,UMOST,STIFN,STIFS,FRIC,BETA,BDT,TMAX,
     *        .                CON1,CON2,ALPHA,GRAVX,GRAVY,LOC1,LOC2,NUPDAT,NB1,
     *        .                NVARB,NFRAG,NPR,NEMPTC,NEMPTD,NEMPTG,IEND1,LBLOCK
0008 *         LOGICAL EFLAG
0009           DIMENSION B(1)
0010           EQUIVALENCE (FIX,JFIX),(A1C,IC),(ANC,NC)
               C
0011           FIX=B(1)
               C------------------- IS THIS BLOCK FIXED ? ---
0012           IF(JFIX.NE.0) RETURN
               C------------------- NO ---
               C------------------- VELOCITIES FROM ACCELERATIONS ---
0013           B(5)=(B(5)*CON1+(B(10)/B(8)+GRAVX)*TDEL)*CON2
0014           B(6)=(B(6)*CON1+(B(11)/B(8)+GRAVY)*TDEL)*CON2
0015           B(7)=(B(7)*CON1+(B(12)/B(9))*TDEL)*CON2
0016           B(10)=0.0
0017           B(11)=0.0
0018           B(12)=0.0
               C------------------- UPDATE BLOCK CORNERS ---
               C
0019           A1C=B(13)
0020           ANC=B(2)
               C
0021           DO 50 NPR=2,NC+1
0022           IEND1=IC
0023           IEND2=IA(IEND1+2)
0024           XARM=A(IEND2)-B(3)
0025           YARM=A(IEND2+1)-B(4)
0026           XDC=B(5)-B(7)*YARM
0027           YDC=B(6)+B(7)*XARM
0028           UDMAX=AMAX1(UDMAX,ABS(XDC),ABS(YDC))
0029           IBX=A(IEND2)
0030           IBY=A(IEND2+1)
0031           A(IEND2)=A(IEND2)+XDC*TDEL
0032           A(IEND2+1)=A(IEND2+1)+YDC*TDEL
               C------------------- IS MOVEMENT GREATER THAN 1.0 UNITS ? ---
0033           IF(IBX.EQ.IFIX(A(IEND2)).AND.IBY.EQ.IFIX(A(IEND2+1))) GO TO 50
               C------------------- YES,CALL SCAN AND DETECT ---
0034           CALL SCAN(2)
0035           IEND1=IC
0036           CALL DETECT
               C------------------- NO,UPDATE NEXT CORNER ---
0037          50 IC=IA(IEND1+2)
               C------------------- RIGID BODY DISPLACEMENTS ---
```

```
FORTRAN IV-PLUS V02-04G          14:36:39     29-MAR-78          PAGE 15
RBMC.FTN          /I4/TR:BLOCKS/WR

0038            B(3)=B(3)+B(5)*TDEL
0039            B(4)=B(4)+B(6)*TDEL
      C-------------------------------------------------------
0040            RETURN
      C
0041            END


PROGRAM SECTIONS

 NAME        SIZE                    ATTRIBUTES

$CODE1  001156    311               RW,I,CON,LCL
$PDATA  000004      2               RW,D,CON,LCL
$IDATA  000030     12               RW,D,CON,LCL
$VARS   000050     20               RW,D,CON,LCL
$TEMPS  000006      3               RW,D,CON,LCL
.$$$$.  027340   6000               RW,D,OVR,GBL
CBLOCK  000424    138               RW,D,OVR,GBL


TOTAL SPACE ALLOCATED = 031254  6486
```

```
FORTRAN IV-PLUS V02-04G          14:36:53     29-MAR-78           PAGE 16
RBMC.FTN         /I4/TR:BLOCKS/WR

0001           SUBROUTINE FORDC
               C
               C------------------- FORCE DISPLACEMENT LAW FOR SINGLE CONTACT ---
               C
0002           INCLUDE 'COMMON.FTN'
0003 *         COMMON A(3000)
0004 *         DIMENSION IA(1)
0005 *         EQUIVALENCE (A,IA)
0006           INCLUDE 'CBLOCK.FTN'
0007 *         COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,
     *        .                NBLOKS,NCYC,MCYCLE,NEMPT,RHO,EFLAG,TFRAC,
     *        .                TDEL,IBOXES,JBOXES,XL,XU,YL,YU,BSIZE,SFACT,XSIZE,
     *        .                YSIZE,UDMAX,UMOST,STIFN,STIFS,FRIC,BETA,BDT,TMAX,
     *        .                CON1,CON2,ALPHA,GRAVX,GRAVY,LOC1,LOC2,NUPDAT,NB1,
     *        .                NVARB,NFRAG,NPR,NEMPTC,NEMPTD,NEMPTG,IEND1,LBLOCK
0008 *         LOGICAL EFLAG
               C
               C------------------- SCAN CONTACT LIST ---
               C
0009           IBE=IA(NB1)
0010        10 JC1=IA(IBE+13)
0011           JTEMP=IBE+13
0012        20 IF(JC1.EQ.0) RETURN
               C------------------  RETURN IF NO CONTACTS OR END OF LIST ---
0013           IC1=IA(JC1)
0014           NB2=IA(IC1+3)
0015           IF(NB1.EQ.NB2) GO TO 30
0016           IBC=IA(NB2)
0017           GO TO 40
0018        30 JTEMP=JC1+1
0019           JC1=IA(JC1+1)
0020           GO TO 20
               C
               C------------------- RELATIVE X AND Y VELOCITIES ACROSS CONTACTS ---
0021        40 XCC=A(IC1+10)-A(IBC+2)
0022           YCC=A(IC1+11)-A(IBC+3)
0023           XCE=A(IC1+10)-A(IBE+2)
0024           YCE=A(IC1+11)-A(IBE+3)
0025           XD=(A(IBC+4)-A(IBC+6)*YCC)-(A(IBE+4)-A(IBE+6)*YCE)
0026           YD=(A(IBC+5)+A(IBC+6)*XCC)-(A(IBE+5)+A(IBE+6)*XCE)
               C------------------- SHEAR AND NORMAL DISPLACEMENT INCREMENTS ---
0027           DUS=(XD*A(IC1+9)+YD*A(IC1+8))*TDEL
0028           DUN=(YD*A(IC1+9)-XD*A(IC1+8))*TDEL
0029           A(IC1+4)=A(IC1+4)+DUS
0030           A(IC1+5)=A(IC1+5)+DUN
               C------------------- NORMAL FORCE ---
0031           DFN=-DUN*STIFN
0032           A(IC1+6)=A(IC1+6)+DFN
               C------------------- TEST FOR TENSION ---
0033           IF(A(IC1+6).GE.0.0) GO TO 50
0034           A(IC1+6)=0.0
0035           A(IC1+7)=0.0
0036           DN=0.0
0037           DS=0.0
               C------------------- SHEAR FORCE ---
0038        50 DFS=DUS*STIFS
```

```
FORTRAN IV-PLUS V02-04G          14:36:53     29-MAR-78           PAGE 17
RBMC.FTN         /14/TR:BLOCKS/WR

0039            A(IC1+7)=A(IC1+7)+DFS
0040            FRICF=FRIC*A(IC1+6)
0041            IF(ABS(A(IC1+7)).LE.FRICF) GO TO 60
0042            A(IC1+7)=SIGN(FRICF,A(IC1+7))
0043            DS=0.0
0044            GOTO 70
          C------------------- DASHPOT FORCES ---
0045         60 DS=BDT*DFS
0046         70 DN=BDT*DFN
          C------------------- GLOBAL CONTACT FORCES ---
0047            FYC=(A(IC1+7)+DS)*A(IC1+8)-(A(IC1+6)+DN)*A(IC1+9)
0048            FXC=(A(IC1+7)+DS)*A(IC1+9)+(A(IC1+6)+DN)*A(IC1+8)
          C------------------- ADD CONTRIBUTION TO BLOCK FORCES ---
0049            A(IBC+9)=A(IBC+9)-FXC
0050            A(IBC+10)=A(IBC+10)-FYC
0051            A(IBC+11)=A(IBC+11)-(FYC*(A(IC1+10)-A(IBC+2))-FXC*(A(IC1+11)
               .           -A(IBC+3)))
0052            A(IBE+9)=A(IBE+9)+FXC
0053            A(IBE+10)=A(IBE+10)+FYC
0054            A(IBE+11)=A(IBE+11)+(FYC*(A(IC1+10)-A(IBE+2))-FXC*(A(IC1+11)
               .           -A(IBE+3)))
          C
0055            GO TO 30
          C
0056            END


PROGRAM SECTIONS

 NAME        SIZE                  ATTRIBUTES

$CODE1   001430    396             RW,I,CON,LCL
$IDATA   000006      3             RW,D,CON,LCL
$VARS    000124     42             RW,D,CON,LCL
$TEMPS   000002      1             RW,D,CON,LCL
.$$$$.   027340   6000             RW,D,OVR,GBL
CBLOCK   000424    138             RW,D,OVR,GBL


TOTAL SPACE ALLOCATED = 031550  6580
```

```
FORTRAN IV-PLUS V02-04G          14:37:12     29-MAR-78            PAGE 18
RBMC.FTN         /I4/TR:BLOCKS/WR

0001            SUBROUTINE CFORCE
                C
                C------------------- LOCATION OF CONTACT DATA ---
                C
0002            INCLUDE 'COMMON.FTN'
0003 *          COMMON A(3000)
0004 *          DIMENSION IA(1)
0005 *          EQUIVALENCE (A,IA)
0006            INCLUDE 'CBLOCK.FTN'
0007 *          COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,
     *         .                NBLOKS,NCYC,MCYCLE,NEMPT,RHO,EFLAG,TFRAC,
     *         .                TDEL,IBOXES,JBOXES,XL,XU,YL,YU,BSIZE,SFACT,XSIZE,
     *         .                YSIZE,UDMAX,UMOST,STIFN,STIFS,FRIC,BETA,BDT,TMAX,
     *         .                CON1,CON2,ALPHA,GRAVX,GRAVY,LOC1,LOC2,NUPDAT,NB1,
     *         .                NVARB,NFRAG,NPR,NEMPTC,NEMPTD,NEMPTG,IEND1,LBLOCK
0008 *          LOGICAL EFLAG
                C
0009            I2=IA(NB1)
0010            NP=0
                C------------------- SCAN CONTACT LIST ---
0011            JC1=IA(I2+13)
0012            NEMPT=NEMPTG-1
                C------------------- NO ENTRIES OR END OF LIST ?---
0013          100 IF(JC1.EQ.0) GO TO 200
0014            IC1=IA(JC1)
0015            IF(IA(IC1+3).EQ.NB1) GO TO 150
                C------------------- STORE POINTERS TO CONTACTS INVOLVING
                C                    THE CORNER OF ANOTHER BLOCK ---
0016            CALL LIMIT(1)
0017            NEMPT=NEMPT+1
0018            IA(NEMPT)=IC1
0019            NP=NP+1
                C------------------- GET NEXT CONTACT POINTER ---
0020          150 JC1=IA(JC1+1)
0021            GO TO 100
                C------------------- TEST CRACK CRITERION ---
0022          200 IF(NP.LT.2) GO TO 250
0023            CALL CRACK
0024          250 RETURN
0025            END


PROGRAM SECTIONS

 NAME        SIZE                  ATTRIBUTES

$CODE1   000402   129              RW,I,CON,LCL
$PDATA   000004     2              RW,D,CON,LCL
$IDATA   000006     3              RW,D,CON,LCL
$VARS    000020     8              RW,D,CON,LCL
.$$$$.   027340  6000              RW,D,OVR,GBL
CBLOCK   000424   138              RW,D,OVR,GBL


TOTAL SPACE ALLOCATED = 030420  6280
```

```
FORTRAN IV-PLUS V02-04G          14:37:18     29-MAR-78            PAGE 20
RBMC.FTN         /14/TR:BLOCKS/WR

0001           SUBROUTINE CRACK
               C
               C------------------- CRACK CRITERION ---
               C
0002           INCLUDE 'COMMON.FTN'
0003  *        COMMON A(3000)
0004  *        DIMENSION IA(1)
0005  *        EQUIVALENCE (A,IA)
0006           INCLUDE 'CBLOCK.FTN'
0007  *        COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,
      *       .                NBLOKS,NCYC,MCYCLE,NEMPT,RHO,EFLAG,TFRAC,
      *       .                TDEL,IBOXES,JBOXES,XL,XU,YL,YU,BSIZE,SFACT,XSIZE,
      *       .                YSIZE,UDMAX,UMOST,STIFN,STIFS,FRIC,BETA,BDT,TMAX,
      *       .                CON1,CON2,ALPHA,GRAVX,GRAVY,LOC1,LOC2,NUPDAT,NB1,
      *       .                NVARB,NFRAG,NPR,NEMPTC,NEMPTD,NEMPTG,IEND1,LBLOCK
0008  *        LOGICAL EFLAG
               C
0009           TENS2=0.0
0010           NTEMP1=NEMPTG+1
0011           NTEMP2=NEMPTG
               C------------------- CHECK EACH POSSIBLE CRACK ---
0012           IFN2=IA(NEMPTG)
0013           IFN3=IA(NEMPTG+1)
0014           FN2=A(IFN2+6)
               C
0015         100 IF(IFN2.EQ.IFN3) GO TO 150
0016           IF(IA(IFN2+1).EQ.IA(IFN3+1)) GO TO 150
0017           FN3=A(IFN3+6)
               C------------------- CALCULATE TENSILE STRESS ---
0018           Z=(A(IFN3+10)-A(IFN2+10))**2
              . +(A(IFN3+11)-A(IFN2+11))**2
0019           IF(Z.LT.1.0) GO TO 150
0020           TENS1=(FN2+FN3)/Z
               C------------------ FIND MAXIMUM TENSILE STRESS ---
0021           TENS2=AMAX1(TENS1,TENS2)
0022           IF(TENS2.GT.TENS1) GO TO 150
               C------------------- SET POINTERS TO CONTACTS CAUSING
               C                   MAXIMUM TENSILE STRESS ---
0023           IF2=IFN2
0024           IF3=IFN3
               C
0025         150 NTEMP1=NTEMP1+1
0026           IF(NTEMP1.GT.NEMPT) GO TO 200
0027           IFN3=IA(NTEMP1)
0028           GO TO 100
               C
0029         200 NTEMP2=NTEMP2+1
0030           IF(NTEMP2.GT.NEMPT) GO TO 300
0031           IFN2=IA(NTEMP2)
0032           IFN3=IA(NEMPTG)
0033           FN2=A(IFN2+6)
0034           NTEMP1=NEMPTG
0035           GO TO 100
               C-------------------END OF CHECK ---
               C
               C-------------------IS TENSILE STRENGTH EXCEEDED ?---
```

```
FORTRAN IV-PLUS V02-04G          14:37:18     29-MAR-78          PAGE 21
RBMC.FTN         /I4/TR:BLOCKS/WR

0036       300 IF(TENS2.LT.TMAX) GO TO 400
          C------------------ YES,INTRODUCE CRACK ---
0037            CALL SPLIT(IF2,IF3)
          C------------------ NO,RETURN ---
0038       400 RETURN
0039           END


PROGRAM SECTIONS

 NAME        SIZE                    ATTRIBUTES

$CODE1   000730   236                RW,I,CON,LCL
$IDATA   000014     6                RW,D,CON,LCL
$VARS    000054    22                RW,D,CON,LCL
.$$$$.   027340  6000                RW,D,OVR,GBL
CBLOCK   000424   138                RW,D,OVR,GBL

TOTAL SPACE ALLOCATED = 031004  6402
```

```
FORTRAN IV-PLUS V02-04G          14:37:31     29-MAR-78           PAGE 22
RBMC.FTN         /I4/TR:BLOCKS/WR

0001           SUBROUTINE SPLIT(IF2,IF3)
               C
               C------------------- CRACK INTRODUCED ---
               C
0002           INCLUDE 'COMMON.FTN'
0003  *        COMMON A(3000)
0004  *        DIMENSION IA(1)
0005  *        EQUIVALENCE (A,IA)
0006           INCLUDE 'CBLOCK.FTN'
0007  *        COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,
      *       .                NBLOKS,NCYC,MCYCLE,NEMPT,RHO,EFLAG,TFRAC,
      *       .                TDEL,IBOXES,JBOXES,XL,XU,YL,YU,BSIZE,SFACT,XSIZE,
      *       .                YSIZE,UDMAX,UMOST,STIFN,STIFS,FRIC,BETA,BDT,TMAX,
      *       .                CON1,CON2,ALPHA,GRAVX,GRAVY,LOC1,LOC2,NUPDAT,NB1,
      *       .                NVARB,NFRAG,NPR,NEMPTC,NEMPTD,NEMPTG,IEND1,LBLOCK
0008  *        LOGICAL EFLAG
               C
0009           I2=IA(NB1)
0010           NC1=IA(I2+1)
               C------------------ SUPPRESS CRACKS TOO CLOSE TO
               C                   EXISTING CORNERS ---
0011           X1=A(IF2+10)
0012           X2=A(IF3+10)
0013           Y1=A(IF2+11)
0014           Y2=A(IF3+11)
0015           IPOINT=IA(I2+12)
0016           DO 10 I=1,NC1
0017           XDIF=X1-A(IPOINT)
0018           YDIF=Y1-A(IPOINT+1)
0019           Z2=XDIF*XDIF+YDIF*YDIF
0020           XDIF=X2-A(IPOINT)
0021           YDIF=Y2-A(IPOINT+1)
0022           Z2=AMIN1(Z2,XDIF*XDIF+YDIF*YDIF)
0023           IF(Z2.LT.4.0) RETURN
0024        10 IPOINT=IA(IPOINT+2)
               C
0025           WRITE(6,2001) NB1,X1,Y1,X2,Y2
               C
               C
0026           NBLOKS=NBLOKS+1
0027           IF(NBLOKS.LT.M2) GO TO 20
0028           WRITE(6,3000)
0029           EFLAG=.TRUE.
0030           CALL FINISH
               C
               C------------------- EDGES INVOLVED IN CRACK ---
0031        20 NP1=IA(IF2+1)
0032           NP2=IA(IF3+1)
               C------------------- IS NP2 GREATER THAN NP1 ? ---
0033           IF(NP2.GT.NP1) GO TO 50
               C------------------- NO, REARRANGE ---
0034           NTEMP=NP1
0035           NP1=NP2
0036           NP2=NTEMP
0037           NTEMP=IF3
0038           IF3=IF2
```

```
FORTRAN IV-PLUS V02-04G          14:37:31     29-MAR-78            PAGE 23
RBMC.FTN          /I4/TR:BLOCKS/WR

0039            IF2=NTEMP
0040          50 CONTINUE
              C------------------- YES ---
0041            IEND1=IA(I2+12)
0042            DO 100 I=1,NC1
0043            IF(NP1.EQ.1) IC2=IEND1
0044            IF(NP2.EQ.1) IC3=IEND1
0045            IF(NP1.LE.1.AND.NP2.LE.1) GO TO 110
0046        100 IEND1=IA(IEND1+2)
              C------------------- CREATE MEMORY FOR NEW BLOCK DATA ---
0047        110 NBN=NBLOKS
0048            CALL LIMIT(NVARB+12)
0049            IA(NBN)=NEMPTG
0050            NEMPTG=NEMPTG+NVARB
              C------------------- DETERMINE CORNER COORDINATES OF NEW BLOCKS ---
0051            IC22=IA(IC2+2)
0052            IC33=IA(IC3+2)
              C------------------- FOR BLOCK NB1 ---
0053            IA(IC2+2)=NEMPTG
0054            A(NEMPTG)=A(IF2+10)
0055            A(NEMPTG+1)=A(IF2+11)
0056            IA(NEMPTG+2)=NEMPTG+3
0057            A(NEMPTG+3)=A(IF3+10)
0058            A(NEMPTG+4)=A(IF3+11)
0059            IA(NEMPTG+5)=IA(IC3+2)
0060            NEMPTG=NEMPTG+6
              C------------------- FOR BLOCK NBN ---
0061            IA(IC3+2)=NEMPTG
0062            A(NEMPTG)=A(IF3+10)
0063            A(NEMPTG+1)=A(IF3+11)
0064            IA(NEMPTG+2)=NEMPTG+3
0065            A(NEMPTG+3)=A(IF2+10)
0066            A(NEMPTG+4)=A(IF2+11)
0067            IA(NEMPTG+5)=IC22
0068            NEMPTG=NEMPTG+6
              C------------------- CREATE NEW BLOCK DATA---
0069            NP=0
0070            I21=IA(NB1)
0071        115 NP=NP+1
0072            NC=0
0073            IEND1=IA(I2+12)
              C------------------- NUMBER OF CORNERS ? ---
0074        120 NC=NC+1
0075            IEND1=IA(IEND1+2)
0076            IF(IEND1.EQ.IA(I2+12)) GO TO 125
0077            GO TO 120
0078        125 IA(I2)=0
0079            IA(I2+1)=NC
              C------------------- AREA AND CENTROID  ---
0080            AREA=0.0
0081            XC=0.0
0082            YC=0.0
0083            IEND1=IA(I2+12)
0084            DO 130 I=1,NC
0085            IEND2=IA(IEND1+2)
0086            AREA=AREA+(A(IEND2)-A(IEND1))*(A(IEND2+1)+A(IEND1+1))
```

```
FORTRAN IV-PLUS V02-04G          14:37:31     29-MAR-78           PAGE 24
KBMC.FTN          /I4/TR:BLOCKS/WR

0087           YC=YC+(A(IEND2)-A(IEND1))*((A(IEND2+1)-A(IEND1+1))*(A(IEND2+1)
              .     +2.0*A(IEND1+1))+3.0*A(IEND1+1)**2)
0088           XC=XC+(A(IEND2+1)-A(IEND1+1))*((A(IEND2)-A(IEND1))*(A(IEND2)
              .     +2.0*A(IEND1))+3.0*A(IEND1)**2)
0089           IEND1=IEND2
0090       130 CONTINUE
0091           AREA=0.5*AREA
0092       135 YC=YC/(6.0*AREA)
0093           XC=-XC/(6.0*AREA)
0094           YC=(YC-YL)*SFACT
0095           XC=(XC-XL)*SFACT
0096           AREA=AREA*SFACT*SFACT
0097           A(I2+2)=XC
0098           A(I2+3)=YC
0099           A(I2+7)=AREA*RHO
0100           A(I2+4)=0.0
0101           A(I2+5)=0.0
0102           A(I2+6)=0.0
0103           A(I2+9)=0.0
0104           A(I2+10)=0.0
0105           A(I2+11)=0.0
               C------------------- LOCAL COORDINATES ---
0106           IEND1=IA(I2+12)
0107           DO 140 I=1,NC
0108           A(IEND1)=(A(IEND1)-XL)*SFACT-XC
0109           A(IEND1+1)=(A(IEND1+1)-YL)*SFACT-YC
0110       140 IEND1=IA(IEND1+2)
               C------------------- MOMENT OF INERTIA ---
0111           RMOI=0.0
0112           IEND1=IA(I2+12)
0113           DO 150 I=1,NC
               C
0114           IEND2=IA(IEND1+2)
0115           AREA=A(IEND1)*A(IEND1+1) + (A(IEND2)-A(IEND1))*(A(IEND2+1)
              .     +A(IEND1+1))-A(IEND2)*A(IEND2+1)
0116           AREA=0.5*AREA
0117           TEMP=A(IEND1)**2+A(IEND1+1)**2+A(IEND2)**2+A(IEND2+1)**2
              .     +A(IEND1)*A(IEND2)+A(IEND1+1)*A(IEND2+1)
0118           RMOI=RMOI+AREA*TEMP/6.0
0119       150 IEND1=IEND2
0120           A(I2+8)=RMOI*RHO
               C
               C------------------- GLOBAL COORDINATES ---
0121           IEND1=IA(I2+12)
0122           DO 160 I=1,NC
0123           A(IEND1)=A(IEND1)+A(I2+2)
0124           A(IEND1+1)=A(IEND1+1)+A(I2+3)
0125       160 IEND1=IA(IEND1+2)
               C
0126           IF(NP.EQ.2) GO TO 180
0127           I2=IA(NBR)
0128           IA(I2+12)=IC22
0129           GO TO 115
               C------------------- DELETION OF CONTACTS CAUSING THE CRACK ---
0130       180 I2I=IA(NBI)
0131           JTEMP=I2I+13
```

```
FORTRAN IV-PLUS V02-04G          14:37:31     29-MAR-78           PAGE 25
RBMC.FTN          /14/TR:BLOCKS/WR

0132            JC1=IA(JTEMP)
0133         190 IF(JC1.EQ.0) GO TO 200
0134            IC1=IA(JC1)
0135            IF(IC1.NE.IF2.AND.IC1.NE.IF3) GOTO 195
0136            CALL DELCON(JTEMP)
0137            GOTO 198
0138         195 JTEMP=JC1+1
0139         198 JC1=IA(JTEMP)
0140            GOTO 190
          C------------------ REARRANGEMENT OF CONTACT DATA ---
0141         200 I2N=IA(NBN)
0142            XT1=(A(IF2+10)-A(IC2))*A(IF2+9)
                .   +(A(IF2+11)-A(IC2+1))*A(IF2+8)
0143            XT2=(A(IF3+10)-A(IC3))*A(IF3+9)
                .   +(A(IF3+11)-A(IC3+1))*A(IF3+8)
          C
0144            JC1=IA(I21+13)
0145            JTEMP=I21+13
0146         210 IF(JC1.EQ.0) GO TO 700
0147            IC1=IA(JC1)
0148            IF(IA(IC1+3).EQ.NB1) GO TO 220
0149            IF(IA(IC1+1).LT.NP1) GO TO 600
0150            IF(IA(IC1+1).GT.NP2) GO TO 300
          C
0151            GO TO 350
          C
0152         220 IF(IA(IC1+2).LE.NP1) GO TO 600
0153            IF(IA(IC1+2).LE.NP2) GO TO 250
          C
0154            NP3=NP2+1
0155            NP4=NP1+3
0156         230 IF(IA(IC1+2).NE.NP3) GO TO 240
0157            IA(IC1+2)=NP4
0158            GO TO 600
0159         240 NP3=NP3+1
0160            NP4=NP4+1
0161            GO TO 230
          C
0162         250 NP3=NP1+1
0163            NP4=1
0164         260 IF(IA(IC1+2).NE.NP3) GO TO 270
0165            IA(IC1+3)=NBN
0166            IA(IC1+2)=NP4
0167            IA(JTEMP)=IA(JC1+1)
0168            IA(JC1+1)=IA(I2N+13)
0169            IA(I2N+13)=JC1
0170            GO TO 600
0171         270 NP3=NP3+1
0172            NP4=NP4+1
0173            GO TO 260
          C
0174         300 NP3=NP2+1
0175            NP4=NP1+3
0176         310 IF(IA(IC1+1).NE.NP3) GO TO 320
0177            IA(IC1+1)=NP4
0178            GO TO 600
```

```
FORTRAN IV-PLUS V02-04G          14:37:31     29-MAR-78            PAGE 26
RBMC.FTN          /I4/TR:BLOCKS/WR

0179       320 NP3=NP3+1
0180           NP4=NP4+1
0181           GO TO 310
           C
0182       350 IF(IA(IC1+1).EQ.NP1) GO TO 380
0183           IF(IA(IC1+1).EQ.NP2) GO TO 400
0184           GO TO 500
0185       380 XT=(A(IC1+10)-A(IC2))*A(IC1+9)
              .  +(A(IC1+11)-A(IC2+1))*A(IC1+8)
0186           IF(XT.LT.XT1) GO TO 600
0187           IA(IC1+1)=IA(I2N+1)
0188           IA(JTEMP)=IA(JC1+1)
0189           IA(JC1+1)=IA(I2N+13)
0190           IA(I2N+13)=JC1
0191           GO TO 600
           C
0192       400 XT=(A(IC1+10)-A(IC3))*A(IC1+9)
              .  +(A(IC1+11)-A(IC3+1))*A(IC1+8)
0193           IF(XT.GT.XT2) GO TO 410
0194           IA(JTEMP)=IA(JC1+1)
0195           IA(IC1+1)=IA(I2N+1)-2
0196           IA(JC1+1)=IA(I2N+13)
0197           IA(I2N+13)=JC1
0198           GO TO 420
0199       410 IA(IC1+1)=NP1+2
0200       420 GO TO 600
           C
0201       500 NP3=NP1+1
0202           NP4=1
0203       510 IF(IA(IC1+1).NE.NP3) GO TO 520
0204           IA(IC1+1)=NP4
0205           IA(JTEMP)=IA(JC1+1)
0206           IA(JC1+1)=IA(I2N+13)
0207           IA(I2N+13)=JC1
0208           GO TO 600
0209       520 NP3=NP3+1
0210           NP4=NP4+1
0211           GO TO 510
0212       600 JTEMP=JC1+1
0213           JC1=IA(JC1+1)
0214           GO TO 210
           C------------------- BOXING OF NEW BLOCKS ---
0215       700 NP=0
0216           TEMP=TDEL
0217       710 NP=NP+1
0218           I2=IA(NB1)
0219           CALL BOXC
           C------------------- CHECK TIMESTEP ---
0220           TN=2.0*(SQRT(A(I2+7)/STIFN))*TFRAC
0221           TS=2.0*(SQRT(A(I2+7)/STIFS))*TFRAC
0222           TDEL=AMIN1(TDEL,TN,TS)
0223           IF(NP.GT.1) GO TO 720
0224           NTEMP=NB1
0225           NB1=NBN
0226           GO TO 710
0227       720 NB1=NTEMP
```

```
FORTRAN IV-PLUS V02-04G          14:37:31     29-MAR-78            PAGE 27
RBMC.FTN         /14/IR:BLOCKS/WR

0228            IF(TDEL.EQ.TEMP) GO TO 750
0229            BDT=BDT*TEMP/TDEL
0230            CON1=1.0-ALPHA*TDEL/2.0
0231            CON2=1.0/(1.0+ALPHA*TDEL/2.0)
0232            WRITE(6,2000) TDEL
0233          750 RETURN
0234         2000 FORMAT(X/,30X,20HNEW TIME INCREMENT =,1PE12.4)
0235         2001 FORMAT(30X,5HBLOCK,I5,24H HAS CRACKED AT LOCATION,
                .       2E11.4,2X,2E11.4)
0236         3000 FORMAT(46H !!! ERROR : MAXIMUM NUMBER OF BLOCKS EXCEEDED)
0237            END


PROGRAM SECTIONS

 NAME        SIZE                  ATTRIBUTES

$CODE1   010164   2106              RW,I,CON,LCL
$IDATA   000232     77              RW,D,CON,LCL
$VARS    000234     78              RW,D,CON,LCL
$TEMPS   000010      4              RW,D,CON,LCL
.$$$$.   027340   6000              RW,D,OVR,GBL
CBLOCK   000424    138              RW,D,OVR,GBL


TOTAL SPACE ALLOCATED = 040646  8403
```

```
FORTRAN IV-PLUS V02-04G          14:39:29     29-MAR-78           PAGE 28
RBMC.FTN         /14/TR:BLOCKS/WR

0001           SUBROUTINE DUMPC
               C
               C------------------- ROUTINE TO PRINT MEMORY ALLOCATION AND CONTENTS -
               C
0002           INCLUDE 'COMMON.FTN'
0003  *        COMMON A(3000)
0004  *        DIMENSION IA(1)
0005  *        EQUIVALENCE (A,IA)
0006           INCLUDE 'CBLOCK.FTN'
0007  *        COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,
      *       .                NBLOKS,NCYC,MCYCLE,NEMPT,RHO,EFLAG,TFRAC,
      *       .                TDEL,IBOXES,JBOXES,XL,XU,YL,YU,BSIZE,SFACT,XSIZE,
      *       .                YSIZE,UDMAX,UMOST,STIFN,STIFS,FRIC,BETA,BDT,TMAX,
      *       .                CON1,CON2,ALPHA,GRAVX,GRAVY,LOC1,LOC2,NUPDAT,NB1,
      *       .                NVARB,NFRAG,NPR,NEMPTC,NEMPTD,NEMPTG,IEND1,LBLOCK
0008  *        LOGICAL EFLAG
               C
0009           IF(LOC2.EQ.0) RETURN
0010           WRITE(6,2000)
0011           WRITE(6,2005) M1,M2,M3,NEMPTG,NEMPTC,NEMPTD,NBLOKS,
              .              NCYC,NEMPT,MCYCLE
0012           IF(LOC2.LT.0) GOTO 100
               C------------------- DUMP CONSECUTIVE MEMORY LOCATIONS ---
0013           WRITE(6,2009)
0014           I2=((LOC1-1)/10)*10
0015        10 I1=I2+1
0016           I2=I1+9
0017           WRITE(6,2001) (IA(I),I=I1,I2),I2
0018           IF(I2.LT.LOC2) GOTO 10
0019           GOTO 500
               C
               C------------------- DUMP BY BOX ---
0020       100 WRITE(6,2002)
0021           DO 120 NBOX=1,NBOXES
0022           I3=IA(M2+NBOX-1)
               C------------------- ANY MORE ENTRIES ? ---
0023       110 IF(I3.EQ.0) GOTO 120
               C------------------- NO, PRINT BOX,BLOCK ---
0024           WRITE(6,2008) NBOX,IA(I3)
0025           I3=IA(I3+1)
0026           GOTO 110
0027       120 CONTINUE
               C------------------- CONTENTS OF EACH BLOCK ---
0028           WRITE(6,2006)
0029           DO 130 NB=1,NBLOKS
0030           IF=IA(NB)
0031           NC=IA(IF+1)
0032           IL=IA(IF+12)
0033           CALL LIMIT(2*NC)
0034           DO 125 IC=NEMPTG,NEMPTG+2*NC,2
0035           A(IC)=A(IL)
0036           A(IC+1)=A(IL+1)
0037       125 IL=IA(IL+2)
0038           WRITE(6,2003) NB,IA(IF),IA(IF+1),(A(I),I=IF+2,IF+11),
              .      (A(J),J=NEMPTG,NEMPTG+2*NC-1)
0039       130 CONTINUE
```

```
FORTRAN IV-PLUS V02-04G          14:39:29     29-MAR-78           PAGE 29
RBMC.FTN         /I4/TR:BLOCKS/WR

      C------------------- DUMP CONTACT DATA ---
0040          WRITE(6,2007)
0041          DO 150 NB=1,NBLOKS
0042          IF=IA(NB)
0043          JC1=IA(IF+13)
0044        135 IF(JC1.EQ.0) GO TO 150
0045          IC1=IA(JC1)
0046          IF(NB.EQ.IA(IC1+3)) GO TO 140
0047          WRITE(6,2004) NB,(IA(I),I=IC1,IC1+3),
     .                       (A(J),J=IC1+4,IC1+11)
0048        140 JC1=IA(JC1+1)
0049          GO TO 135
0050        150 CONTINUE
      C------------------
0051        500 WRITE(6,2000)
      C
0052          RETURN
      C
0053       2000 FORMAT(1X,130(1H-))
0054       2001 FORMAT(1X,10I12,16)
0055       2002 FORMAT(10X,3HBOX,7X,5HBLOCK)
0056       2003 FORMAT(/1X,3I3,1P10E10.2/(10X,1P10E10.2))
0057       2004 FORMAT(/1X,5I10/1X,1P8E10.2)
0058       2005 FORMAT(9X,2HM1,8X,2HM2,8X,2HM3,4X,6HNEMPTG,4X,6HNEMPTC,
     .          4X,6HNEMPTD,4X,6HNBLOKS,6X,4HNCYC,5X,5HNEMPT,4X,6HMCYCLE/
     .          1X,11I10)
0059       2006 FORMAT(11H BLOCK DATA,10(1H-)/1X,9H NB IF NC,8X,2HXC,8X,2HYC,
     .          6X,4HXDOT,6X,4HYDOT,6X,4HTDOT,6X,4HMASS,3X,7HINERTIA,
     .          5X,5HXFSUM,5X,5HYFSUM,6X,4HMSUM)
0060       2007 FORMAT(13H CONTACT DATA,10(1H-)/8X,3HNBE,7X,3HPRE,7X,3HNPE,7X,
     .          3HNPC,7X,3HNBC/10X,1HS,9X,1HN,8X,2HFN,8X,2HFS,
     .          7X,3HSIN,7X,3HCOS,7X,3HXCP,7X,3HYCP)
0061       2008 FORMAT(1X,3I12)
0062       2009 FORMAT(1X,30(1H*),19HVALUES ARE INTEGERS,30(1H*))
      C
0063          END


PROGRAM SECTIONS

 NAME        SIZE                    ATTRIBUTES

$CODE1   002714    742               RW,I,CON,LCL
$PDATA   000004      2               RW,D,CON,LCL
$IDATA   000660    216               RW,D,CON,LCL
$VARS    000064     26               RW,D,CON,LCL
$TEMPS   000022      9               RW,D,CON,LCL
.$$$$.   027340   6000               RW,D,OVR,GBL
CBLOCK   000424    138               RW,D,OVR,GBL


TOTAL SPACE ALLOCATED = 033672  7133
```

```
FORTRAN IV-PLUS V02-04G          14:40:19     29-MAR-78           PAGE 30
RBMC.FTN         /I4/TR:BLOCKS/WR

0001           SUBROUTINE DETECT
               C
               C------------------- ROUTINE TO DETECT AND UPDATE CONTACTS ---
               C
0002           INCLUDE 'COMMON.FTN'
0003 *         COMMON A(3000)
0004 *         DIMENSION IA(1)
0005 *         EQUIVALENCE (A,IA)
0006           INCLUDE 'CBLOCK.FTN'
0007 *         COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,
     *        .                NBLOKS,NCYC,MCYCLE,NEMPT,RHO,EFLAG,TFRAC,
     *        .                TDEL,IBOXES,JBOXES,XL,XU,YL,YU,BSIZE,SFACT,XSIZE,
     *        .                YSIZE,UDMAX,UMOST,STIFN,STIFS,FRIC,BETA,BDT,TMAX,
     *        .                CON1,CON2,ALPHA,GRAVX,GRAVY,LOC1,LOC2,NUPDAT,NB1,
     *        .                NVARB,NFRAG,NPR,NEMPTC,NEMPTD,NEMPTG,IEND1,LBLOCK
0008 *         LOGICAL EFLAG
0009           LOGICAL FIRST
0010           DATA TOL/1.0/
               C
0011           I21=IA(NB1)
0012           NC1=IA(I21+1)
               C
               C------------------- CORNER COORDINATES ---
0013           X1=A(IEND1)
0014           Y1=A(IEND1+1)
0015           IEND2=IA(IEND1+2)
0016           X2=A(IEND2)
0017           Y2=A(IEND2+1)
0018           IEND3=IA(IEND2+2)
0019           NP=0
               C------------------- START DETECTION ---
0020         100 NP=NP+1
0021           FIRST=.TRUE.
               C------------------- EDGE NUMBER ---
0022           NP1=NPR+NP-2
0023           IF(NP1.GT.NC1) NP1=NP1-NC1
               C
               C------------------- DETERMINE LIMITS OF SCAN ---
0024         200 XX1=AMIN1(X1,X2)
0025           XX2=AMAX1(X1,X2)
0026           YY1=AMIN1(Y1,Y2)
0027           YY2=AMAX1(Y1,Y2)
               C------------------- DETERMINE BOXES TO BE SCANNED ---
0028           NXL=IFIX((XX1-TOL)/BSIZE)+1
0029           NXU=MIN0(IFIX((XX2+TOL)/BSIZE)+1,IBOXES)
0030           NYL=IFIX((YY1-TOL)/BSIZE)+1
0031           NYU=MIN0(IFIX((YY2+TOL)/BSIZE)+1,JBOXES)
               C------------------- SEARCH BOXES ---
0032           DO 350 JBOX=NYL,NYU
0033           NBOX1=(JBOX-1)*IBOXES+NXL-1
               C
0034           DO 300 IBOX=NXL,NXU
0035           NBOX1=NBOX1+1
               C------------------- SET BOX POINTER ---
0036           I3=M2+NBOX1-1
               C------------------- IS BOX EMPTY ?---
```

```
FORTRAN IV-PLUS V02-04G          14:40:19     29-MAR-78            PAGE 31
RBMC.FTN         /I4/TR:BLOCKS/WR

0037           IF(IA(I3).EQ.0) GO TO 300
0038           I31=IA(I3)
       C------------------- IS ENTRY FOR ANOTHER BLOCK ?---
0039         210 IF(IA(I31).NE.NB1) GO TO 240
       C------------------- NO,ANY MORE ENTRIES ?---
0040         215 IF(IA(I31+1).EQ.0) GO TO 300
       C------------------- YES ---
0041           I31=IA(I31+1)
0042           GO TO 210
       C
       C------------------ SET UP DATA FOR SECOND BLOCK ---
0043         240 NB2=IA(I31)
0044           I22=IA(NB2)
0045           NC2=IA(I22+1)
0046           IC2=IA(I22+12)
0047           IC=IC2
0048           NC=0
       C------------------- CHECK POSSIBLE CONTACT ---
0049         250 NC=NC+1
0050           IF(NC.GT.NC2) GO TO 215
       C------------------- EDGE TO CORNER CONTACT ?---
0051           IF(NP.GT.2) GO TO 251
       C------------------- NO, SCAN EDGES OF SECOND BLOCK ---
       C------------------- YES, SCAN CORNERS OF SECOND BLOCK ---
0052           NP2=NC+1
0053           IF(NP2.GT.NC2) NP2=NP2-NC2
0054           ICL=IC2
0055           IC2=IA(ICL+2)
0056           ICR=IA(IC2+2)
0057           X=A(IC2)
0058           Y=A(IC2+1)
0059           IBOX2=MIN0(IFIX(X/BSIZE)+1,IBOXES)
0060           JBOX2=MIN0(IFIX(Y/BSIZE),JBOXES-1)
0061           NBOX2=JBOX2*IBOXES+IBOX2
       C------------------- IS CORNER IN SAME BOX AS EDGE ?
0062           IF(NBOX1.EQ.NBOX2) GO TO 255
       C------------------- YES, CHECK CONTACT CONDITION ---
       C------------------- NO, GET NEXT CORNER ---
0063           GO TO 250
       C------------------- CORNER TO EDGE CONTACT ---
0064         251 FIRST=.TRUE.
0065           NP1=NC
0066           NP2=NPR
0067           IF(NP2.GT.NC1) NP2=NP2-NC1
0068           NB2=NB1
0069           X1=A(IC)
0070           Y1=A(IC+1)
0071           IC=IA(IC+2)
0072           X2=A(IC)
0073           Y2=A(IC+1)
0074           IC2=IEND2
       C------------------- NORMAL EDGE TO CORNER CONTACT TEST ---
0075         255 IF(.NOT.FIRST) GO TO 260
0076           XDIF=X2-X1
0077           YDIF=Y2-Y1
0078           Z=SQRT(XDIF**2+YDIF**2)
```

```
FORTRAN IV-PLUS V02-04G          14:40:19     29-MAR-78            PAGE 32
RBMC.FTN         /14/TR:BLOCKS/WR

0079           SINA=YDIF/Z
0080           COSA=XDIF/Z
0081           FIRST=.FALSE.
       C
0082         260 YT= (A(IC2+1)-Y1)*COSA
              .    -(A(IC2)-X1)*SINA
0083           IF(YT.GT.2.0) GO TO 250
0084           IF(YT.LT.-3.0) GO TO 250
       C
0085           XT= (A(IC2)-X1)*COSA
              .    +(A(IC2+1)-Y1)*SINA
0086           IF(XT.GT.Z+2.0) GO TO 250
0087           IF(XT.LT.-2.0) GO TO 250
0088           IPFLG=1
0089           IF(YT.GT.1.0.OR.XT.GT.Z.OR.XT.LT.0.0) IPFLG=0
       C------------------- IS CONTACT ALREADY DETECTED ? ---
0090           JC1=IA(I21+13)
       C--------------------- CHECK CONTACT LIST FOR THIS BLOCK ---
0091         265 IF(JC1.EQ.0) GO TO 280
0092           IC1=IA(JC1)
       C------------------- CHECK EACH STORED CONTACT ---
0093           IF(IA(IC1+3).EQ.NB2.AND.IA(IC1+2).EQ.NP2.
              .    AND.IA(IC1+1).EQ.NP1) GO TO 270
0094           JC1=IA(JC1+1)
0095           GO TO 265
       C
0096         270 CONTINUE
       C
0097           IF(YT.GT.-2.0) GO TO 275
0098           WRITE(6,3000)
       C
       C------------------- UPDATE EXISTING CONTACT DATA ---
0099         275 A(IC1+8)=SINA
0100           A(IC1+9)=COSA
0101           A(IC1+10)=A(IC2)
0102           A(IC1+11)=A(IC2+1)
0103           IA(IC1)=IPFLG
       C
0104           IF(NP.GT.2) GO TO 400
0105           GO TO 250
       C
0106         280 CONTINUE
       C
0107           IF(IPFLG.EQ.0) GOTO 250
0108           IF(YT.GT.0.0) GO TO 250
       C
0109           YTR=(A(ICR+1)-Y1)*COSA-(A(ICR)-X1)*SINA
0110           IF(YTR.LE.-2.0) GO TO 250
       C
0111           YTL=(A(ICL+1)-Y1)*COSA-(A(ICL)-X1)*SINA
0112           IF(YTL.LE.-2.0) GO TO 250
       C
       C------------------- LOCATE EMPTY CONTACT DATA SPACE ---
0113           CALL EMPTYC
0114           IC1=NEMPT
       C------------------- LOCATE EMPTY CONTACT LIST 'DOUBLES' ---
```

```
FORTRAN IV-PLUS V02-04G          14:40:19     29-MAR-78            PAGE 33
RBMC.FTN         /I4/TR:BLOCKS/WR

0115            CALL EMPTYD
              C------------------- ENTER CONTACT INTO THE CONTACT LIST  ---
0116            IA(NEMPT+1)=IA(I21+13)
0117            IA(I21+13)=NEMPT
0118            IA(NEMPT)=IC1
              C------------------- REPEAT FOR SECOND BLOCK ---
0119            CALL EMPTYD
0120            IA(NEMPT+1)=IA(I22+13)
0121            IA(I22+13)=NEMPT
0122            IA(NEMPT)=IC1
              C------------------- NEW CONTACT DATA ---
0123            A(IC1+4)=0.0
0124            A(IC1+5)=0.0
0125            A(IC1+6)=0.0
0126            A(IC1+7)=0.0
0127            IA(IC1+1)=NP1
0128            IA(IC1+2)=NP2
0129            IA(IC1+3)=NB2
0130            GO TO 275
0131          300 CONTINUE
0132          350 CONTINUE
              C------------------- END SCAN OF BOXES ---
0133            IF(NP.GT.1) GO TO 400
0134            X1=X2
0135            Y1=Y2
0136            X2=A(IEND3)
0137            Y2=A(IEND3+1)
0138            GO TO 100
              C------------------- CHECK ON CORNER TO EDGE CONTACT ? ---
0139          400 NP=NP+1
0140            IF(NP.GT.3) GO TO 500
              C------------------- NO , END DETECTION ---
              C------------------- YES, RETURN TO SCANNING LOGIC ---
0141            X1=A(IEND2)
0142            Y1=A(IEND2+1)
0143            X2=X1
0144            Y2=Y1
0145            ICR=IEND1
0146            ICL=IEND3
0147            GO TO 200
              C
              C------------------- END DETECTION ---
0148          500 NUPDAT=NUPDAT+1
0149            UMOST=0.0
              C------------------- SCAN CONTACTS AGAIN TO
              C                    DELETE CONTACTS NOT
              C                    FLAGGED FOR PRESERVATION ---
0150            DO 650 NBB=1,NBLOKS
0151            IBE=IA(NBB)
0152          610 JC1=IA(IBE+13)
0153            JTEMP=IBE+13
0154          620 IF(JC1.EQ.0) GOTO 650
0155            IC1=IA(JC1)
0156            IF(NBB.EQ.IA(IC1+3)) GOTO 630
0157            IF(IA(IC1).NE.0) GOTO 630
0158            CALL DELCON(JTEMP)
```

```
FORTRAN IV-PLUS V02-04G          14:40:19     29-MAR-78          PAGE 34
RBMC.FTN          /I4/TR:BLOCKS/WR

0159            GOTO 610
0160         630 JTEMP=JC1+1
0161            JC1=IA(JTEMP)
0162            GOTO 620
0163         650 CONTINUE
             C
0164        3000 FORMAT(30X,26H DRIFT CORRECTION REQUIRED)
0165            END


PROGRAM SECTIONS

 NAME        SIZE                    ATTRIBUTES

$CODE1   005250   1364               RW,I,CON,LCL
$IDATA   000112     37               RW,D,CON,LCL
$VARS    000330    108               RW,D,CON,LCL
$TEMPS   000014      6               RW,D,CON,LCL
.$$$$.   027340   6000               RW,D,OVR,GBL
CBLOCK   000424    138               RW,D,OVR,GBL


TOTAL SPACE ALLOCATED = 035712  7653
```

```
FORTRAN IV-PLUS V02-04G          14:41:37     29-MAR-78            PAGE 35
RBMC.FTN          /I4/TR:BLOCKS/WK

        C
0001            SUBROUTINE DELENT
        C
        C------------------- DELETION OF OBSOLETE BOX ENTRIES ---
        C
0002            INCLUDE 'COMMON.FTN'
0003 *          COMMON A(3000)
0004 *          DIMENSION IA(1)
0005 *          EQUIVALENCE (A,IA)
0006            INCLUDE 'CBLOCK.FTN'
0007 *          COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,
     *         .                NBLOKS,NCYC,MCYCLE,NEMPT,RHO,EFLAG,TFRAC,
     *         .                TDEL,IBOXES,JBOXES,XL,XU,YL,YU,BSIZE,SFACT,XSIZE,
     *         .                YSIZE,UDMAX,UMOST,STIFN,STIFS,FRIC,BETA,BDT,TMAX,
     *         .                CON1,CON2,ALPHA,GRAVX,GRAVY,LOC1,LOC2,NUPDAT,NB1,
     *         .                NVARB,NFRAG,NPR,NEMPTC,NEMPTD,NEMPTG,IEND1,LBLOCK
0008 *          LOGICAL EFLAG
0009            DATA TOL/5.0/
        C
0010            I2=IA(NB1)
0011            NC=IA(I2+1)
0012            NP=0
        C------------------- DETERMINE LIMITS OF BLOCK ---
0013            XX1=XU
0014            XX2=XL
0015            YY1=YU
0016            YY2=YL
0017            IEND2=IA(I2+12)
0018          100 NP=NP+1
0019            X1=A(IEND2)
0020            Y1=A(IEND2+1)
0021            IEND2=IA(IEND2+2)
0022            XX1=AMIN1(XX1,X1)
0023            XX2=AMAX1(XX2,X1)
0024            YY1=AMIN1(YY1,Y1)
0025            YY2=AMAX1(YY2,Y1)
0026            IF(NP.EQ.NC) GO TO 150
0027            GO TO 100
        C------------------- DETERMINE BOXES TO BE SEARCHED ---
0028          150 NXL=MAX0(IFIX((XX1-TOL)/BSIZE),1)
0029            NXU=MIN0(IFIX((XX2+TOL)/BSIZE)+2,IBOXES)
0030            NYL=MAX0(IFIX((YY1-TOL)/BSIZE),1)
0031            NYU=MIN0(IFIX((YY2+TOL)/BSIZE)+2,JBOXES)
        C------------------ SEARCH BOXES ---
0032            DO 500 JBOX=NYL,NYU
0033            NBOX1=(JBOX-1)*IBOXES+NXL-1
        C
0034            DO 400 IBOX=NXL,NXU
0035            NBOX1=NBOX1+1
        C
        C------------------- SET BOX POINTER ---
0036            I3=M2+NBOX1-1
0037            JTEMP=I3
0038            I3=IA(I3)
        C------------------- NO ENTRIES OR END OF LIST ?---
0039          200 IF(I3.EQ.0) GO TO 400
```

```
FORTRAN IV-PLUS V02-04G          14:41:37     29-MAR-78           PAGE 36
RBMC.FTN         /I4/TR:BLOCKS/WR

        C------------------- NO,IS ENTRY FOR THE BLOCK CONCERNED ?---
0040            IF(ABS(IA(I3)).NE.NB1) GO TO 250
        C------------------- YES,IS ENTRY VALID ?---
        C------------------- NO,DELETE ---
0041            IF(IA(I3).EQ.NB1) GO TO 300
        C------------------- YES,PRESERVE ---
0042            IF(IA(I3).EQ.-NB1) IA(I3)=-IA(I3)
        C------------------- GET NEXT BOX ENTRY ---
0043          250 JTEMP=I3+1
0044            I3=IA(I3+1)
0045            GO TO 200
        C------------------- DELETION OF OBSOLETE ENTRY ---
0046          300 IA(JTEMP)=IA(I3+1)
        C------------------- RECOVERY OF EMPTY 'DOUBLE' ---
0047            IF(NEMPTD.NE.0) GO TO 350
0048            NEMPTD=I3
0049            IA(NEMPTD)=0
0050            GO TO 400
0051          350 IA(I3)=NEMPTD
0052            NEMPTD=I3
0053          400 CONTINUE
0054          500 CONTINUE
        C------------------- END OF BOX SEARCH ---
0055            RETURN
0056            END


PROGRAM SECTIONS

 NAME        SIZE                  ATTRIBUTES

$CODE1  001726    491               RW,I,CON,LCL
$PDATA  000004      2               RW,D,CON,LCL
$IDATA  000060     24               RW,D,CON,LCL
$VARS   000120     40               RW,D,CON,LCL
$TEMPS  000010      4               RW,D,CON,LCL
.$$$$.  027340   6000               RW,D,OVR,GBL
CBLOCK  000424    138               RW,D,OVR,GBL


TOTAL SPACE ALLOCATED = 032126  6699
```

```
FORTRAN IV-PLUS V02-04G          14:42:00     29-MAR-78            PAGE 37
RBMC.FTN         /I4/TR:BLOCKS/WR

0001            SUBROUTINE DELCON(JTEMP)
                C
                C------------------ DELETION OF OBSOLETE CONTACTS ---
                C
0002            INCLUDE 'COMMON.FTN'
0003 *          COMMON A(3000)
0004 *          DIMENSION IA(1)
0005 *          EQUIVALENCE (A,IA)
0006            INCLUDE 'CBLOCK.FTN'
0007 *          COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,
     *         .                NBLOKS,NCYC,MCYCLE,NEMPT,RHO,EFLAG,TFRAC,
     *         .                TDEL,IBOXES,JBOXES,XL,XU,YL,YU,BSIZE,SFACT,XSIZE,
     *         .                YSIZE,UDMAX,UMOST,STIFN,STIFS,FRIC,BETA,BDT,TMAX,
     *         .                CON1,CON2,ALPHA,GRAVX,GRAVY,LOC1,LOC2,NUPDAT,NB1,
     *         .                NVARB,NFRAG,NPR,NEMPTC,NEMPTD,NEMPTG,IEND1,LBLOCK
0008 *          LOGICAL EFLAG
                C
                C------------------- POINTER TO OBSOLETE DOUBLE ---
0009            JC1=IA(JTEMP)
                C------------------- POINTER TO OBSOLETE CONTACT DATA ---
0010            IC1=IA(JC1)
                C------------------- REMOVE FROM CONTACT LIST ---
0011            IA(JTEMP)=IA(JC1+1)
                C------------------- RECOVER EMPTY 'DOUBLE' ---
0012            IA(JC1)=NEMPTD
0013            NEMPTD=JC1
                C------------------- LOCATE CONTACT POINTER FROM SECOND BLOCK ---
0014           30 NB2=IA(IC1+3)
0015            I22=IA(NB2)
0016            JTEMP2=I22+13
0017            JC2=IA(JTEMP2)
                C--------------------IS THIS THE CORRECT POINTER ?---
0018           40 IF(IA(JC2).NE.IC1) GO TO 50
                C------------------- YES,DELETE ENTRY,RECOVER EMPTY 'DOUBLE' ---
0019            IA(JTEMP2)=IA(JC2+1)
0020            IA(JC2)=NEMPTD
0021            NEMPTD=JC2
0022            GO TO 60
                C------------------- NO,GET NEXT CONTACT POINTER ---
0023           50 JTEMP2=JC2+1
0024            JC2=IA(JTEMP2)
0025            GO TO 40
                C------------------ DELETION OF CONTACT DATA, RECOVERY OF SPACE ---
0026           60 IA(IC1)=NEMPTC
0027            NEMPTC=IC1
                C
0028           80 RETURN
0029            END


PROGRAM SECTIONS

 NAME        SIZE                  ATTRIBUTES

$CODE1  000516   167                RW,I,CON,LCL
```

```
FORTRAN IV-PLUS V02-04G          14:42:07     29-MAR-78          PAGE 39
RBMC.FTN         /I4/IR:BLOCKS/WR

        C
0001            SUBROUTINE EMPTYC
        C
        C------------------- ROUTINE TO LOCATE EMPTY CONTACT DATA SPACE ---
        C
0002            INCLUDE 'COMMON.FTN'
0003  *         COMMON A(3000)
0004  *         DIMENSION IA(1)
0005  *         EQUIVALENCE (A,IA)
0006            INCLUDE 'CBLOCK.FTN'
0007  *         COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,
      *        .                NBLOKS,NCYC,MCYCLE,NEMPT,RHO,EFLAG,TFRAC,
      *        .                TDEL,IBOXES,JBOXES,XL,XU,YL,YU,BSIZE,SFACT,XSIZE,
      *        .                YSIZE,UDMAX,UMUST,STIFN,STIFS,FRIC,BETA,BDT,TMAX,
      *        .                CON1,CON2,ALPHA,GRAVX,GRAVY,LOC1,LOC2,NUPDAT,NB1,
      *        .                NVARB,NFRAG,NPR,NEMPTC,NEMPTD,NEMPTG,IEND1,LBLOCK
0008  *         LOGICAL EFLAG
        C
        C------------------- DOES LIST OF EMPTY CONTACT DATA EXIST ?---
0009            IF(NEMPTC.NE.0) GO TO 10
        C------------------- NO ---
0010            CALL LIMIT(12)
0011            NEMPT=NEMPTG
0012            NEMPTG=NEMPTG+12
0013            GO TO 20
        C------------------- YES ---
0014         10 NEMPT=NEMPTC
0015            NEMPTC=IA(NEMPTC)
        C
0016         20 RETURN
        C
0017            END


PROGRAM SECTIONS

 NAME        SIZE                  ATTRIBUTES

$CODE1   000144     50             RW,I,CON,LCL
$PDATA   000004      2             RW,D,CON,LCL
$IDATA   000004      2             RW,D,CON,LCL
.$$$$.   027340   6000             RW,D,OVR,GBL
CBLOCK   000424    138             RW,D,OVR,GBL


TOTAL SPACE ALLOCATED = 030140  6192

NO FPP INSTRUCTIONS GENERATED
```

```
FORTRAN IV-PLUS V02-04G          14:42:14     29-MAR-78           PAGE 40
RBMC.FTN         /14/IR:BLOCKS/WR

0001            SUBROUTINE EMPTYD
                C
                C------------------- ROUTINE TO LOCATE EMPYTY 'DOUBLES' ---
                C
0002            INCLUDE 'COMMON.FTN'
0003 *          COMMON A(3000)
0004 *          DIMENSION IA(1)
0005 *          EQUIVALENCE (A,IA)
0006            INCLUDE 'CBLOCK.FTN'
0007 *          COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,
     *                .                  NBLOKS,NCYC,MCYCLE,NEMPT,RHO,EFLAG,TFRAC,
     *                .                  TDEL,IBOXES,JBOXES,XL,XU,YL,YU,BSIZE,SFACT,XSIZE,
     *                .                  YSIZE,UDMAX,UMOST,STIFN,STIFS,FRIC,BETA,BDT,TMAX,
     *                .                  CON1,CON2,ALPHA,GRAVX,GRAVY,LOC1,LOC2,NUPDAT,NB1,
     *                .                  NVARB,NFRAG,NPR,NEMPTC,NEMPTD,NEMPTG,IEND1,LBLOCK
0008 *          LOGICAL EFLAG
                C
                C------------------- DOES LIST OF EMPTY 'DOUBLES' EXIST ?---
0009            IF(NEMPTD.NE.0) GO TO 10
                C------------------- NO ---
0010            CALL LIMIT(2)
0011            NEMPT=NEMPTG
0012            NEMPTG=NEMPTG+2
0013            GO TO 20
                C------------------- YES ---
0014           10 NEMPT=NEMPTD
0015            NEMPTD=IA(NEMPTD)
                C
0016           20 RETURN
                C
0017            END


PROGRAM SECTIONS

 NAME       SIZE                 ATTRIBUTES

$CODE1  000144     50            RW,I,CON,LCL
$PDATA  000004      2            RW,D,CON,LCL
$IDATA  000004      2            RW,D,CON,LCL
.$$$$.  027340   6000            RW,D,OVR,GBL
CBLOCK  000424    138            RW,D,OVR,GBL

TOTAL SPACE ALLOCATED = 030140  6192

NO FPP INSTRUCTIONS GENERATED
```

```
FORTRAN IV-PLUS V02-04G          14:42:18     29-MAR-78            PAGE 41
RBMC.FTN         /I4/TR:BLOCKS/WR

0001           SUBROUTINE SCAN(NC)
               C
               C------------------- ROUTINE TO CHECK BOX ENTRIES ALONG
               C                    THE EDGE OF A BLOCK ---
               C
0002           INCLUDE 'COMMON.FTN'
0003  *        COMMON A(3000)
0004  *        DIMENSION IA(1)
0005  *        EQUIVALENCE (A,IA)
0006           INCLUDE 'CBLOCK.FTN'
0007  *        COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,
      *       .                NBLOKS,NCYC,MCYCLE,NEMPT,RHO,EFLAG,TFRAC,
      *       .                TDEL,IBOXES,JBOXES,XL,XU,YL,YU,BSIZE,SFACT,XSIZE,
      *       .                YSIZE,UDMAX,UMOST,STIFN,STIFS,FRIC,BETA,BDT,IMAX,
      *       .                CON1,CON2,ALPHA,GRAVX,GRAVY,LOC1,LOC2,NUPDAT,NB1,
      *       .                NVARB,NFRAG,NPR,NEMPTC,NEMPTD,NEMPTG,IEND1,LBLOCK
0008  *        LOGICAL EFLAG
0009           DATA TOL/1.0/
               C
0010           I2=IA(NB1)
               C------------------- DETERMINE COORDINATES OF THE ENDS OF
               C                    THE EDGE ---
0011           X1=A(IEND1)
0012           Y1=A(IEND1+1)
0013           IEND2=IA(IEND1+2)
0014           X2=A(IEND2)
0015           Y2=A(IEND2+1)
               C------------------- IS THE BLOCK FIXED ?---
0016           IF(X2.GT.XL.AND.X2.LT.XU.
              .   AND.Y2.GT.YL.AND.Y2.LT.YU) GO TO 100
               C------------------- YES, SET FIXED FLAG ---
0017           IA(I2)=2
0018           A(I2+4)=0.0
0019           A(I2+5)=0.0
0020           A(I2+6)=0.0
0021           RETURN
               C------------------- NO, CONTINUE ---
0022         100 XSTEP=1.0
0023           YSTEP=1.0
0024           NP=0
0025           I2N=0
               C------------------- START SCAN ---
0026         150 NP=NP+1
0027           NS=1
0028           XTOL=0.0
0029           YTOL=0.0
               C------------------- DETERMINE SCANNING DIRECTION ---
0030           IF(ABS(X2-X1).GT.ABS(Y2-Y1)) NS=0
               C------------------- DETERMINE X AND Y INCREMENTS ---
0031           IF(X1.GT.X2) XSTEP=-XSTEP
0032           IF(Y1.GT.Y2) YSTEP=-YSTEP
               C------------------- DETERMINE TOLERANCES ---
0033           IF(NS.EQ.0) XTOL=TOL*XSTEP
0034           IF(NS.EQ.1) YTOL=TOL*YSTEP
               C------------------- DETERMINE LIMITS OF EDGE ---
0035           IBOX1=MIN0(IFIX((X1-XTOL)/BSIZE)+1,IBOXES)
```

```
FORTRAN IV-PLUS V02-04G          14:42:18     29-MAR-78           PAGE 42
RBMC.FTN         /I4/TR:BLOCKS/WR

0036           JBOX1=MIN0(IFIX((Y1-YTOL)/BSIZE),JBOXES-1)
0037           IBOXN=MIN0(IFIX((X2+XTOL)/BSIZE)+1,IBOXES)
0038           JBOXN=MIN0(IFIX((Y2+YTOL)/BSIZE),JBOXES-1)
0039           NBOX1=JBOX1*IBOXES+IBOX1
0040           NBOXN=JBOXN*IBOXES+IBOXN
       C------------------- SET BOX POINTERS ---
0041           I21=M2+NBOX1-1
0042           I22=M2+NBOXN-1
0043           IF(I2N.EQ.I22) GO TO 750
0044           I2N=0
0045           GO TO 450
       C------------------- DETERMINE DIRECTION OF BOX INCREMENT ---
0046         200 IF(IBOX1.EQ.IBOXN) GO TO 650
0047           IF(JBOX1.EQ.JBOXN) GO TO 700
0048           IF(I2N.EQ.0) GO TO 210
0049           IF(NS.EQ.0) GO TO 650
0050           GO TO 700
       C------------------- SET UP EQUATION OF LINE ---
0051         210 TEMP1=(Y2-Y1)/(X2-X1)
0052           TEMP2=Y1-X1*TEMP1
0053           IF(NS.EQ.0) GO TO 250
       C------------------- DETERMINE COORDINATES OF INTERSECTION
       C                    WITH BOX GRID ---
0054           YN=FLOAT(JBOX1)*BSIZE
0055           IF(Y1.GT.Y2) YN=YN+BSIZE
0056           GO TO 300
0057         250 XN=FLOAT(IBOX1-1)*BSIZE
0058           IF(X1.GT.X2) XN=XN+BSIZE
0059         300 IF(NS.EQ.0) GO TO 350
0060           YN=YN+YSTEP*BSIZE
0061           XN=(YN-TEMP2)/TEMP1
0062           GO TO 400
0063         350 XN=XN+XSTEP*BSIZE
0064           YN=TEMP1*XN+TEMP2
       C------------------- DETERMINE THE BOX IN WHICH THE
       C                     INTERSECTION OCCURS ---
0065         400 JBOXN=MIN0(IFIX((YN+YTOL)/BSIZE),JBOXES-1)
0066           IBOXN=MIN0(IFIX((XN+XTOL)/BSIZE)+1,IBOXES)
0067           NBOXN=JBOXN*IBOXES+IBOXN
       C------------------- SET BOX POINTER ---
0068           I2N=M2+NBOXN-1
0069           GO TO 200
       C------------------- IS BLOCK ALREADY ENTERED ?---
0070         450 IF(IA(I21).EQ.0) GO TO 550
0071           I23=IA(I21)
0072         500 IF(IA(I23).EQ.NB1) GO TO 600
0073           IF(IA(I23+1).EQ.0) GO TO 550
0074           I23=IA(I23+1)
0075           GO TO 500
       C------------------- NO, CALL BOXING ROUTINE ---
0076         550 CALL BOXC
0077           GO TO 800
       C------------------- IS ALL THE EDGE SCANNED ?---
0078         600 IF(I21.EQ.I22) GO TO 750
       C------------------- YES, GET NEXT EDGE OR RETURN ---
       C------------------- NO, INCREMENT BOX NUMBER ---
```

```
FORTRAN IV-PLUS V02-04G          14:42:18     29-MAR-78           PAGE 43
RBMC.FTN         /14/TR:BLOCKS/WR

0079           IF(I21.EQ.I2N) GO TO 300
0080           GO TO 200
       C------------------ INCREMENT IN Y DIRECTION ---
0081         650 I21=I21+IFIX(YSTEP)*IBOXES
0082           JBOX1=JBOX1+IFIX(YSTEP)
0083           GO TO 450
       C------------------ INCREMENT IN X DIRECTION ---
0084         700 I21=I21+IFIX(XSTEP)
0085           IBOX1=IBOX1+IFIX(XSTEP)
0086           GO TO 450
0087         750 XSTEP=ABS(XSTEP)
0088           YSTEP=ABS(YSTEP)
       C------------------ IS SCAN COMPLETE ?---
0089           IF(NP.EQ.NC) GO TO 800
       C------------------ NO, GET NEXT EDGE ---
0090           I2N=I22
0091           X1=A(IEND2)
0092           Y1=A(IEND2+1)
0093           IEND2=IA(IEND2+2)
0094           X2=A(IEND2)
0095           Y2=A(IEND2+1)
0096           GO TO 150
       C------------------ YES, RETURN ---
0097         800 RETURN
0098           END


PROGRAM SECTIONS

 NAME        SIZE                  ATTRIBUTES

$CODE1   003216    839              RW,I,CON,LCL
$IDATA   000046     19              RW,D,CON,LCL
$VARS    000154     54              RW,D,CON,LCL
$TEMPS   000004      2              RW,D,CON,LCL
.$$$$.   027340   6000              RW,D,OVR,GBL
CBLOCK   000424    138              RW,D,OVR,GBL


TOTAL SPACE ALLOCATED = 033430  7052
```

```
FORTRAN IV-PLUS V02-04G          14:43:12     29-MAR-78           PAGE 44
RBMC.FTN         /I4/TR:BLOCKS/WR

0001            SUBROUTINE LIMIT(NREQ)
                C
                C------------------- CHECK MEMORY ALLOCATION ---
                C
0002            INCLUDE 'CBLOCK.FTN'
0003 *          COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,
     *                         NBLOKS,NCYC,MCYCLE,NEMPT,RHO,EFLAG,TFRAC,
     *                         TOEL,IBOXES,JBOXES,XL,XU,YL,YU,BSIZE,SFACT,XSIZE,
     *                         YSIZE,UDMAX,UMOST,STIFN,STIFS,FRIC,BETA,BDT,TMAX,
     *                         CON1,CON2,ALPHA,GRAVX,GRAVY,LOC1,LOC2,NUPDAT,NB1,
     *                         NVARB,NFRAG,NPR,NEMPTC,NEMPTD,NEMPTG,IEND1,LBLOCK
0004 *          LOGICAL EFLAG
                C
0005            IF((NEMPTG+NREQ).LE.M4.AND.(NEMPT+NREQ).LE.M4) RETURN
                C
0006            EFLAG=.TRUE.
0007            WRITE(6,1000)
0008            CALL FINISH
                C
0009             1000 FORMAT(39H !!! ERROR : MEMORY ALLOCATION EXCEEDED)
0010            END


PROGRAM SECTIONS

 NAME        SIZE                    ATTRIBUTES

$CODE1   000204     66               RW,I,CON,LCL
$IDATA   000054     22               RW,D,CON,LCL
CBLOCK   000424    138               RW,D,OVR,GBL


TOTAL SPACE ALLOCATED = 000704    226

NO FPP INSTRUCTIONS GENERATED
```

```
FORTRAN IV-PLUS V02-04G          14:43:16     29-MAR-78            PAGE 45
RBMC.FTN         /I4/TR:BLOCKS/WR

0001            SUBROUTINE FINISH
                C
                C------------------- TIDY UP AND STOP ---
                C
0002            INCLUDE 'COMMON.FTN'
0003 *          COMMON A(3000)
0004 *          DIMENSION IA(1)
0005 *          EQUIVALENCE (A,IA)
0006            INCLUDE 'CBLOCK.FTN'
0007 *          COMMON /CBLOCK/ HED(20),NBLOKM,NBOXES,M1,M2,M3,M4,
     *         .                NBLOKS,NCYC,MCYCLE,NEMPT,RHO,EFLAG,TFRAC,
     *         .                TDEL,IBOXES,JBOXES,XL,XU,YL,YU,BSIZE,SFACT,XSIZE,
     *         .                YSIZE,UDMAX,UMOST,STIFN,STIFS,FRIC,BETA,BDT,TMAX,
     *         .                CON1,CON2,ALPHA,GRAVX,GRAVY,LOC1,LOC2,NUPDAT,NB1,
     *         .                NVARB,NFRAG,NPR,NEMPTC,NEMPTD,NEMPTG,IEND1,LBLOCK
0008 *          LOGICAL EFLAG
                C
0009            CALL PLOTND
0010            WRITE(6,2000) MCYCLE,NUPDAT
                C
                C------------------- WRITE RESTART FILE IF NO ERRORS ---
0011            IF(EFLAG) GOTO 100
0012            REWIND 1
0013            WRITE(1) (HED(I),I=1,LBLOCK)
0014            WRITE(1) (A(I),I=1,M4)
0015            WRITE(6,2001)
0016          100 STOP
                C
0017           2000 FORMAT(30X,13H TOTAL CYCLES,I10/
                .       30X,13H NO. UPDATES ,I10)
0018           2001 FORMAT(30X,32H A RESTART FILE HAS BEEN WRITTEN)
                C
0019            END


PROGRAM SECTIONS

 NAME        SIZE                  ATTRIBUTES

$CODE1   000364    122              RW,I,CON,LCL
$IDATA   000120     40              RW,D,CON,LCL
$VARS    000004      2              RW,D,CON,LCL
.$$$$.   027340   6000              RW,D,OVR,GBL
CBLOCK   000424    138              RW,D,OVR,GBL

TOTAL SPACE ALLOCATED = 030474  6302

NO FPP INSTRUCTIONS GENERATED

,CUNDALL.LST/-SP=RBMC/I4/LI:1/CO:10
```

APPENDIX XV: LISTING OF PROGRAM JOINT

NOTE: All arguments, common block symbols and array variables are defined on the listing itself.

```
      SUBROUTINE JOINT(DUS,DUN,STRS,STRN,JID,RIGID)
      LOGICAL RIGID
      COMMON /FRIC/ PHIB,AKS,AKN,AKD,AJRC,DEGRAD
      DIMENSION SS(50),SN(50),US(50),UN(50),UND(50),D(50)
C
C   DUS  = SHEAR DISPLACEMENT INCREMENT - INPUT TO <JOINT>
C   DUN  = NORMAL      "          "        "    "     "
C   STRS = NEW SHEAR STRESS - OUTPUT FROM <JOINT>
C   STRN =  "  NORMAL  "          "      "     "
C
C   JID = JOINT ID;  IF JID < 0 THE JOINT WILL BE INITIALIZED
C                    I.E. DAMAGE WILL BE RESET TO ZERO AND
C                    STRESSES AND DISPLACEMENTS ZEROED
C
C   IF RIGID = .TRUE. JOINT IS RIGID IN THE NORMAL DIRECTION
C      (IN THIS CASE, STRN MUST BE INPUT TO <JOINT>)
C                                -----
C
C   PHIB = RESIDUAL (OR BASE) FRICTION ANGLE IN DEGREES
C   AKS  = ELASTIC SHEAR STIFFNESS
C   AKN  =    "     NORMAL     "
C   AKD  = DILATION CONSTANT
C   AJRC = JOINT ROUGHNESS COEFFICIENT (SEE BARTON)
C   DEGRAD CONVERSION FACTOR FROM DEGREES TO RADIANS
C
C   SS(JID) = SHEAR STRESS FOR JOINT NUMBER JID
C   SN(JID) = NORMAL   "      "     "      "      "
C   US(JID) = SHEAR DISPLACEMENT "      "      "
C   UN(JID) = NORMAL      "        "      "      "
C  UND(JID) = VIRTUAL NORMAL DISPLACEMENT DUE TO DILATION
C    D(JID) = "DAMAGE" (SEE TEXT)
C
      IF(JID.LE.0) GOTO 300
C-------------------- GET NORMAL STRESS FIRST ---
      UN(JID)=UN(JID)+DUN
      SNOLD=SN(JID)
      IF(.NOT.RIGID) GOTO 10
      SN(JID)=STRN
      GOTO 15
   10 SN(JID)=FSN(UN(JID))
      STRN=SN(JID)
   15 SNAV=0.5*(SNOLD+SN(JID))
C
C
C-------------------- NOW FOR THE SHEAR STRESS ---
C
C---ELASTIC INCREMENTS FIRST---
      SSNEW=SS(JID)+AKS*DUS
C---CHECK FOR UNLOADING---
      IF(SIGN(1.0,SNNEW).NE.SIGN(1.0,DUS)) GOTO 50
C---MUST BE LOADING---
C---CHECK IF BELOW YIELD---
      SSMAX=SNAV*TAN(PHIB*DEGRAD)
      IF(ABS(SSNEW).LE.SSMAX) GOTO 50
C---WE MUST BE IN PEAK REGION---
C---APPARENT SHEAR DISP (AS FAR AS CURVE IS CONCERNED)---
```

```
      USAPP=D(JID)/SNAV+DUS
C---MAGNITUDE OF STRENGTH---
      SSADD=FSS(USAPP,SNAV)
      SSPK=SSMAX+SSADD
C---CHECK IF WE EXCEED STRENGTH---
      IF(ABS(SSNEW).LE.SSPK) GOTO 50
C---YES, REDUCE SHEAR STRENGTH AND ACCUMULATE DAMAGE---
      SSNEW=SIGN(SSPK,SSNEW)
      IF(SSADD.EQ.0.0) GOTO 50
      D(JID)=D(JID)+SNAV*ABS(DUS)
C---COMPUTE VIRTUAL NORMAL DISPLACEMENT DUE TO DILATION---
      UND(JID)=UND(JID)+AKD*(1.0/SNAV-1.0)*DUS
C---NEW, FINAL SHEAR STRESS---
   50 SS(JID)=SSNEW
      STRS=SSNEW
      RETURN
C
C---INITIALIZATION---
C
  300 JJ=-JID
      SS(JJ)=0.0
      SN(JJ)=0.0
      US(JJ)=0.0
      UN(JJ)=0.0
      UND(JJ)=0.0
  310 D(JJ) =0.0
      RETURN
      END
```

```
      FUNCTION FSS(US,SN)
C
C    "BUMP" ON STRESS/STRAIN CURVE
C    (NORMALIZED)
C
      COMMON /FRIC/ PHIB,AKS,AKN,AKD,AJRC,DEGRAD
      DIMENSION Y(11)
      DATA Y /0.0,0.32,0.61,0.82,0.94,1.0,0.93,0.54,0.14,
     .       0.02,0.0/
      DATA NP /10/
C---SCALE DISP, SO THAT MAX DISP = NORMAL STRESS ---
      SCL=SN
C---CHECK FOR END OF CURVE--
      IF(US.GE.SCL) GOTO 100
      USSCL=US/SCL*FLOAT(NP)
      IGRF=USSCL+1.0
      IGRF=MIN0(IGRF,NP)
      YINT=Y(IGRF)+(USSCL-FLOAT(IGRF-1))*(Y(IGRF+1)-Y(IGRF))
C---SCALE STRESS ACCORDING TO BARTON'S FORMULA---
      YMAX=SN*(TAN((AJRC*ALOG10(1.0/SN)+PHIB)*DEGRAD)
     .     -TAN(PHIB*DEGRAD))
      FSS=YINT*YMAX
      RETURN
  100 FSS=0.0
      RETURN
      END
```

```
      FUNCTION FSN(UN)
      COMMON /FRIC/ PHIB,AKS,AKN
C
C   LINEAR NORMAL STRESS/DISPLACEMENT
C
      FSN=AKN*UN
      RETURN
      END
```

APPENDIX XVI : LISTING OF PROGRAM DBLOCK AND LIST OF FORTRAN NAMES

NMAX = maximum number of nodes

MMAX = maximum number of zones

IMAX = maximum number of zones about a node

IFLAG = indicator of type of input provided

KTOT = number of 4-zone cells in the generation of a regular columnar mesh

LCONT = number of initial contacts

X(N), Y(N) = coordinates of node N

XD(N), YD(N) = velocity components of node N

N = node number

M = zone number

I = index designating ordering of zones about a node

J = index designating ordering of nodes about a zone

XX(M), YY(M), XY(M) = stress components in zone M

AM(M) = mass of zone M

GPM(N) = grid point mass of node N

LAME1(M), LAME2(M) = Lamé constants in zone M

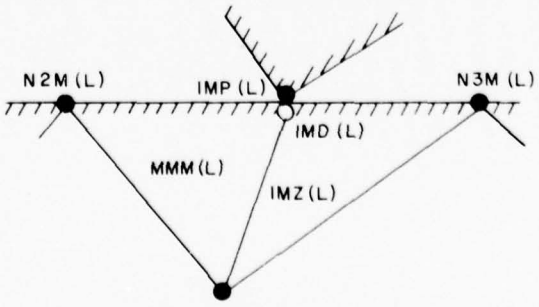
COHES(M), TANPHI(M), TANPSI(M), = plastic parameters in zone M

MAN(N,I) = I^{th} zone about node N

NAM(M,J) = J^{th} node about zone M

IMP(L), IMD(L), N2M(L), N3M(L) see adjacent figure

XNC(L) = mass of impacting node, if isolated; zero otherwise



LOCAL NAMES FOR CONTACT L

STNL = contact stiffness (loading)

STNU = contact stiffness (unloading)

DN = maximum normal displacement

STSH = shear stiffness

XMU = coefficient of friction

NITER = maximum number of iterations

FRAC = fraction of critical time step

GRAV = gravity

RMIN, FMIN = parameters for stiffness-proportional damping

ARAT = minimum aspect ratio

TFRAC = parameter for "tickling"

NPRI = number of iterations between printed outputs

NPLOT = number of iterations between points in plotted history

SCALEX, SCALEY = scales for mesh plots

ITPL(I) = iteration numbers at which mesh plots are requested

NBLOCK = number of blocks

SCALE = scale for plot of computer generated mesh

NCORN = number of corners defining the block

LIST(l,I) = I^{th} corner of block l

AMAXL = maximum edge length desired

PROGRAM LISTING - DBLOCK

```
FORTRAN IV-PLUS V02-04G          11:24:54     29-MAR-78           PAGE 1
DBLOCK.FTN       /TR:BLOCKS/WR

0001           PROGRAM DBLOCK
        C*****************************************************************
        C                                                                *
        C        DRIVING ROUTINE FOR THE PROGRAM                         *
        C                                                                *
        C*****************************************************************
0002           INCLUDE 'COMMON.FTN'
0003 *         REAL LAME1,LAME2
0004 *         COMMON /CGRID/ X(70),Y(70),XD(70),YD(70),GPM(70),
     *        .   XNPH(70),YNPH(70),
     *        .   XX(70),YY(70),XY(70),DXXFV(70),DYYFV(70),DXYFV(70),
     *        .   AM(70),RHO(70),LAME1(70),LAME2(70),COHES(70),TANPH1(70),
     *        .   TANPSI(70),MAN(70,10),NAM(70,3)
0005 *         COMMON /COTHR/ DT,NITER,AKAT,GRAV,ITER,NMAX,MMAX,IMAX,TIME,
     *        .   ZERO,NPRI,LCONT,IFLAG,KTOT,NBUF,NPL,NPLOT,ITPL(10),
     *        .   SCALEX,SCALEY
0006 *         COMMON /CPLOT/ BUF1(202),BUF2(202),BUF3(202)
0007 *         COMMON /CIMPC/ IMP(10),IMD(10),IFL(10),IMZ(10),IFG(10),
     *        .   IFR(10),N2M(10),N3M(10),MMM(10),N4M(10),TOLC(10),
     *        .   STNL,STNU,STSH,XMU,TOL,BDT,BDT1,CON1,CON2,TFRAC,
     *        .   FN(10),FS(10),DRN(10),DRS(10),DRST(10),XNC(10),YNC(10),DN
0008           CALL SETUP
0009           CALL CYCLE
0010           STOP
0011           END


PROGRAM SECTIONS

 NAME        SIZE                  ATTRIBUTES

$CODE1   000054     22             RW,I,CON,LCL
$IDATA   000002      1             RW,D,CON,LCL
CGRID    016374   3710             RW,D,OVR,GBL
COTHR    000110     36             RW,D,OVR,GBL
CPLOT    004570   1212             RW,D,OVR,GBL
CIMPC    001064    282             RW,D,OVR,GBL


TOTAL SPACE ALLOCATED = 024436  5263

NO FPP INSTRUCTIONS GENERATED

,LIST.LST/LI:1/-SP=DBLOCK
```

```
FORTRAN IV-PLUS V02-04G          11:24:59     29-MAR-78          PAGE 1
SETUP.FTN       /TR:BLOCKS/WR

0001          SUBROUTINE SETUP
        C*****************************************************************
        C                                                                *
        C       INPUT RECEPTION AND HANDLING                             *
        C                                                                *
        C*****************************************************************
0002          INCLUDE 'COMMON.FTN'
0003  *       REAL LAME1,LAME2
0004  *       COMMON /CGRID/ X(70),Y(70),XD(70),YD(70),GPM(70),
      *      .   XNPH(70),YNPH(70),
      *      .   XX(70),YY(70),XY(70),DXXFV(70),DYYFV(70),DXYFV(70),
      *      .   AM(70),RHO(70),LAME1(70),LAME2(70),COHES(70),TANPHI(70),
      *      .   TANPSI(70),MAN(70,10),NAM(70,3)
0005  *       COMMON /CUTHR/ DT,NITER,ARAT,GRAV,ITER,NMAX,MMAX,IMAX,TIME,
      *      .   ZERO,NPRI,LCONT,IFLAG,KTOT,NBUF,NPL,NPLOT,ITPL(10),
      *      .   SCALEX,SCALEY
0006  *       COMMON /CPLOT/ BUF1(202),BUF2(202),BUF3(202)
0007  *       COMMON /CIMPC/ IMP(10),IMD(10),IFL(10),IMZ(10),IFG(10),
      *      .   IFR(10),N2M(10),N3M(10),MMM(10),N4M(10),TOLC(10),
      *      .   STNL,STNU,STSH,XMU,TOL,BDT,BDT1,CON1,CON2,TFRAC,
      *      .   FN(10),FS(10),DRN(10),DRS(10),DRST(10),XNC(10),YNC(10),DN
0008          AREA3(X1,Y1,X2,Y2,X3,Y3) = -0.5*((Y1+Y2)*(X2-X1)+
             .       (Y3+Y2)*(X3-X2)+(Y1+Y3)*(X1-X3))
        C-------INPUT VARIABLES ----------------------------------
0009          READ  1000, NMAX,MMAX,IMAX,IFLAG,KTOT,LCONT
0010          PRINT 2000, NMAX,MMAX,IMAX
0011          IF (IFLAG .EQ. 1) GO TO 12
0012          IF (IFLAG .EQ. 0) GO TO 18
0013          CALL MESH
0014          GO TO 7
0015       18 DO 19 N=1,NMAX
0016       19 READ  1001, X(N),Y(N),XD(N),YD(N)
0017          PRINT 2001, (N,X(N),Y(N),XD(N),YD(N),N=1,NMAX)
0018          DO 11 M=1,MMAX
0019       11 READ  1005, XX(M),YY(M),XY(M),RHO(M),LAME1(M),LAME2(M),
             .            COHES(M),TANPHI(M),TANPSI(M)
0020          PRINT 2002, (M,XX(M),YY(M),XY(M),RHO(M),LAME1(M),LAME2(M),
             .            COHES(M),TANPHI(M),TANPSI(M),M=1,MMAX)
0021          DO 1 N=1,NMAX
0022        1 READ  1003, (MAN(N,I),I=1,IMAX)
0023          PRINT 2003
0024          DO 2 N=1,NMAX
0025        2 PRINT 3003, N,(MAN(N,I),I=1,IMAX)
0026          READ  1004, ((NAM(M,J),J=1,3),M=1,MMAX)
0027          PRINT 2004, (M,(NAM(M,J),J=1,3),M=1,MMAX)
0028        7 READ  1006, (IMP(L),N2M(L),N3M(L),XNC(L),L=1,LCONT)
0029          PRINT 2005, LCONT,(L,IMP(L),XNC(L),N2M(L),N3M(L),L=1,LCONT)
0030          READ  1001, STNL,STNU,DN,STSH,XMU
0031          PRINT 2009, STNL,STNU,DN,STSH,XMU
0032          GO TO 13
0033       12 CALL INPUT
0034       13 READ  1002, NITER,FRAC,GRAV,ARAT,TFRAC,NPRI,NPLOT
0035          PRINT 2006, NITER,FRAC,GRAV,ARAT,TFRAC
0036          READ  1000, (ITPL(I),I=1,10)
0037          PRINT 2010, (ITPL(I),I=1,10)
0038          READ  1001, SCALEX,SCALEY
```

```
FORTRAN IV-PLUS V02-04G          11:24:59     29-MAR-78            PAGE 2
SETUP.FTN        /TR:BLOCKS/WR

0039           PRINT 2011, SCALEX,SCALEY
0040           READ  1001, RMIN,FMIN
0041           PRINT 2012, RMIN,FMIN
       C-------INITIALIZATIONS ---------------------------------------
0042           DO 3 N=1,NMAX
0043           XNPH(N) = X(N)
0044         3 YNPH(N) = Y(N)
       C
0045           IF (LCONT .EQ. 0) GO TO 6
0046           DO 4 L=1,LCONT
0047           GPM(IMP(L)) = XNC(L)
0048           DRN(L) = 0
0049           DRS(L) = 0
0050           XNC(L) = 0
0051           YNC(L) = 0
0052           FN(L)  = 0
0053           FS(L)  = 0
0054         4 DRST(L)= 0
0055         6 NPL  = 0
0056           DRNPH= 0
0057           DRSPH= 0
0058           DS   = 0
0059           ZERO = 0
0060           PI2  = 8.*ATAN(1.)
       C-------COMPUTE CRITICAL TIME STEP --------------------------
0061           DT   = 1.E20
0062           DO 20 N=1,NMAX
0063           DO 15 I=1,IMAX
0064           M    = MAN (N,1)
0065           IF (M .EQ. 0) GO TO 20
0066           NN   = NAM(M,2)
0067           IF (NAM(M,1) .EQ. N) NN=NAM(M,3)
0068           IF (NAM(M,2) .EQ. N) NN=NAM(M,1)
0069           DS   = (X(N)-X(NN))**2+(Y(N)-Y(NN))**2
0070           MM   = MAN(N,I+1)
0071           IF (MM .EQ. 0) MM=MAN(N,1)
0072           VMS  = (LAME1(M)+2.*LAME2(M))/RHO(M)
0073           VMMS = (LAME1(MM)+2.*LAME2(MM))/RHO(MM)
0074           DTN  = SQRT(DS/AMAX1(VMS,VMMS))
0075           DT   = AMIN1(DT,DTN)
0076        15 CONTINUE
0077        20 CONTINUE
0078           FRED = 2.*ARAT
0079           DT   = DT*AMIN1(FRED,1.)*FRAC
0080           PRINT 2008,DT
       C-------COMPUTE ZONE MASSES ---------------------------------
0081           DO 40 M=1,MMAX
0082        40 AM(M)  = RHO(M)*AREA3(X(NAM(M,1)),Y(NAM(M,1)),X(NAM(M,2)),
              .       Y(NAM(M,2)),X(NAM(M,3)),Y(NAM(M,3)))
       C-------COMPUTE GRID-POINT MASSES --------------------------
0083           DO 10 N=1,NMAX
0084           GPMP   = 0
0085           DO  5 I=1,IMAX
0086           M      = MAN(N,I)
0087           IF (M .EQ. 0) GO TO 10
0088         5 GPMP   = GPMP+AM(M)
```

```
FORTRAN IV-PLUS V02-04G          11:24:59     29-MAR-78           PAGE 3
SETUP.FTN        /TR:BLOCKS/WR

0089        10 GPM(N) = GPMP/3.
0090           PRINT 2007,(N,GPM(N),N=1,NMAX)
           C-------RAYLEIGH DAMPING ------------------------------------
0091           ALPHA  = PI2*RMIN*FMIN*DT/2.
0092           CON1   = 1.-ALPHA
0093           CON2   = 1./(1.+ALPHA)
0094           BETA   = RMIN/(PI2*FMIN)
0095           BDT    = BETA/DT
0096           BDT1   = 1.+BDT
           C-------CREATION OF NEW GRID POINTS UPON CONTACT -------------
0097           IF (IFLAG .EQ. 1) RETURN
0098           DO 50 L=1,LCONT
0099           IFL(L) = 0
0100           IFG(L) = 1
0101           CALL CREATE(L)
0102        50 CONTINUE
0103           RETURN
           C
0104       1000 FORMAT (8I10)
0105       1001 FORMAT (8F10.0)
0106       1002 FORMAT (I10,4F10.0,2I10)
0107       1003 FORMAT (1015)
0108       1004 FORMAT (315)
0109       1005 FORMAT (9F8.0)
0110       1006 FORMAT (3I10,F10.0)
0111       2000 FORMAT ('0MAX NUMBER OF NODES                =',I6,/
                .        ' MAX NUMBER OF ZONES               =',I6,/
                .        ' MAX NUMBER OF ZONES ABOUT A NODE  =',I6)
0112       2001 FORMAT (//' NODE DATA'//5X,'N',6X,'X',11X,'Y',10X,'XD',
                .        10X,'YD',/(1X,I5,1P4E12.4))
0113       2002 FORMAT (//' ZONE DATA'//5X,'M',5X,'XX',10X,'YY',10X,'XY',
                .        10X,'AM',9X,'LAME1'7X,'LAME2'7X,'COHES'7X,'TANPHI'
                .        6X,'TANPSI'/(I6,1P9E12.4))
0114       2003 FORMAT (//' ZONES SURROUNDING EACH NODE'//5X,'N',5X,'1',
                .        5X,'2',5X,'3',5X,'4',5X,'5',5X,'6',5X,'7',5X,'8',
                .        5X,'9',5X,'10')
0115       3003 FORMAT (1X,11I6)
0116       2004 FORMAT (//' NODES SURROUNDING EACH ZONE'//5X,'M',5X,'1',
                .        5X,'2',5X,'3',/(4I6))
0117       2005 FORMAT (//' CONTACTS ESTABLISHED:'I3
                .        //' CONTACT   IMPACTING     IMPACTING'11X,'BETWEEN'
                .         /'  NUMBER     NODE          MASS'11X,'NODE AND NODE'
                .         /(3X,I3,7X,I3,7X,1PE12.4,7X,I3,6X,I3))
0118       2006 FORMAT (//' NUMBER OF ITERATIONS             ='I5,
                .         /' FRACTION OF CRITICAL TIME STEP  ='1PE12.4,
                .         /' ACCELERATION OF GRAVITY         ='1PE12.4,
                .         /' MINIMUM ASPECT RATIO OF ZONES   ='1PE12.4,
                .         /' TOLERANCE / ZONE LENGTH         ='1PE12.4)
0119       2007 FORMAT (//' GRID POINT MASSES'//5X,'N'4X,'MASS'
                .         /(I6,1PE12.4))
0120       2008 FORMAT (//' TIME INCREMENT ='1PE12.4)
0121       2009 FORMAT(//' CONTACT PROPERTIES'
                .        //' LOADING NORMAL STIFFNESS        ='1PE12.4,
                .         /' UNLOADING NORMAL STIFFNESS      ='1PE12.4,
                .         /' MAX RELATIVE NORMAL DISPLACEMENT='1PE12.4,
                .         /' SHEAR STIFFNESS                 ='1PE12.4,
```

```
FORTRAN IV-PLUS V02-04G          11:24:59     29-MAR-78          PAGE 4
SETUP.FTN       /TR:BLOCKS/WR

                .          /' COEFFICIENT OF FRICTION          ='1PE12.4)
0122      2010 FORMAT (//' MESH PLOTTED AT ITERATIONS          ='(1515))
0123      2011 FORMAT (//' SCALE FOR PLOTTING: X-DIRECTION ='1PE12.4,
                .          /'                        Y-DIRECTION ='1PE12.4)
0124      2012 FORMAT (//' RAYLEIGH DAMPING PARAMETERS'
                .          /'           PERCENT DAMPING          ='2PF12.4,
                .          /'           AT FREQUENCY (HZ)        ='1PE12.4)
0125      2100 FORMAT (//' NO SPACE TO STORE NEW LINK -- JOB ABORTED')
0126           END


PROGRAM SECTIONS

 NAME        SIZE                    ATTRIBUTES

$CODE1   005266  1371            RW,I,CON,LCL
$PDATA   000010     4            RW,D,CON,LCL
$IDATA   002634   718            RW,D,CON,LCL
$VARS    000106    35            RW,D,CON,LCL
$TEMPS   000010     4            RW,D,CON,LCL
CGRID    016374  3710            RW,D,OVR,GBL
COTHR    000110    36            RW,D,OVR,GBL
CPLOT    004570  1212            RW,D,OVR,GBL
CIMPC    001064   282            RW,D,OVR,GBL


TOTAL SPACE ALLOCATED = 034630  7372

,LIST.LST/LI:I/-SP=SETUP
```

```
FORTRAN IV-PLUS V02-04G          11:25:45     29-MAR-78           PAGE 1
CYCLE.FTN        /TR:BLOCKS/WR

0001            SUBROUTINE CYCLE
              C*****************************************************************
              C                                                                *
              C        DRIVING ROUTINE FOR THE COMPUTATION CYCLE               *
              C                                                                *
              C*****************************************************************
0002            INCLUDE 'COMMON.FTN'
0003 *          REAL LAME1,LAME2
0004 *          COMMON /CGRID/ X(70),Y(70),XD(70),YD(70),GPM(70),
     *         .   XNPH(70),YNPH(70),
     *         .   XX(70),YY(70),XY(70),DXXFV(70),DYYFV(70),DXYFV(70),
     *         .   AM(70),RHO(70),LAME1(70),LAME2(70),COMES(70),TANPH1(70),
     *         .   TANPSI(70),MAN(70,10),NAM(70,3)
0005 *          COMMON /COTHR/ DT,NITER,ARAT,GRAV,ITER,NMAX,MMAX,IMAX,TIME,
     *         .   ZERO,NPRI,LCONT,IFLAG,KTOT,NBUF,NPL,NPLOT,ITPL(10),
     *         .   SCALEX,SCALEY
0006 *          COMMON /CPLOT/ BUF1(202),BUF2(202),BUF3(202)
0007 *          COMMON /C1MPC/ IMP(10),IMD(10),IFL(10),IMZ(10),IFG(10),
     *         .   IFR(10),N2M(10),N3M(10),MMM(10),N4M(10),TOLC(10),
     *         .   STNL,STNO,STSH,XMU,TOL,BDT,BDT1,CON1,CON2,TFRAC,
     *         .   FN(10),FS(10),DRN(10),DRS(10),DRST(10),XNC(10),YNC(10),DN
0008            CALL PLOTST(0.025,'CM')
0009            DO 100 ITER=1,NITER
0010            CALL OUTPUT
0011            CALL MOTION
0012            CALL STRESS
0013            TIME = TIME+DT
0014          100 CONTINUE
0015            CALL OUTPUT
0016            WRITE (2) BUF1
0017            WRITE (2) BUF2
0018            WRITE (4) BUF3
0019            WRITE (4) BUF2
0020            CALL PLOT(0.,0.,3)
0021            CALL PLOTND
0022            RETURN
0023            END


PROGRAM SECTIONS

 NAME        SIZE                    ATTRIBUTES

$CODE1   000310    100              RW,I,CON,LCL
$PDATA   000020      8              RW,D,CON,LCL
$IDATA   000050     20              RW,D,CON,LCL
$TEMPS   000002      1              RW,D,CON,LCL
CGRID    016374   3710              RW,D,OVR,GBL
COTHR    000110     36              RW,D,OVR,GBL
CPLOT    004570   1212              RW,D,OVR,GBL
C1MPC    001064    282              RW,D,OVR,GBL


TOTAL SPACE ALLOCATED = 024762  5369

,LIST.LST/LI:1/-SP=CYCLE
```

```
FORTRAN IV-PLUS V02-04G          11:25:52     29-MAR-78           PAGE 1
MOTION.FTN       /TR:BLOCKS/WR

0001            SUBROUTINE MOTION
              C*****************************************************************
              C                                                                *
              C       MOMENTUM BALANCE IN CONTINUA                             *
              C                                                                *
              C*****************************************************************
0002            INCLUDE 'COMMON.FTN'
0003  *         REAL LAME1,LAME2
0004  *         COMMON /CGRID/ X(70),Y(70),XD(70),YD(70),GPM(70),
      *        .   XNPH(70),YNPH(70),
      *        .   XX(70),YY(70),XY(70),DXXFV(70),DYYFV(70),DXYFV(70),
      *        .   AM(70),RHO(70),LAME1(70),LAME2(70),COHES(70),TANPHI(70),
      *        .   TANPSI(70),MAN(70,10),NAM(70,3)
0005  *         COMMON /COTHR/ DT,NITER,ARAT,GRAV,ITER,NMAX,MMAX,IMAX,TIME,
      *        .   ZERO,NPRI,LCONT,IFLAG,KTOT,NBUF,NPL,NPLOT,ITPL(10),
      *        .   SCALEX,SCALEY
0006  *         COMMON /CPLOT/ BUF1(202),BUF2(202),BUF3(202)
0007  *         COMMON /CIMPC/ IMP(10),IMD(10),IFL(10),IMZ(10),IFG(10),
      *        .   IFR(10),N2M(10),N3M(10),MMM(10),N4M(10),TOLC(10),
      *        .   STNL,STNU,STSH,XMU,TOL,BDT,BDT1,CON1,CON2,TFRAC,
      *        .   FN(10),FS(10),DRN(10),DRS(10),DRST(10),XNC(10),YNC(10),DN
              C.....................................
0008            COMMON /SHEAR/ FXSUM,FYSUM
0009            FXSUM = 0
0010            FYSUM = 0
              C.....................................
              C-------MAIN ITERATION LOOP ---------------------
0011            DO 100 N=1,NMAX
0012            IF (GPM(N) .LE. 0) GO TO 100
0013            FX    = 0
0014            FY    = 0
0015            DO 90 I=1,IMAX
0016            M     = MAN(N,I)
0017            IF (M .EQ. 0) GO TO 90
0018            IF (N .NE. NAM(M,1)) GO TO 10
0019            DX    = X(NAM(M,3))-X(NAM(M,2))
0020            DY    = Y(NAM(M,3))-Y(NAM(M,2))
0021            GO TO 30
0022         10 IF (N .NE. NAM(M,2)) GO TO 20
0023            DX    = X(NAM(M,1))-X(NAM(M,3))
0024            DY    = Y(NAM(M,1))-Y(NAM(M,3))
0025            GO TO 30
0026         20 DX    = X(NAM(M,2))-X(NAM(M,1))
0027            DY    = Y(NAM(M,2))-Y(NAM(M,1))
0028         30 FX    = FX+XX(M)*DY-XY(M)*DX
0029            FY    = FY+XY(M)*DY-YY(M)*DX
0030         90 CONTINUE
0031         95 FX    = FX/2.
0032            FY    = FY/2.
              C.....................................
0033            IF (N.LT.25 .OR. N.GT.29) GO TO 96
0034            FXSUM = FXSUM+FX
0035            FYSUM = FYSUM+FY
0036         96 CONTINUE
              C.....................................
0037            ACCX = FX/GPM(N)
```

```
FORTRAN IV-PLUS V02-04G          11:25:52     29-MAR-78           PAGE 2
MOTION.FTN       /TR:BLOCKS/WR

0038            ACCY = FY/GPM(N)-GRAV
0039            XD(N)= XD(N)+ACCX*DT
0040            YD(N)= YD(N)+ACCY*DT
0041          100 CONTINUE
              C-------CONTACT AND BOUNDARY CONDITIONS --------------------
0042            IF (IFLAG .NE. 1) CALL INTRAC
0043            CALL BOUNDY
0044            IF (IFLAG .NE. 1) CALL INTRA1
0045            CALL BOUNDY
              C-------NEW COORDINATES ---------------------------
0046            DO 110 N=1,NMAX
0047            XN    = X(N)+XD(N)*DT
0048            YN    = Y(N)+YD(N)*DT
0049            XNPH(N) = 0.5*(X(N)+XN)
0050            YNPH(N) = 0.5*(Y(N)+YN)
0051            X(N) = XN
0052            Y(N) = YN
0053          110 CONTINUE
0054            RETURN
0055            END


PROGRAM SECTIONS

 NAME        SIZE                    ATTRIBUTES

$CODE1   001250    340              RW,I,CON,LCL
$IDATA   000002      1              RW,D,CON,LCL
$VARS    000046     19              RW,D,CON,LCL
$TEMPS   000004      2              RW,D,CON,LCL
CGRID    016374   3710              RW,D,OVR,GBL
COTHR    000110     36              RW,D,OVR,GBL
CPLOT    004570   1212              RW,D,OVR,GBL
CIMPC    001064    282              RW,D,OVR,GBL
SHEAR    000010      4              RW,D,OVR,GBL

TOTAL SPACE ALLOCATED = 025714  5606

,LIST.LST/LI:1/-SP=MOTION
```

```
FORTRAN IV-PLUS V02-04G          11:26:06     29-MAR-78           PAGE 1
BOUNDY.FTN       /TR:BLOCKS/WR

0001            SUBROUTINE BOUNDY
        C*****************************************************************
        C                                                                *
        C       BOUNDARY CONDITIONS                                      *
        C                                                                *
        C*****************************************************************
0002            INCLUDE 'COMMON.FTN'
0003 *          REAL LAME1,LAME2
0004 *          COMMON /CGRID/ X(70),Y(70),XD(70),YD(70),GPM(70),
     *         .   XNPH(70),YNPH(70),
     *         .   XX(70),YY(70),XY(70),DXXFV(70),DYYFV(70),DXYFV(70),
     *         .   AM(70),RHO(70),LAME1(70),LAME2(70),COHES(70),TANPH1(70),
     *         .   TANPS1(70),MAN(70,10),NAM(70,3)
0005 *          COMMON /COTHR/ DT,NITER,ARAT,GRAV,ITER,NMAX,MMAX,IMAX,TIME,
     *         .   ZERO,NPR1,LCONT,IFLAG,KTOT,NBUF,NPL,NPLOT,ITPL(10),
     *         .   SCALEX,SCALEY
0006 *          COMMON /CPLOT/ BUF1(202),BUF2(202),BUF3(202)
0007 *          COMMON /CIMPC/ IMP(10),IMD(10),IFL(10),IMZ(10),IFG(10),
     *         .   IFR(10),N2M(10),N3M(10),MMM(10),N4M(10),TOLC(10),
     *         .   STNL,STNU,STSH,XMU,TOL,BDT,BDT1,CON1,CON2,TFRAC,
     *         .   FN(10),FS(10),DRN(10),DRS(10),DRST(10),XNC(10),YNC(10),DN
        C-------BOTTOM BOUNDARY FIXED ----------------------------
0008            XD(3) = 0
0009            YD(1) = 0
0010            YD(2) = 0
0011            YD(3) = 0
0012            YD(4) = 0
0013            YD(5) = 0
        C-------TOP BOUNDARY RATHER PECULIAR, I'D SAY -------------
0014            FREDA = 0
0015            FREDB = 0
0016            IF (ITER .LT. 100) FREDB=-0.1
0017            IF (ITER .GT. 1000) FREDA=-0.1
0018            DO 10 N=25,29
0019            XD(N) = FREDA
0020         10 YD(N) = FREDB
0021            RETURN
0022            END


PROGRAM SECTIONS

 NAME        SIZE                    ATTRIBUTES

$CODE1   000166     59          RW,I,CON,LCL
$PDATA   000004      2          RW,D,CON,LCL
$VARS    000012      5          RW,D,CON,LCL
CGRID    016374   3710          RW,D,OVR,GBL
COTHR    000110     36          RW,D,OVR,GBL
CPLOT    004570   1212          RW,D,OVR,GBL
CIMPC    001064    282          RW,D,OVR,GBL


TOTAL SPACE ALLOCATED = 024564  5306

,LIST.LST/L1:1/-SP=BOUNDY
```

```
FORTRAN IV-PLUS V02-04G          11:26:12     29-MAR-78            PAGE 1
INTRAC.FTN       /TR:BLOCKS/WR

0001           SUBROUTINE INTRAC
0002           INCLUDE 'COMMON.FTN'
0003 *         REAL LAME1,LAME2
0004 *         COMMON /CGRID/ X(70),Y(70),XD(70),YD(70),GPM(70),
     *        .   XNPH(70),YNPH(70),
     *        .   XX(70),YY(70),XY(70),DXXFV(70),DYYFV(70),DXYFV(70),
     *        .   AM(70),RHO(70),LAME1(70),LAME2(70),COHES(70),TANPHI(70),
     *        .   TANPS1(70),MAN(70,10),NAM(70,3)
0005 *         COMMON /COTHR/ DT,NITER,ARAT,GRAV,ITER,NMAX,MMAX,IMAX,TIME,
     *        .   ZERO,NPRI,LCONT,IFLAG,KTOT,NBUF,NPL,NPLOT,ITPL(10),
     *        .   SCALEX,SCALEY
0006 *         COMMON /CPLOT/ BUF1(202),BUF2(202),BUF3(202)
0007 *         COMMON /C1MPC/ IMP(10),IMD(10),IFL(10),IMZ(10),IFG(10),
     *        .   IFR(10),N2M(10),N3M(10),MMM(10),N4M(10),TOLC(10),
     *        .   STNL,STNU,STSH,XMU,TOL,BDT,BDT1,CON1,CON2,IFRAC,
     *        .   FN(10),FS(10),DRN(10),DRS(10),DRST(10),XNC(10),YNC(10),DN
0008           COMMON /INTRA/ FLAG(10)
0009           AREA3(X1,Y1,X2,Y2,X3,Y3) = -0.50*((Y1+Y2)*(X2-X1)+
              .        (Y3+Y2)*(X3-X2)+(Y1+Y3)*(X1-X3))
           C-------VELOCITY CORRECTION DUE TO CONTACT FORCES ----------
           C-------------------------------------------------------
0010           DO 50 L=1,LCONT
0011           IF (IFG(L) .NE. 1) GO TO 50
0012           IF (IFR(L) .NE. 0) GO TO 50
0013           N1     = IMP(L)
0014           NW     = IMD(L)
0015           N2     = N2M(L)
0016           N3     = N3M(L)
           C
0017           IF (IFL(L) .EQ. 0) GO TO 30
0018           IF (DRST(L)) 20,30,10
0019        10 N2     = NW
0020           GO TO 30
0021        20 N3     = NW
           C
0022        30 XND    = YNPH(N2)-YNPH(N3)
0023           YND    = XNPH(N3)-XNPH(N2)
0024           FRED   = SQRT(XND*XND+YND*YND)
0025           XND    = XND/FRED
0026           YND    = YND/FRED
           C
0027           C11    = YND*YNC(L)+XND*XNC(L)
0028           C12    = YND*XNC(L)-XND*YNC(L)
0029           FNP    = C11*FN(L)+C12*FS(L)
0030           FSP    =-C12*FN(L)+C11*FS(L)
           C
0031           BDT2   = BDT1
           C       IF (ABS(FSP) .GE. 0.98*XMU*ABS(FNP)) BDT2=1.
0032           FNP    = FNP*BDT1
0033           FSP    = FSP*BDT2
0034           XD(N1)= XD(N1)+(FNP*XND-FSP*YND)*DT/GPM(N1)
0035           YD(N1)= YD(N1)+(FNP*YND+FSP*XND)*DT/GPM(N1)
0036           XD(NW)= XD(NW)-(FNP*XND-FSP*YND)*DT/GPM(NW)
0037           YD(NW)= YD(NW)-(FNP*YND+FSP*XND)*DT/GPM(NW)
0038           XNC(L)= XND
0039           YNC(L)= YND
```

```
FORTRAN IV-PLUS V02-04G          11:26:12     29-MAR-78          PAGE 2
INTRAC.FTN       /TR:BLOCKS/WR

0040            FN(L) = FNP/BDT1
0041            FS(L) = FSP/BDT2
0042         50 CONTINUE
0043            RETURN
0044            END


PROGRAM SECTIONS

 NAME        SIZE                    ATTRIBUTES

$CODE1   001054    278               RW,I,CON,LCL
$VARS    000052     21               RW,D,CON,LCL
$TEMPS   000002      1               RW,D,CON,LCL
CGRID    016374   3710               RW,D,OVR,GBL
COTHR    000110     36               RW,D,OVR,GBL
CPLOT    004570   1212               RW,D,OVR,GBL
CIMPC    001064    282               RW,D,OVR,GBL
INTRA    000050     20               RW,D,OVR,GBL


TOTAL SPACE ALLOCATED = 025560  5560

,LIST.LST/LI:1/-SP=INTRAC
```

```
FORTRAN IV-PLUS V02-04G          11:26:23     29-MAR-78          PAGE 1
INTRA1.FTN       /TR:BLOCKS/WR

0001           SUBROUTINE INTRA1
               C*****************************************************************
               C                                                                *
               C        REZONING CAPABILITIES                                   *
               C                                                                *
               C*****************************************************************
0002           INCLUDE 'COMMON.FTN'
0003 *         REAL LAME1,LAME2
0004 *         COMMON /CGRID/ X(70),Y(70),XD(70),YD(70),GPM(70),
     *        .   XNPH(70),YNPH(70),
     *        .   XX(70),YY(70),XY(70),DXXFV(70),DYYFV(70),DXYFV(70),
     *        .   AM(70),RHO(70),LAME1(70),LAME2(70),COHES(70),TANPH1(70),
     *        .   TANPSI(70),MAN(70,10),NAM(70,3)
0005 *         COMMON /COTHR/ DT,NITER,AKAT,GRAV,ITER,NMAX,MMAX,IMAX,TIME,
     *        .   ZERO,NPRI,LCONT,IFLAG,KTOT,NBUF,NPL,NPLOT,ITPL(10),
     *        .   SCALEX,SCALEY
0006 *         COMMON /CPLOT/ BUF1(202),BUF2(202),BUF3(202)
0007 *         COMMON /CIMPC/ IMP(10),IMD(10),IFL(10),IMZ(10),IFG(10),
     *        .   IFR(10),N2M(10),N3M(10),MMM(10),N4M(10),TOLC(10),
     *        .   STNL,STNU,STSH,XMU,TOL,BDT,BDT1,CON1,CON2,TFRAC,
     *        .   FN(10),FS(10),DRN(10),DRS(10),DRST(10),XNC(10),YNC(10),DN
0008           COMMON /INTRA/ FLAG(10)
0009           COMMON/C11/N1,NW,GPM1,GPMW,XND,YND,VNN1,VNNW,VRN,VRS,DRNPH,DRSPH,L
0010           AREA3(X1,Y1,X2,Y2,X3,Y3) = -0.5*((Y1+Y2)*(X2-X1)+
              .        (Y3+Y2)*(X3-X2)+(Y1+Y3)*(X1-X3))
               C-------COMPUTE FORCES FROM DISPLACEMENTS ------------------
               C-----------------------------------------------------------
0011           DO 500 L=1,LCONT
0012           IF (IFG(L) .NE. 1) GO TO 500
0013           N1     = IMP(L)
0014           NW     = IMD(L)
0015           GPM1   = GPM(N1)
0016           GPMW   = GPM(NW)
               C-------RELATIVE MOVEMENTS ---------------------------------
0017           XND    = XNC(L)
0018           YND    = YNC(L)
0019           VNN1   = XD(N1)*XND+YD(N1)*YND
0020           VNNW   = XD(NW)*XND+YD(NW)*YND
0021           VRN    = VNNW-VNN1
0022           VSN1   =-XD(N1)*YND+YD(N1)*XND
0023           VSNW   =-XD(NW)*YND+YD(NW)*XND
0024           VRS    = VSNW-VSN1
0025           DRNN   = DRN(L)+VRN*DT
0026           DRNPH  = 0.5*(DRN(L)+DRNN)
0027           DRN(L) = DRNN
0028           DRSS   = DRS(L)+VRS*DT
0029           DRSPH  = 0.5*(DRS(L)+DRSS)
0030           DRS(L) = DRSS
0031           DRST(L)= DRST(L)+VRS*DT
               C-------BROKEN CONTACT -------------------------------------
0032           IF (DRNPH .GE. -0.1) GO TO 10
               CCCCC PRINT 4000,L,ITER
0033            4000 FORMAT(' BRE'I3,I5)
0034           RTOLC  = TOLC(L)-DRST(L)
               CCCCC PRINT 2728,TOLC(L),DRST(L),RTOLC
0035            2728 FORMAT(' TOLC,DRST,RTOLC'1P3E12.4)
```

```
FORTRAN IV-PLUS V02-04G          11:26:23     29-MAR-78          PAGE 2
INTRA1.FTN       /TR:BLOCKS/WR

0036            IFG(L) = 0
0037            FN(L)  = 0
0038            FS(L)  = 0
0039            DRN(L) = 0
0040            DRS(L) = 0
0041            DRST(L)= 0
          C-------FOR REDUNDANT CONTACTS -------------
0042            DO 5 LC=1,LCONT
0043            IF (IFG(LC) .EQ.  0) GO TO 5
0044            IF (IMP(LC) .NE. NW) GO TO 5
0045            IFR(LC) = 0
0046            GO TO 500
0047          5 CONTINUE
          C
0048            IF (IFL(L) .EQ. 1) GO TO 215
0049            GO TO 500
          C-------CALL APPROPRIATE INTERACTION ROUTINE ---------------
0050         10 IF (IFR(L) .EQ. 1) GO TO 95
0051            CALL POINT
          C
          C-------TICKLE REZONING ----------------------------------
          C-------IS IT NEEDED ----------------------
0052         50 IF (ABS(DRST(L)) .LT. TOL) GO TO 500
0053            IF (IFL(L) .NE. 0) GO TO 95
          C-------DO IT ----------------------------
          CCCCC PRINT 999,L,ITER,DRST(1),DRST(2),TOLC(1),TOLC(2)
0054        999 FORMAT (' TIC'I3,I5,1P4E12.4)
0055            N2      = N2M(L)
0056            N3      = N3M(L)
0057            N4      = N4M(L)
0058            MW      = IMZ(L)
0059            MM      = MMM(L)
0060            DRST(L)= 0
          C
0061            AREAP   = AREA3(X(N1),Y(N1),X(NW),Y(NW),X(N4),Y(N4))
0062            IF (AREAP .GT. 0) GO TO 53
0063            NN      = N2
0064            M2      = MM
0065            M3      = MW
0066            FLAG(L)= 1.
0067            GO TO 55
0068         53 NN      = N3
0069            M2      = MW
0070            M3      = MM
0071            FLAG(L)=-1.
          C
0072         55 AREA    = ABS(AREA3(X(NN),Y(NN),X(NW),Y(NW),X(N4),Y(N4)))
0073            TMASS   = AM(M2)*AREAP/AREA
0074            TMAS3   = TMASS/3.
0075            GPM(N2)= GPM(N2)+TMAS3
0076            GPM(N3)= GPM(N3)-TMAS3
0077            AM(MM)  = AM(MM)+TMASS
0078            AM(MW)  = AM(MW)-TMASS
          C
0079            X(NW)   = X(N1)-DRN(L)*XND-DRS(L)*YND
0080            Y(NW)   = Y(N1)-DRN(L)*YND+DRS(L)*XND
```

```
FORTRAN IV-PLUS V02-04G          11:26:23     29-MAR-78          PAGE 3
INTRA1.FTN       /TR:BLOCKS/WR

0081            XD(NW) = XD(NW)+IMAS3*(XD(N3)-XD(N2))/GPMW
0082            YD(NW) = YD(NW)+IMAS3*(YD(N3)-YD(N2))/GPMW
        C
0083            XND    = YNPH(NN)-YNPH(N4)
0084            YND    = XNPH(N4)-XNPH(NN)
0085            FRED   = XND*XND+YND*YND
        C
        C       GO TO 56
0086            DX = X(N3)-X(N2)
0087            DY = Y(N3)-Y(N2)
0088            DA = X(NW)-X(N2)
0089            DB = Y(NW)-Y(N2)
0090            FR = 0.05*(DX*DX+DY*DY)
0091            IF (DB*DX-DA*DY .LT. FR) GO TO 56
        CCCCC   PRINT 4000,L,ITER
0092            IFG(L) = 0
0093            FN(L)  = 0
0094            FS(L)  = 0
0095            DRN(L) = 0
0096            DRS(L) = 0
0097            DRST(L)= 0
0098            GO TO 500
0099         56 CONTINUE
        C-------DELETION OF TRIANGLES ------------------------------
        C-------IS IT NEEDED ----------------------
0100            IF (2.*AREA/FRED .GE. ARAT) GO TO 500
        C-------DO IT ------------------------------
        CCCCC   PRINT 1001, L,ITER,DRST(L),TOLC(L)
0101     1001   FORMAT(' DEL'I3,I5,1P2E12.4)
        C-------LOCATE SIDE ZONE AND NODE
0102            NWT    = NW
0103            MS     = M2
        C-------LOCATE NODE
0104         64 DO 61 I=1,3
0105            N6T    = NAM(MS,I)
0106            IF (N6T .NE. NN .AND. N6T .NE. NWT) GO TO 62
0107         61 CONTINUE
0108         62 N6     = N6T
        C-------LOCATE ZONE
0109            DO 631 I=1,IMAX
0110            MST    = MAN(NN,I)
0111            IF (MST .EQ. MS .OR. MST .EQ. 0) GO TO 631
0112            DO 63 J=1,IMAX
0113            IF (MST .NE. MAN(N6,J)) GO TO 63
0114            MS     = MST
0115            NWT    = N6
0116            GO TO 64
0117         63 CONTINUE
0118        631 CONTINUE
0119            IF (MS .EQ. M2) MS=M3
        C-------ADJUST VELOCITIES
0120            DENOM  = GPMW+GPM(NN)-TMAS3
0121            NO     = N2+N3-NN
0122            XD(NN) = (GPMW*XD(NW)+GPM(NN)*XD(NN)-TMAS3*XD(NO))/DENOM
0123            YD(NN) = (GPMW*YD(NW)+GPM(NN)*YD(NN)-TMAS3*YD(NO))/DENOM
0124            DRST(L)= SQRT((X(NW)-X(N2))**2+(Y(NW)-Y(N2))**2)*FLAG(L)
```

```
FORTRAN IV-PLUS V02-04G          11:26:23     29-MAR-78          PAGE 4
INTRA1.FTN       /TR:BLOCKS/WR

0125            TOLC(L)= 1.5*ABS(DRST(L))
      C-------ADJUST MASSES
0126            TMAS3  = AM(M2)/3.
0127            GPM(NN)= GPM(NN)+GPMw-TMAS3
0128            GPM(NO)= GPM(NO)+TMAS3
0129            GPM(Nw)= 0
0130            AM(M3) = AM(M3)+AM(M2)
0131            AM(M2) = 0
      C-----ADJUST LINKS
0132            IF (FLAG(L) .GT. 0) N2M(L)=N6
0133            IF (FLAG(L) .LT. 0) N3M(L)=N6
0134            DO 69 I=1,LCONT
0135            IF (N2M(I) .EQ. Nw) N2M(I)=N2
0136            IF (N3M(I) .EQ. Nw) N3M(I)=N3
0137         69 CONTINUE
      C
0138            DO 70 I=1,IMAX
0139            IF (MAN(NN,I) .NE. M2) GO TO 70
0140            MAN(NN,I) = M3
0141            GO TO 72
0142         70 CONTINUE
      C
0143         72 DO 75 I=1,IMAX
0144            IF (MAN(N4,I) .NE. M2) GO TO 75
0145            MAN(N4,I) = 0
0146            GO TO 80
0147         75 CONTINUE
      C
0148         80 DO 90 I=1,3
0149            IF (NAM(Mw,I) .NE. Nw) GO TO 90
0150            NAM(Mw,I) = NN
0151            GO TO 92
0152         90 CONTINUE
      C
0153         92 IMD(L) = NN
0154            IFL(L) = 1
0155            IF (FLAG(L) .GT. 0) MMM(L)=MS
0156            IF (FLAG(L) .LT. 0) IM2(L)=MS
      C-------REDUNDANT CONTACTS
0157            DO 57 LC=1,LCONT
0158            IF (IFG(LC) .EQ.  0) GO TO 57
0159            IF (IMP(LC) .NE. NN) GO TO 57
0160            IF (IMD(LC) .NE. N1) GO TO 57
0161            IF (IFR(LC) .EQ.  1) GO TO 57
0162            IFR(L) = 1
      CCCCC PRINT 1002, L,ITER
0163       1002 FORMAT (' RED'I3,I5)
0164            GO TO 58
0165         57 CONTINUE
      C
0166         58 CONTINUE
      CCCCC PRINT 2003
0167            DO 93 N=1,NMAX
      CCCCC PRINT 3003, N,(MAN(N,I),I=1,IMAX)
0168         93 CONTINUE
      CCCCC PRINT 2004, (M,(NAM(M,J),J=1,3),AM(M),M=1,MMAX)
```

```
FORTRAN IV-PLUS V02-04G          11:26:23     29-MAR-78           PAGE 5
INTRA1.FTN       /TR:BLOCKS/WR

      CCCCC PRINT 2222, (K,IFG(K),IFL(K),IFR(K),IMP(K),IMD(K),N2M(K),N3M(K),
      CCCCC.             MMM(K),IMZ(K),K=1,LCONT)
0169   2222 FORMAT (///' CONTACT LINKS'
     .          /'    L   IFG  IFL  IFR  IMP  IMD  N2M  N3M  MMM  IMZ'
     .          /(1X,10I5))
0170   2003 FORMAT (///' ZONES SURROUNDING EACH NODE'//5X,'N',5X,'1',
     .         5X,'2',5X,'3',5X,'4',5X,'5',5X,'6',5X,'7',5X,'8',
     .         5X,'9',5X,'10')
0171   3003 FORMAT (1X,1116)
0172   2004 FORMAT (///' NODES SURROUNDING EACH ZONE'//5X,'M',5X,'1',
     .         5X,'2',5X,'3',5X,'MASS'/(416,1PE12.4))
0173        GO TO 500
      C-------RE-CREATION OF A DELETED NODE ----------------------
      C-------IS IT NEEDED ----------------------
0174     95 IF (ABS(DRST(L)) .LE. TOLC(L)) GO TO 500
      C-------DO IT -----------------------------
      CCCCC PRINT 1000, L
0175   1000 FORMAT(' CRE'I3)
0176        IFR(L) = 0
0177        IF (DRST(L) .GT. 0) N2M(L)=NW
0178        IF (DRST(L) .LT. 0) N3M(L)=NW
0179        IF (IMZ(L).EQ.0 .OR. MMM(L).EQ.0) INOT=1
      C
0180        IF (IMZ(L) .NE. MMM(L)) GO TO 197
0181        IFG(L) = 0
0182        GO TO 198
      C
0183     197 CALL CREATE(L)
0184     198 IF (INOT .EQ. 0) GO TO 199
0185        INOT = 0
0186        GO TO 500
      C-------MAKE REVERSE CONTACT NON-REDUNDANT
0187     199 DO 210 LC=1,LCONT
0188        IF (IFG(LC) .NE.  1) GO TO 210
0189        IF (IMP(LC) .NE. NW) GO TO 210
0190        IFR(LC) = 0
0191        GO TO 500
0192     210 CONTINUE
      C-------CREATE REVERSE CONTACT -------------
      C-------FIND CONTACT
0193     215 DO 220 LC=1,LCONT
0194        IF (IFG(LC) .EQ. 1) GO TO 220
0195        LL = LC
0196        GO TO 230
0197     220 CONTINUE
0198        LCONT = LCONT+1
0199        LL = LCONT
      C-------FIND A ZONE
0200     230 DO 240 I=1,IMAX
0201        IF (MAN(N1,I) .EQ. 0) GO TO 240
0202        MT = MAN(N1,I)
0203        GO TO 250
0204     240 CONTINUE
      C-------FIND SECOND NODE
0205     250 DO 260 J=1,3
0206        IF (NAM(MT,J) .EQ. N1) GO TO 270
```

```
FORTRAN IV-PLUS V02-04G          11:26:23     29-MAR-78          PAGE 6
INTRA1.FTN      /TR:BLOCKS/WR

0207       260 CONTINUE
0208       270 J   = J+IFIX(FLAG(L))
0209           IF (J .EQ. 0) J=3
0210           IF (J .EQ. 4) J=1
0211           NT = NAM(MT,J)
       C-------ITERATE TO BOUNDARY
0212         272 DO 280 I=1,IMAX
0213           M   = MAN(N1,I)
0214           IF (M .EQ. 0 .OR. M .EQ. MT) GO TO 280
0215           DO 275 J=1,IMAX
0216           IF (M .EQ. MAN(NT,J)) GO TO 290
0217         275 CONTINUE
0218         280 CONTINUE
0219           GO TO 310
       C
0220         290 MT = M
0221           DO 300 I= 1,3
0222           N   = NAM(MT,I)
0223           IF (N .EQ. N1 .OR. N .EQ. NT) GO TO 300
0224           NT = N
0225           GO TO 272
0226         300 CONTINUE
       C-------CREATE NEW CONTACT
0227         310 IMP(LC) = NW
       C...CHECK......
0228           N2M(LC) = NT
0229           N3M(LC) = N1
       C..............
0230           IFL(LC) = 0
0231           IFG(LC) = 1
0232           IFR(LC) = 0
       C
0233           IF (RTOLC .EQ. 0) GO TO 320
0234           IMD(LC) = N1
0235           MMM(LC) = MT
0236           IMZ(LC) = 0
0237           IFL(LC) = 1
0238           TOLC(LC)= RTOLC
0239           RTOLC   = 0
0240           FLAG(LC)= FLAG(L)
0241           DRST(LC)= 0
0242           XND     = YNPH(N2M(LC))-YNPH(N3M(LC))
0243           YND     = XNPH(N3M(LC))-XNPH(N2M(LC))
0244           FRED    = SQRT(XND*XND+YND*YND)
0245           XNC(LC) = XND/FRED
0246           YNC(LC) = YND/FRED
0247           GO TO 500
       C
0248         320 CALL CREATE(LC)
       C
0249         500 CONTINUE
0250           RETURN
0251           END


PROGRAM SECTIONS
```

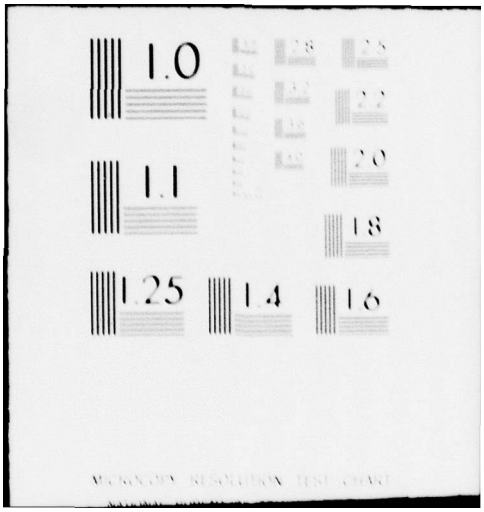
AD-A061 658 DAMES AND MOORE LOS ANGELES CA F/G 8/7
COMPUTER MODELING OF JOINTED ROCK MASSES.(U)
AUG 78 T MAINI, P CUNDALL, J MARTI DACA39-77-C-0004
UNCLASSIFIED WES-TR-N-78-4 NL

5 OF 6
AD
A061 658



END
DATE FILMED
2-79
DDC

CONT


1.0
1.1
1.25
1.4
1.6
2.8
2.5
2.2
2.0
1.8

```
FORTRAN IV-PLUS V02-04G          11:27:34     29-MAR-78           PAGE 1
POINT.FTN        /TR:BLOCKS/WR

0001           SUBROUTINE POINT
       C*****************************************************************
       C                                                                *
       C        CONSTITUTIVE RELATIONS FOR CORNER-EDGE CONTACTS         *
       C                                                                *
       C*****************************************************************
0002           INCLUDE 'COMMON.FTN'
0003 *         REAL LAME1,LAME2
0004 *         COMMON /CGRID/ X(70),Y(70),XD(70),YD(70),GPM(70),
     *        .   XNPH(70),YNPH(70),
     *        .   XX(70),YY(70),XY(70),DXXFV(70),DYYFV(70),DXYFV(70),
     *        .   AM(70),RHO(70),LAME1(70),LAME2(70),COHES(70),TANPHI(70),
     *        .   TANPSI(70),MAN(70,10),NAM(70,3)
0005 *         COMMON /COTHR/ DT,NITER,ARAT,GRAV,ITER,NMAX,MMAX,IMAX,TIME,
     *        .   ZERO,NPRI,LCONT,IFLAG,KTOT,NBUF,NPL,NPLOT,ITPL(10),
     *        .   SCALEX,SCALEY
0006 *         COMMON /CPLOT/ BUF1(202),BUF2(202),BUF3(202)
0007 *         COMMON /CIMPC/ IMP(10),IMD(10),IFL(10),IMZ(10),IFG(10),
     *        .   IFR(10),N2M(10),N3M(10),MMM(10),N4M(10),TOLC(10),
     *        .   STNL,STNU,STSH,XMU,TOL,BDT,BDT1,CON1,CON2,TFRAC,
     *        .   FN(10),FS(10),DRN(10),DRS(10),DRST(10),XNC(10),YNC(10),DN
0008           COMMON/CI1/N1,NW,GPM1,GPMW,XND,YND,VNN1,VNNW,VRN,VRS,DRNPH,DRSPH,L
       C
       C-------NORMAL COMPONENT -----------------------------------
       C-------RIGID CONTACT ----------------------
0009           IF (DRNPH .LE. DN) GO TO 20
0010           DVNN1  = (VNNW-VNN1-2.*(DN-DRN(L))/DT)*GPMW/(GPM1+GPMW)
0011           DVNNW  = DVNN1*GPM1/GPMW
0012           FN(L)  = DN*STNL+GPM1*DVNN1/DT
0013           XD(N1) = XD(N1)+DVNN1*XND
0014           YD(N1) = YD(N1)+DVNN1*YND
0015           XD(NW) = XD(NW)+DVNNW*XND
0016           YD(NW) = YD(NW)+DVNNW*YND
0017           DRN(L) = DN
0018           DRNPH  = DN
0019           GO TO 30
       C-------FLEXIBLE CONTACT -------------------
0020        20 FN(L)  = AMIN1(FN(L)+VRN*DT*STNU,DRNPH*STNL)
0021           FN(L)  = AMAX1(FN(L),ZERO)
       C
       C-------SHEAR COMPONENT -----------------------------------
0022        30 DSST   = FN(L)*XMU
0023           DS     = DSST/STSH
       C-------NO SLIDING ------------------------
0024           IF (ABS(DRSPH) .GT. DS) GO TO 40
0025           FS(L)  = STSH*DRSPH
0026           RETURN
       C-------SLIDING ---------------------------
0027        40 DRS(L) = SIGN(DS,DRS(L))
0028           DRSPH  = DRS(L)
0029           FS(L)  = SIGN(DSST,DRS(L))
       C
0030           RETURN
0031           END
```

```
FORTRAN IV-PLUS V02-04G          11:27:45     29-MAR-78           PAGE 1
CREATE.FTN       /TR:BLOCKS/WR

0001           SUBROUTINE CREATE(L)
          C*****************************************************************
          C                                                                *
          C        CREATION OF A NEW GRID POINT UPON CONTACT               *
          C                                                                *
          C*****************************************************************
0002           INCLUDE 'COMMON.FTN'
0003  *        REAL LAME1,LAME2
0004  *        COMMON /CGRID/ X(70),Y(70),XD(70),YD(70),GPM(70),
      *       .   XNPH(70),YNPH(70),
      *       .   XX(70),YY(70),XY(70),DXXFV(70),DYYFV(70),DXYFV(70),
      *       .   AM(70),RHO(70),LAME1(70),LAME2(70),COMES(70),TANPHI(70),
      *       .   TANPSI(70),MAN(70,10),NAM(70,3)
0005  *        COMMON /COTHR/ DT,NITER,ANAT,GRAV,ITER,NMAX,MMAX,IMAX,TIME,
      *       .   ZERO,NPRI,LCONT,IFLAG,NTOT,NBUF,NPL,NPLOT,ITPL(10),
      *       .   SCALEX,SCALEY
0006  *        COMMON /CPLOT/ BUF1(202),BUF2(202),BUF3(202)
0007  *        COMMON /CIMPC/ IMP(10),IMD(10),IFL(10),IMZ(10),IFG(10),
      *       .   IFR(10),N2M(10),N3M(10),MMM(10),N4M(10),TOLC(10),
      *       .   STNL,STNU,STSH,XMU,TOL,BDT,BDT1,CON1,CON2,TFRAC,
      *       .   FN(10),FS(10),DRN(10),DRS(10),DRST(10),XNC(10),YNC(10),DN
          C-------NEW NODE AND ZONE -------------------------------
0008           DO 10 N=1,NMAX
0009           IF (GPM(N) .NE. 0) GO TO 10
0010           IMD(L)= N
0011           GO TO 20
0012        10 CONTINUE
0013           NMAX  = NMAX+1
0014           IMD(L)= NMAX
0015        20 NW    = IMD(L)
          C
0016           DO 30 M=1,MMAX
0017           IF (AM(M) .NE. 0) GO TO 30
0018           IMZ(L)= M
0019           GO TO 40
0020        30 CONTINUE
0021           MMAX  = MMAX+1
0022           IMZ(L)= MMAX
0023        40 MW    = IMZ(L)
          C-------COORDINATES AND VELOCITIES ----------------------
0024           N1    = IMP(L)
0025           N2    = N2M(L)
0026           N3    = N3M(L)
0027           X(NW) = X(N1)-DRN(L)*XNC(L)-DRS(L)*YNC(L)
0028           Y(NW) = Y(N1)-DRN(L)*YNC(L)+DRS(L)*XNC(L)
0029           XNPH(NW) = X(N1)
0030           YNPH(NW) = Y(N1)
0031           D1    = SQRT((X(N2)-X(NW))**2+(Y(N2)-Y(NW))**2)
0032           D2    = SQRT((X(N3)-X(NW))**2+(Y(N3)-Y(NW))**2)
0033           FRED  = D1/(D1+D2)
0034           BILL  = 1.-FRED
0035           XD(NW)= FRED*XD(N2)+BILL*XD(N3)
0036           YD(NW)= FRED*YD(N2)+BILL*YD(N3)
0037           DRST(L) = 0
          C-------DETERMINATION OF ZONE IMPACTED ------------------
0038           DO 42 I=1,IMAX
```

```
FORTRAN IV-PLUS V02-04G          11:27:45     29-MAR-78          PAGE 2
CREATE.FTN       /FR:BLOCKS/WR

0039            NIN     = MAN(N2,I)
0040            IF (NIN .EQ. 0) GO TO 42
0041            DO 42 J=1,IMAX
0042            IF (NIN .NE. MAN(N3,J)) GO TO 42
0043            MM      = NIN
0044            MMM(L) = NIN
0045            GO TO 43
0046         42 CONTINUE
          C--------REASSIGNMENT OF MASS AND OTHER PROPERTIES -------
0047         43 TOL     = TFRAC*(D1+D2)
0048            AMM     = AM(MM)
0049            GPM(Nw)= AMM/3.
0050            GPM(N2)= GPM(N2)-AMM*BILL/3.
0051            GPM(N3)= GPM(N3)-AMM*FRED/3.
0052            AM(MM) = AMM*FRED
0053            AM(Nw) = AMM*BILL
0054            LAME1(Nw) = LAME1(MM)
0055            LAME2(Nw) = LAME2(MM)
0056            COHES(Nw) = COHES(MM)
0057            TANPH1(Nw)= TANPH1(MM)
0058            TANPSI(Nw)= TANPSI(MM)
          C--------ESTABLISHMENT OF NEW LINKS ----------------------
0059            DO 44 J=1,3
0060            N4      = NAM(MM,J)
0061            IF (N4 .NE. N2 .AND. N4 .NE. N3) GO TO 45
0062         44 CONTINUE
0063         45 N4M(L) = N4
          C
0064            IFL(L)     = 0
0065            NAM(MM,1) = N4
0066            NAM(MM,2) = Nw
0067            NAM(MM,3) = N2
0068            NAM(Nw,1) = N4
0069            NAM(Nw,2) = N3
0070            NAM(Nw,3) = Nw
0071            DO 50 I=1,IMAX
0072            IF (MM .EQ. MAN(N3,I)) GO TO 60
0073         50 CONTINUE
0074            STOP
0075         60 MAN(N3,I) = Nw
0076            DO 70 I=1,IMAX
0077            IF (MAN(N4,I) .EQ. 0) GO TO 80
0078         70 CONTINUE
          CCCCC PRINT 2100
0079         80 MAN(N4,I) = Nw
0080            DO 90 I=1,IMAX
0081         90 MAN(Nw,I) = 0
0082            MAN(Nw,1) = MM
0083            MAN(Nw,2) = Nw
          C--------CORRECT OLD CONTACTS ----------------------------
0084            DO 100 LC=1,LCONT
0085            IF (IFG(LC) .EQ. 0) GO TO 100
0086            IF (N3 .NE. IMD(LC)) GO TO 92
0087            N2M(LC)    = Nw
0088            MMM(LC)    = IMZ(L)
0089            GO TO 100
```

```
FORTRAN IV-PLUS V02-04G          11:27:45     29-MAR-78           PAGE 3
CREATE.FTN       /IR:BLOCKS/WR

0090        92 IF (N2 .NE. IMD(LC)) GO TO 100
0091           N3M(LC)    = NM
0092           IM2(LC)    = MMM(L)
0093         100 CONTINUE
             C
             CCCCC PRINT 2000, L
             CCCCC PRINT 2003
0094             DO 101 N=1,NMAX
             CCCCC PRINT 3003, N,(MAN(N,I),I=1,IMAX)
0095         101 CONTINUE
             CCCCC PRINT 2004, (N,(NAM(M,J),J=1,3),AM(M),M=1,MMAX)
             CCCCC PRINT 2007, (N,GPM(N),N=1,NMAX)
             CCCCC PRINT 2222, (K,IFG(K),IFL(K),IFR(K),IMP(K),IMD(K),N2M(K),N3M(K),
             CCCCC.            MMM(K),IM2(K),K=1,LCONT)
0096         2222 FORMAT (///' CONTACT LINKS'
                 .          /'    L   IFG  IFL  IFR  IMP  IMD  N2M  N3M  MMM  IM2'
                 .          /(1X,1015))
0097             RETURN
             C
0098         2000 FORMAT (///' **** NEW CONTACT NODE CREATED. CONTACT NO' I4)
0099         2001 FORMAT (///' NODE DATA'//5X,'N',6X,'X',11X,'Y',10X,'XD',
                 .          10X,'YD',/(1X,I5,1P4E12.4))
0100         2002 FORMAT (///' ZONE DATA'//5X,'M',5X,'XX',10X,'YY',10X,'XY',
                 .          10X,'AM',9X,'LAME1'7X,'LAME2'7X,'COHES'7X,'TANPHI'
                 .          6X,'TANPSI'/(I6,1P9E12.4))
0101         2003 FORMAT (///' ZONES SURROUNDING EACH NODE'//5X,'N',5X,'1',
                 .          5X,'2',5X,'3',5X,'4',5X,'5',5X,'6',5X,'7',5X,'8',
                 .          5X,'9',5X,'10')
0102         3003 FORMAT (1X,1116)
0103         2004 FORMAT (///' NODES SURROUNDING EACH ZONE'//5X,'M',5X,'1',
                 .          5X,'2',5X,'3',5X,'MASS'/(4I6,1PE12.4))
0104         2007 FORMAT (///' GRID POINT MASSES'//5X,'N'4X,'MASS'
                 .          /(I6,1PE12.4))
0105         2100 FORMAT ('OND SPACE TO STORE NEW LINK -- JOB ABORTED')
0106             END


PROGRAM SECTIONS

 NAME        SIZE                    ATTRIBUTES

$CODE1   002422   649            RW,I,CON,LCL
$VARS    000056    23            RW,D,CON,LCL
$TEMPS   000022     9            RW,D,CON,LCL
CGRID    016374  3710            RW,D,OVR,GBL
COTHR    000110    36            RW,D,OVR,GBL
CPLOT    004570  1212            RW,D,OVR,GBL
CIMPC    001064   282            RW,D,OVR,GBL


TOTAL SPACE ALLOCATED = 027102  5921

,LIST.LST/LI:1/-SP=CREATE
```

```
FORTRAN IV-PLUS V02-04G          11:28:12     29-MAR-78           PAGE 1
STRESS.FTN       /TR:BLOCKS/WR

0001           SUBROUTINE STRESS
       C*****************************************************************
       C                                                                *
       C        MOMENTUM BALANCE AT CONTACTS                            *
       C                                                                *
       C*****************************************************************
0002           INCLUDE 'COMMON.FTN'
0003 *         REAL LAME1,LAME2
0004 *         COMMON /CGRID/ X(70),Y(70),XD(70),YD(70),GPM(70),
     *        .   XNPH(70),YNPH(70),
     *        .   XX(70),YY(70),XY(70),DXXFV(70),DYYFV(70),DXYFV(70),
     *        .   AM(70),RHO(70),LAME1(70),LAME2(70),COHES(70),TANPHI(70),
     *        .   TANPS1(70),MAN(70,10),NAM(70,3)
0005 *         COMMON /COTHR/ DT,NITER,ARAT,GRAV,ITER,NMAX,MMAX,IMAX,TIME,
     *        .   ZERO,NPRI,LCONT,IFLAG,KTOT,NBUF,NPL,NPLOT,ITPL(10),
     *        .   SCALEX,SCALEY
0006 *         COMMON /CPLOT/ BUF1(202),BUF2(202),BUF3(202)
0007 *         COMMON /CIMPC/ IMP(10),IMD(10),IFL(10),IMZ(10),IFG(10),
     *        .   IFR(10),N2M(10),N3M(10),MMM(10),N4M(10),TOLC(10),
     *        .   STNL,STNU,STSH,XMU,TOL,BDT,BDT1,CON1,CON2,TFRAC,
     *        .   FN(10),FS(10),DRN(10),DRS(10),DRST(10),XNC(10),YNC(10),DN
       C-------MAIN ITERATION LOOP ----------------------------
0008           DO 100 M=1,MMAX
0009           IF (AM(M) .LE. 0) GO TO 100
0010           XNPH2= XNPH(NAM(M,2))-XNPH(NAM(M,1))
0011           XNPH3= XNPH(NAM(M,3))-XNPH(NAM(M,1))
0012           YNPH2= YNPH(NAM(M,2))-YNPH(NAM(M,1))
0013           YNPH3= YNPH(NAM(M,3))-YNPH(NAM(M,1))
0014           DET  = XNPH2*YNPH3-YNPH2*XNPH3
0015           XD2  = XD(NAM(M,2))-XD(NAM(M,1))
0016           XD3  = XD(NAM(M,3))-XD(NAM(M,1))
0017           YD2  = YD(NAM(M,2))-YD(NAM(M,1))
0018           YD3  = YD(NAM(M,3))-YD(NAM(M,1))
       C-------INCREMENTAL STRAINS AND ROTATIONS ----------------
0019           EDXX = (XD2*YNPH3-XD3*YNPH2)/DET
0020           EDYY = (YD3*XNPH2-YD2*XNPH3)/DET
0021           EDXY = 0.5*(XD3*XNPH2-XD2*XNPH3+YD2*YNPH3-YD3*YNPH2)/DET
0022           RDXY = 0.5*(XD3*XNPH2-XD2*XNPH3-YD2*YNPH3+YD3*YNPH2)/DET
0023           EXX  = EDXX*DT
0024           EYY  = EDYY*DT
0025           EXY  = EDXY*DT
0026           RXY  = RDXY*DT
0027           EV   = EXX+EYY
       C-------ELASTIC STRESS INCREMENTS -----------------------
0028           DXXE = LAME1(M)*EV+2.*LAME2(M)*EXX
0029           DYYE = LAME1(M)*EV+2.*LAME2(M)*EYY
0030           DXYE = 2.*LAME2(M)*EXY
       C-------STRESS INCREMENTS ARISING FROM STRESS ROTATION ---
0031           SDIFF= XX(M)-YY(M)
0032           DXXR = -RXY*(SDIFF*RXY+2.*XY(M))
0033           DYYR = -DXXR
0034           DXYR = RXY*(-2.*RXY*XY(M)+SDIFF)
       C-------VISCOUS STRESS INCREMENT -----------------------
0035           DXXV = DXXE*BDT
0036           DYYV = DYYE*BDT
0037           DXYV = DXYE*BDT
```

```
FORTRAN IV-PLUS V02-04G          11:28:12     29-MAR-78           PAGE 2
STRESS.FTN       /TR:BLOCKS/WR

0038            XXM   = XX(M)+DXXE+DXXR-DXXFV(M)
0039            YYM   = YY(M)+DYYE+DYYR-DYYFV(M)
0040            XYM   = XY(M)+DXYE+DXYR-DXYFV(M)
        C-------YIELD CALCULATIONS ------------------------------
        C-------NEW STRESSES ------------------------------------
0041            XX(M) = XXM+DXXV
0042            YY(M) = YYM+DYYV
0043            XY(M) = XYM+DXYV
0044            DXXFV(M) = DXXV
0045            DYYFV(M) = DYYV
0046            DXYFV(M) = DXYV
0047         100 CONTINUE
0048            RETURN
0049            END


PROGRAM SECTIONS

 NAME        SIZE                    ATTRIBUTES

$CODE1   001200    320               RW,I,CON,LCL
$VARS    000176     63               RW,D,CON,LCL
$TEMPS   000002      1               RW,D,CON,LCL
CGRID    016374   3710               RW,D,OVR,GBL
COTHR    000110     36               RW,D,OVR,GBL
CPLOT    004570   1212               RW,D,OVR,GBL
CIMPC    001064    282               RW,D,OVR,GBL


TOTAL SPACE ALLOCATED = 025760  5624

,LIST.LST/LI:1/-SP=STRESS
```

```
FORTRAN IV-PLUS V02-046          11:28:27     29-MAR-78          PAGE 1
OUTPUT.FTN      /IR:BLOCKS/WR

0001            SUBROUTINE OUTPUT
                C*****************************************************************
                C                                                                *
                C       CREATION OF OUTPUT FILES                                 *
                C                                                                *
                C*****************************************************************
0002            INCLUDE 'COMMON.FTN'
0003 *          REAL LAME1,LAME2
0004 *          COMMON /CGRID/ X(70),Y(70),XD(70),YD(70),GPM(70),
     *         .   XNPH(70),YNPH(70),
     *         .   XX(70),YY(70),XY(70),DXXFV(70),DYYFV(70),DXYFV(70),
     *         .   AM(70),RHO(70),LAME1(70),LAME2(70),COHES(70),TANPH1(70),
     *         .   TANPS1(70),MAN(70,10),NAM(70,3)
0005 *          COMMON /COTHR/ DT,NITER,ARAT,GRAV,ITER,NMAX,MMAX,IMAX,TIME,
     *         .   ZERO,NPRI,LCONT,IFLAG,KTOT,NBUF,NPL,NPLOT,ITPL(10),
     *         .   SCALEX,SCALEY
0006 *          COMMON /CPLOT/ BUF1(202),BUF2(202),BUF3(202)
0007 *          COMMON /CIMPC/ IMP(10),IMD(10),IPL(10),IMZ(10),IFG(10),
     *         .   IFR(10),N2M(10),N3M(10),MMM(10),N4M(10),TOLC(10),
     *         .   SINL,SINO,SISH,XMU,TOL,BDT,BDT1,CON1,CON2,TFRAC,
     *         .   FN(10),FS(10),DRN(10),DRS(10),DRSI(10),XNC(10),YNC(10),DN
0008            COMMON /SHEAR/ FXSUM,FYSUM
                C
0009            IF (MOD(ITER,NPRI)) 10,5,10
0010          5 PRINT 2000, ITER,TIME
0011            PRINT 2001, (N,X(N),Y(N),XD(N),YD(N),N=1,NMAX)
0012            PRINT 2002, (M,XX(M),YY(M),XY(M),AM(M),LAME1(M),LAME2(M),
                .            COHES(M),TANPH1(M),TANPS1(M),M=1,MMAX)
0013         10 IF (MOD(ITER,NPLOT)) 20,15,20
0014         15 NPL = NPL+1
0015            BUF1(NPL) = FYSUM
0016            BUF2(NPL) = TIME
0017            BUF3(NPL) = FXSUM
0018         20 DO 30 I=1,10
0019            IF (ITER .EQ. ITPL(I)) CALL PLOTM
0020         30 CONTINUE
0021            RETURN
0022       2000 FORMAT (////' ITERATION NUMBER ='I5,
                .             /' TIME              ='1PE12.4)
0023       2001 FORMAT (//' NODE DATA'//5X,'N',6X,'X',11X,'Y',10X,'XD',
                .          10X,'YD',/(1X,I5,1P4E12.4))
0024       2002 FORMAT (//' ZONE DATA'//5X,'M',5X,'XX',10X,'YY',10X,'XY',
                .          10X,'AM',9X,'LAME1'7X,'LAME2'7X,'COHES'7X,'TANPH1'
                .          6X,'TANPS1'/(I6,1P9E12.4))
0025            END


PROGRAM SECTIONS

 NAME        SIZE                  ATTRIBUTES

$CODE1   000636    207             RW,I,CON,LCL
$IDATA   000330    108             RW,D,CON,LCL
$VARS    000006      3             RW,D,CON,LCL
CGRID    016374   3710             RW,D,OVR,GBL
COTHR    000110     36             RW,D,OVR,GBL
```

```
FORTRAN IV-PLUS V02-04G          11:28:46     29-MAR-78           PAGE 1
INPUT.FTN        /TR:BLOCKS/WR

0001            SUBROUTINE INPUT
              C*****************************************************************
              C                                                                *
              C       GENERATION OF A UNIFORM COLUMNAR MESH                    *
              C                                                                *
              C*****************************************************************
0002            INCLUDE 'COMMON.FTN'
0003  *         REAL LAME1,LAME2
0004  *         COMMON /CGRID/ X(70),Y(70),XD(70),YD(70),GPM(70),
      *        .   XNPH(70),YNPH(70),
      *        .   XX(70),YY(70),XY(70),DXXFV(70),DYYFV(70),DXYFV(70),
      *        .   AM(70),RHO(70),LAME1(70),LAME2(70),COHES(70),TANPHI(70),
      *        .   TANPSI(70),MAN(70,10),NAM(70,3)
0005  *         COMMON /COTHR/ DT,NITER,AKAT,GRAV,ITER,NMAX,MMAX,IMAX,TIME,
      *        .   ZERO,NPRI,LCONT,IFLAG,KTOT,NBUF,NPL,NPLOT,ITPL(10),
      *        .   SCALEX,SCALEY
0006  *         COMMON /CPLOT/ BUF1(202),BUF2(202),BUF3(202)
0007  *         COMMON /CIMPC/ IMP(10),IMD(10),IFL(10),IMZ(10),IFG(10),
      *        .   IFR(10),N2M(10),N3M(10),MMM(10),N4M(10),TOLC(10),
      *        .   SINL,STNU,STSH,XMU,TOL,BDT,BDT1,CON1,CON2,TFRAC,
      *        .   FN(10),FS(10),DRN(10),DRS(10),DRST(10),XNC(10),YNC(10),DN
              C-------ZONE-NODE LINKS ---------------------------------
0008            NMAX = 4*KTOT+2
0009            MMAX = 4*KTOT
0010            DO 10 K=1,KTOT
0011            KK = 4*(K-1)
0012            MAN(KK+1,1) = KK+1
0013            MAN(KK+1,2) = KK
0014            MAN(KK+2,1) = KK+1
0015            MAN(KK+2,2) = KK+2
0016            MAN(KK+2,3) = KK-1
0017            MAN(KK+2,4) = KK
0018            MAN(KK+3,1) = KK+4
0019            MAN(KK+3,2) = KK+3
0020            MAN(KK+3,3) = KK+2
0021            MAN(KK+3,4) = KK+1
0022            MAN(KK+4,1) = KK+2
0023            MAN(KK+4,2) = KK+3
              C
0024            NAM(KK+1,1) = KK+2
0025            NAM(KK+1,2) = KK+1
0026            NAM(KK+1,3) = KK+3
0027            NAM(KK+2,1) = KK+2
0028            NAM(KK+2,2) = KK+3
0029            NAM(KK+2,3) = KK+4
0030            NAM(KK+3,1) = KK+4
0031            NAM(KK+3,2) = KK+3
0032            NAM(KK+3,3) = KK+6
0033            NAM(KK+4,1) = KK+3
0034            NAM(KK+4,2) = KK+5
0035         10 NAM(KK+4,3) = KK+6
              C
0036            MAN(1,2) = 0
0037            MAN(2,3) = 0
0038            MAN(2,4) = 0
0039            KK = 4*KTOT
```

```
FORTRAN IV-PLUS V02-04G          11:28:46     29-MAR-78          PAGE 2
INPUT.FTN       /TR:BLOCKS/WR

0040            MAN(NN+1,1) = NN
0041            MAN(NN+2,1) = NN-1
0042            MAN(NN+2,2) = NN
        C-------POSITIONS AND PROPERTIES -------------------------
0043            N2 = NMAX/2
0044            DO 20 N1=1,N2
0045            N  = N1-1
0046            NN = 2*N
0047            X(NN+1) = 0
0048            X(NN+2) = 1.
0049            Y(NN+1) = FLOAT(N2-N1)
0050            Y(NN+2) = FLOAT(N2-N1)
        C
0051            XD(NN+1) = 0
0052            YD(NN+1) = 0
0053            XD(NN+2) = 0
0054         20 YD(NN+2) = 0
        C
0055            DO 30 M=1,MMAX
0056            XX(M) = 0
0057            YY(M) = 0
0058            XY(M) = 0
0059            RHO(M) = 10.
0060            LAME1(M) = 3567.1
0061            LAME2(M) = 343.14
0062            COHES(M) = 0
0063            TANPH1(M)= 0
0064         30 TANPS1(M)= 0
        C
0065            PRINT 2001, (N,X(N),Y(N),XD(N),YD(N),N=1,NMAX)
0066            PRINT 2002, (M,XX(M),YY(M),XY(M),RHO(M),LAME1(M),LAME2(M),
               .            COHES(M),TANPH1(M),TANPS1(M),M=1,MMAX)
0067            PRINT 2003, (N,(MAN(N,I),I=1,IMAX),N=1,NMAX)
0068            PRINT 2004, (M,(NAM(M,J),J=1,3),M=1,MMAX)
0069            RETURN
0070      2001 FORMAT (///' NODE DATA'//5X,'N',6X,'X',11X,'Y',10X,'XD',
               .          10X,'YD',/(1X,I5,1P4E12.4))
0071      2002 FORMAT (///' ZONE DATA'//5X,'M',5X,'XX',10X,'YY',10X,'XY',
               .          10X,'AM',9X,'LAME1'7X,'LAME2'7X,'COHES'7X,'TANPH1'
               .          6X,'TANPS1'/(I6,1P9E12.4))
0072      2003 FORMAT (///' ZONES SURROUNDING EACH NODE'//5X,'N',5X,'1',
               .          5X,'2',5X,'3',5X,'4',5X,'5',5X,'6',5X,'7',5X,'8',
               .          5X,'9',5X,'10',/(5I6))
0073      2004 FORMAT (///' NODES SURROUNDING EACH ZONE'//5X,'M',5X,'1',
               .          5X,'2',5X,'3',/(4I6))
0074            END


PROGRAM SECTIONS

 NAME        SIZE                    ATTRIBUTES

$CODE1   001552    437               RW,I,CON,LCL
$PDATA   000010      4               RW,D,CON,LCL
$IDATA   000460    152               RW,D,CON,LCL
$VARS    000020      8               RW,D,CON,LCL
```

```
FORTRAN IV-PLUS V02-04G          11:29:11     29-MAR-78           PAGE 1
PLOTM.FTN        /TR:BLOCKS/wR

0001           SUBROUTINE PLOTM
       C*****************************************************************
       C                                                                *
       C        PREPARATION OF MESH PLOTS                               *
       C                                                                *
       C*****************************************************************
0002           INCLUDE 'COMMON.FTN'
0003  *        REAL LAME1,LAME2
0004  *        COMMON /CGRID/ X(70),Y(70),XD(70),YD(70),GPM(70),
      *       .   XNPH(70),YNPH(70),
      *       .   XX(70),YY(70),XY(70),DXXFV(70),DYYFV(70),DXYFV(70),
      *       .   AM(70),RHO(70),LAME1(70),LAME2(70),COHES(70),TANPHI(70),
      *       .   TANPSI(70),MAN(70,10),NAM(70,3)
0005  *        COMMON /COTHR/ DT,NITER,AKAT,GRAV,ITER,NMAX,MMAX,IMAX,TIME,
      *       .   ZERO,NPRI,LCONT,IFLAG,KTOT,NBUF,NPL,NPLOT,ITPL(10),
      *       .   SCALEX,SCALEY
0006  *        COMMON /CPLOT/ BUF1(202),BUF2(202),BUF3(202)
0007  *        COMMON /CIMPC/ IMP(10),IMD(10),IFL(10),IMZ(10),IFG(10),
      *       .   IFR(10),N2M(10),N3M(10),MMM(10),N4M(10),TOLC(10),
      *       .   STNL,STNU,STSH,XMU,TOL,BDT,BDT1,CON1,CON2,TFRAC,
      *       .   FN(10),FS(10),DRN(10),DRS(10),DRST(10),XNC(10),YNC(10),DN
0008           CALL PLOT(11.,0.,-3)
       C
0009           DO 20 M=1,MMAX
0010           IF (AM(M) .EQ. 0) GO TO 20
0011           AX = SCALEX*X(NAM(M,3))
0012           AY = SCALEY*Y(NAM(M,3))
0013           CALL PLOT(AX,AY,3)
0014           DO 10 J=1,3
0015           AX = SCALEX*X(NAM(M,J))
0016           AY = SCALEY*Y(NAM(M,J))
0017           CALL PLOT(AX,AY,2)
0018        10 CONTINUE
0019        20 CONTINUE
       C-------PRINT RESULTS -----------------------------------
0020           PRINT 2003
0021           DO 101 N=1,NMAX
0022           PRINT 3003, N,(MAN(N,I),I=1,IMAX)
0023       101 CONTINUE
0024           PRINT 2004, (M,(NAM(M,J),J=1,3),AM(M),M=1,MMAX)
0025           PRINT 2007, (N,GPM(N),N=1,NMAX)
0026           PRINT 2222, (K,IFG(K),IFL(K),IFR(K),IMP(K),IMD(K),N2M(K),N3M(K),
      *       .            MMM(K),IMZ(K),K=1,LCONT)
0027      2222 FORMAT (//' CONTACT LINKS'
      *       .         /'    L   IFG  IFL  IFR  IMP  IMD  N2M  N3M  MMM  IMZ'
      *       .         /(1X,10I5))
0028      2001 FORMAT (//' NODE DATA'//5X,'N',6X,'X',11X,'Y',10X,'XD',
      *       .        10X,'YD',/(1X,I5,1P4E12.4))
0029      2002 FORMAT (//' ZONE DATA'//5X,'M',5X,'XX',10X,'YY',10X,'XY',
      *       .        10X,'AM',9X,'LAME1'7X,'LAME2'7X,'COHES'7X,'TANPHI'
      *       .        6X,'TANPSI'/(I6,1P9E12.4))
0030      2003 FORMAT (//' ZONES SURROUNDING EACH NODE'//5X,'N',5X,'1',
      *       .        5X,'2',5X,'3',5X,'4',5X,'5',5X,'6',5X,'7',5X,'8',
      *       .        5X,'9',5X,'10')
0031      3003 FORMAT (1X,11I6)
0032      2004 FORMAT (//' NODES SURROUNDING EACH ZONE'//5X,'M',5X,'1',
```

```
FORTRAN IV-PLUS V02-04G          11:29:11     29-MAR-78          PAGE 2
PLOTM.FTN       /TR:BLOCKS/WR

                .        5X,'2',5X,'3',5X,'MASS'/(4I6,1PE12.4))
0033      2007 FORMAT (//' GRID POINT MASSES'//5X,'N'4X,'MASS'
                .         /(I6,1PE12.4))
0034           RETURN
0035           END


PROGRAM SECTIONS

 NAME        SIZE                    ATTRIBUTES

$CODE1   001152    309               RW,I,CON,LCL
$PDATA   000024     10               RW,D,CON,LCL
$IDATA   000472    157               RW,D,CON,LCL
$VARS    000022      9               RW,D,CON,LCL
$TEMPS   000002      1               RW,D,CON,LCL
CGRID    016374   3710               RW,D,OVR,GBL
COTHR    000110     36               RW,D,OVR,GBL
CPLOT    004570   1212               RW,D,OVR,GBL
CIMPC    001064    282               RW,D,OVR,GBL


TOTAL SPACE ALLOCATED = 026274  5726

,LIST.LST/LI:1/-SP=PLOTM
```

```
FORTRAN IV-PLUS V02-046          11:29:38     29-MAR-78          PAGE 1
MESH.FTN         /IR:BLOCKS/WR

0001           SUBROUTINE MESH
       C*****************************************************************
       C                                                                *
       C        GENERATION OF ARBITRARY MESHES                          *
       C                                                                *
       C*****************************************************************
0002           BYTE LIST,LENG,IFLA,LWAIT
0003           INCLUDE 'COMMON.FTN'
0004  *        REAL LAME1,LAME2
0005  *        COMMON /CGRID/ X(70),Y(70),XD(70),YD(70),GPM(70),
      *       .   XNPH(70),YNPH(70),
      *       .   XX(70),YY(70),XY(70),DXXFV(70),DYYFV(70),DXYFV(70),
      *       .   AM(70),RHO(70),LAME1(70),LAME2(70),COHES(70),TANPH1(70),
      *       .   TANPS1(70),MAN(70,10),NAM(70,3)
0006  *        COMMON /COTHR/ DT,NITER,AKA1,GRAV,ITER,NMAX,MMAX,IMAX,TIME,
      *       .   ZERO,NPRI,LCONT,IFLAG,NTOT,NBUF,NPL,NPLOT,ITPL(10),
      *       .   SCALEX,SCALEY
0007  *        COMMON /CPLOT/ BUF1(202),BUF2(202),BUF3(202)
0008  *        COMMON /CIMPC/ IMP(10),IMD(10),IFL(10),IMZ(10),IFG(10),
      *       .   IFR(10),N2M(10),N3M(10),MMM(10),N4M(10),TOLC(10),
      *       .   SINL,SINU,STSH,XMU,TOL,BDT,BDT1,CON1,CON2,IFRAC,
      *       .   FN(10),FS(10),DRN(10),DRS(10),DRST(10),XNC(10),YNC(10),DN
0009           COMMON /CNICE/ NTIMES,FACTOR,IBOUND(100)
0010           DIMENSION LIST(20,20),LENG(20),IFLA(20),LWAIT(20),
              .   LTEMPA(20),LTEMPB(20)
0011           DATA NTIMES,FACTOR /10,0.5/
       C
       C-----READ NUMBER OF BLOCKS AND SCALE FOR PLOTTING
0012           READ  1000, NBLOCK
0013           PRINT 2000, NBLOCK
0014           READ  1001, SCALE
0015           PRINT 2001, SCALE
0016           SCALEX = SCALE
0017           SCALEY = SCALE
0018           MMAX   = 0
0019           NMAX   = 0
0020           IMAX   = 10
       C
0021           DO 100 NB=1,NBLOCK
       C-------READ NUMBER OF CORNERS AND THEIR COORDINATES
0022           READ  1000, NCORN
0023           PRINT 2002, NCORN
0024           READ  1002, (LIST(1,I),X(I),Y(I),I=1,NCORN)
0025           PRINT 2005, (I,LIST(1,I),X(I),Y(I),I=1,NCORN)
       C-------INITIALIZATIONS
0026           NMAX = NMAX+NCORN
0027           NW   = 0
0028           DO 5 I=2,20
0029           IFLA(I) = 0
0030           LENG(I) = 0
0031           LWAIT(I)= 0
0032           DO 5 J=1,20
0033         5 LIST(I,J) = 0
0034           IF (NCORN .GT. 3) GO TO 10
       C-------CASE OF TRIANGULAR BLOCK
0035           MMAX = MMAX+1
```

```
FORTRAN IV-PLUS V02-04G          11:29:38     29-MAR-78           PAGE 2
MESH.FTN         /TR:BLOCKS/WR

0036           AM(MMAX) = 1.
0037           DO 6 I=1,3
0038         6 NAM(MMAX,I) = LIST(1,I)
0039           GO TO 100
       C-------FORM INITIAL LIST
0040        10 IFLA(1)  = 1
0041           NLIST = 1
0042           NSPLT = 1
0043           LENG(NSPLT) = NCORN
       C-------FIND THE TWO CLOSEST NON CONTIGUOUS CORNERS---------
       C-------------------------------------------------------
0044        22 DISM = 1.E10
0045           NCL  = LENG(NSPLT)
0046           NLIM = NCL-2
       C
0047           DO 30 IX=1,NLIM
0048           II = IX+2
0049           IP = LIST(NSPLT,IX)
0050           DO 30 JX=II,NCL
0051           JP = LIST(NSPLT,JX)
0052           DISTS = (X(IP)-X(JP))**2+(Y(IP)-Y(JP))**2
0053           PRINT 3010,IP,JP,DISTS,DISM
0054         3010 FORMAT('IP,JP,DISTS,DISM'2I3,1P2E12.4)
0055           IF (DISTS .GT. DISM) GO TO 30
0056           IF (IX .EQ.1 .AND. JX .EQ. NCL) GO TO 30
0057           IM     = IX
0058           JM     = JX
0059           PRINT 3000,IM,JM
0060         3000 FORMAT(' IM,JM'2I3)
       C-------FORM FIRST NEW LIST
0061           DO 40 I=1,IM
0062        40 LTEMPA(I) = LIST(NSPLT,I)
0063           DO 50 I=JM,NCL
0064           II = I+IM-JM+1
0065        50 LTEMPA(II)= LIST(NSPLT,I)
0066           LA = IM+NCL-JM+1
0067           DO 55 I=LA,19
0068        55 LTEMPA(I+1) = 0
0069           PRINT 3001,(LTEMPA(I),I=1,6)
0070         3001 FORMAT(' LIST   '6I3)
       C-------CHECK AREA
0071           AR = 0
0072           DO 56 I=2,LA
0073           N1 = LTEMPA(I-1)
0074           N2 = LTEMPA(I)
0075        56 AR = AR+(Y(N1)+Y(N2))*(X(N1)-X(N2))
0076           N1 = LTEMPA(LA)
0077           N2 = LTEMPA(1)
0078           AR = AR+(Y(N1)+Y(N2))*(X(N1)-X(N2))
0079           ARC= 0.1*DISTS
0080           IF (AR .GT. ARC) GO TO 59
0081           GO TO 30
0082        59 CONTINUE
       C-------FORM SECOND NEW LIST
0083           DO 60 I=IM,JM
0084           II = I-IM+1
```

```
FORTRAN IV-PLUS V02-04G          11:29:38     29-MAR-78           PAGE 3
MESH.FTN         /TR:BLOCKS/WR

0085        60 LTEMPB(II) = LIST(NSPLT,1)
0086           DO 70 I=11,19
0087        70 LTEMPB(I+1) = 0
0088           PRINT 3001,(LTEMPB(I),I=1,6)
0089           LB = JM-IM+1
       C-------CHECK AREA
0090           AR = 0
0091           DO 66 I=2,LB
0092           N1 = LTEMPB(I-1)
0093           N2 = LTEMPB(I)
0094        66 AR = AR+(Y(N1)+Y(N2))*(X(N1)-X(N2))
0095           N1 = LTEMPB(LB)
0096           N2 = LTEMPB(1)
0097           AR = AR+(Y(N1)+Y(N2))*(X(N1)-X(N2))
0098           IF (AR .LT. ARC) GO TO 30
0099           DISM = DISTS
0100           IMIN = IM
0101           JMIN = JM
0102           PRINT 4000,IMIN,JMIN,DISM
0103      4000 FORMAT(' IMIN,JMIN,DISM' 213,1PE12.4)
0104        30 CONTINUE
       C
       C-------FIND AN UNUSED LIST
0105           DO 35 I=1,NLIST
0106           IF (IFLA(I) .NE. 0) GO TO 35
0107           NN     = I
0108           GO TO 36
0109        35 CONTINUE
0110           NLIST = NLIST+1
0111           NN     = NLIST
0112        36 IFLA(NN) = 1
       C-------FORM FINAL LISTS
0113           DO 41 I=1,IMIN
0114        41 LIST(NN,I) = LIST(NSPLT,I)
0115           DO 42 I=JMIN,NCL
0116           II = I+IMIN-JMIN+1
0117        42 LIST(NN,II)= LIST(NSPLT,I)
0118           LENG(NN) = IMIN+NCL-JMIN+1
0119           DO 43 I=LENG(NN),19
0120        43 LIST(NN,I+1) = 0
       C
0121           DO 44 I=IMIN,JMIN
0122           II = I-IMIN+1
0123        44 LIST(NSPLT,II) = LIST(NSPLT,I)
0124           LENG(NSPLT) = JMIN-IMIN+1
0125           DO 45 I=LENG(NSPLT),19
0126        45 LIST(NSPLT,I+1) = 0
       C-------CASE OF TRIANGULAR LISTS
0127           IF (LENG(NN) .GT. 3) GO TO 57
0128           MMAX = MMAX+1
0129           AM(MMAX) = 1.
0130           NAM(MMAX,1) = LIST(NN,1)
0131           NAM(MMAX,2) = LIST(NN,2)
0132           NAM(MMAX,3) = LIST(NN,3)
0133           IFLA(NN) = 0
0134           GO TO 58
```

```
FORTRAN IV-PLUS V02-04G          11:29:38     29-MAR-78            PAGE 4
MESH.FTN          /TR:BLOCKS/WR

0135        57 NW = NW+1
0136           LWAIT(NW) = NN
           C
0137        58 IF (LENG(NSPLT) .GT. 3) GO TO 22
0138           MMAX = MMAX+1
0139           AM(MMAX) = 1.
0140           NAM(MMAX,1) = LIST(NSPLT,1)
0141           NAM(MMAX,2) = LIST(NSPLT,2)
0142           NAM(MMAX,3) = LIST(NSPLT,3)
0143           IFLA(NSPLT) = 0
           C-------GET NEXT LIST FOR SPLITTING IT
0144           IF (NW .LE. 0) GO TO 100
0145           NSPLT = LWAIT(NW)
0146           NW = NW-1
0147           GO TO 22
0148         100 CONTINUE
           C-------GENERATE MAN (ZONES AROUND EACH NODE)
0149           DO 108 M=1,MMAX
0150           DO 108 J=1,3
0151           N =NAM(M,J)
0152           DO 106 I=1,IMAX
0153           IF (MAN(N,I) .NE. 0) GO TO 106
0154           MAN(N,I) = M
0155           GO TO 108
0156         106 CONTINUE
0157           PRINT 5000
0158        5000 FORMAT(' NO SPACE FOR THIS LINK - ABORTED')
0159           STOP
0160         108 CONTINUE
           C-------PRINT RESULTS
0161           PRINT 2003, (N,(MAN(N,I),I=1,IMAX),N=1,NMAX)
0162           PRINT 2004, (M,(NAM(M,J),J=1,3),M=1,MMAX)
           C-------REDUCE ZONE SIZE AND IMPROVE MESH
0163           CALL REDUCE
           C      CALL DIAG
0164           IF (NTIMES .GT. 0) CALL NICE
           C-------PLOT MESH
0165           CALL PLOTST(0.025,'CM')
0166           CALL PLOTM
0167           CALL PLOTND
0168           RETURN
           C
0169        1000 FORMAT(8I10)
0170        1001 FORMAT(8F10.0)
0171        1002 FORMAT(I10,2F10.0)
0172        2000 FORMAT(//' NUMBER OF BLOCKS       ='I3)
0173        2001 FORMAT(//' SCALE FOR PLOTTING     ='1PE12.4)
0174        2002 FORMAT(//' NUMBER OF CORNERS      ='I3)
0175        2005 FORMAT(//' CORNER COORDINATES'
                .          /'   SEQ.NO.  COR.NO.         X            Y'
                .          /(4X,I3,5X,I3,5X,1P2E12.4))
0176        2003 FORMAT (///' ZONES SURROUNDING EACH NODE'//5X,'N',5X,'1',
                .          5X,'2',5X,'3',5X,'4',5X,'5',5X,'6',5X,'7',5X,'8',
                .          5X,'9',5X,'10',/(11I6))
0177        2004 FORMAT (///' NODES SURROUNDING EACH ZONE'//5X,'M',5X,'1',
                .          5X,'2',5X,'3',/(4I6))
```

```
FORTRAN IV-PLUS V02-04G          11:31:30     29-MAR-78           PAGE 1
REDUCE.FTN       /TR:BLOCKS/WR

0001           SUBROUTINE REDUCE
       C*****************************************************************
       C                                                                *
       C        CLOSE MESH TO DESIRED DENSITY                           *
       C                                                                *
       C*****************************************************************
0002           INCLUDE 'COMMON.FTN'
0003 *         REAL LAME1,LAME2
0004 *         COMMON /CGRID/ X(70),Y(70),XD(70),YD(70),GPM(70),
     *        .   XNPH(70),YNPH(70),
     *        .   XX(70),YY(70),XY(70),DXXFV(70),DYYFV(70),DXYFV(70),
     *        .   AM(70),RHO(70),LAME1(70),LAME2(70),COHES(70),TANPHI(70),
     *        .   TANPSI(70),MAN(70,10),NAM(70,3)
0005 *         COMMON /COTHR/ DT,NITER,ARAT,GRAV,ITER,NMAX,MMAX,IMAX,TIME,
     *        .   ZERO,NPRI,LCONT,IFLAG,KTOT,NBUF,NPL,NPLOT,ITPL(10),
     *        .   SCALEX,SCALEY
0006 *         COMMON /CPLOT/ BUF1(202),BUF2(202),BUF3(202)
0007 *         COMMON /CIMPC/ IMP(10),IMD(10),IFL(10),IMZ(10),IFG(10),
     *        .   IFR(10),N2M(10),N3M(10),MMM(10),N4M(10),TOLC(10),
     *        .   STNL,STNU,STSH,XMU,TOL,BDT,BDT1,CON1,CON2,IFRAC,
     *        .   FN(10),FS(10),DRN(10),DRS(10),DRST(10),XNC(10),YNC(10),DN
0008           COMMON /CNICE/ NTIMES,FACTOR,IBOUND(100)
0009           READ  1001, AMAXL
0010           PRINT 2000, AMAXL
0011           AMAXS = AMAXL*AMAXL
0012           M1A  = 0
       C-------MAIN ITERATION LOOP ------------------------------------
0013         1 M1A = M1A+2
0014           IF (M1A .GT. MMAX) GO TO 501
0015           N1  = NAM(M1A,1)
0016           N2  = NAM(M1A,2)
0017           N3  = NAM(M1A,3)
       C
0018           D1S = (X(N1)-X(N2))**2+(Y(N1)-Y(N2))**2
0019           D2S = (X(N2)-X(N3))**2+(Y(N2)-Y(N3))**2
0020           D3S = (X(N3)-X(N1))**2+(Y(N3)-Y(N1))**2
       C
0021           IF (AMAXS .GE. AMAX1(D1S,D2S,D3S)) GO TO 500
0022           IF (D1S .GE. AMAX1(D2S,D3S)) GO TO 10
0023           IF (D2S .GE. AMAX1(D1S,D3S)) GO TO 20
0024           NA  = N3
0025           NB  = N1
0026           NN1 = N2
0027           GO TO 30
0028        10 NA  = N1
0029           NB  = N2
0030           NN1 = N3
0031           GO TO 30
0032        20 NA  = N2
0033           NB  = N3
0034           NN1 = N1
       C
0035        30 DO 42 I=1,IMAX
0036           M3A = MAN(NA,I)
0037           IF (M3A .EQ. 0) GO TO 42
0038           IF (M3A .EQ. M1A) GO TO 42
```

```
FORTRAN IV-PLUS V02-04G          11:31:30     29-MAR-78          PAGE 2
REDUCE.FTN       /TR:BLOCKS/WR

0039           DO 40 J=1,IMAX
0040           IF (M3A .NE. MAN(NB,J)) GO TO 40
0041           M2A = M3A
0042           GO TO 50
0043        40 CONTINUE
0044        42 CONTINUE
       C
0045           IF2 = 0
0046           GO TO 60
0047        50 IF2 = 1
       C
0048           DO 55 I=1,3
0049           IF (NA .NE. NAM(M2A,I)) GO TO 55
0050           I2  = I+1
0051           IF (I2 .EQ. 4) I2=1
0052           NN2 = NAM(M2A,I2)
0053        55 CONTINUE
       C-------REORGANISE LINKS IN THE SAFE SIDE
0054        60 NMAX = NMAX+1
0055           IBOUND(NMAX) = IF2
0056           X(NMAX) = 0.5*(X(NA)+X(NB))
0057           Y(NMAX) = 0.5*(Y(NA)+Y(NB))
0058           MMAX = MMAX+1+IF2
0059           AM(MMAX) = 1.
0060           AM(MMAX-IF2) = 1.
       C
0061           DO 70 I=1,IMAX
0062           IF (MAN(NB,I) .NE. M1A) GO TO 70
0063           MAN(NB,I) = MMAX
0064           GO TO 80
0065        70 CONTINUE
       C
0066        80 DO 90 I=1,IMAX
0067           IF (MAN(NN1,I) .NE. 0) GO TO 90
0068           MAN(NN1,I) = MMAX
0069           GO TO 100
0070        90 CONTINUE
       C
0071       100 MAN(NMAX,1) = MMAX
0072           MAN(NMAX,2) = M1A
       C
0073           DO 110 I=1,3
0074           IF (NAM(M1A,I) .EQ. NB) NAM(M1A,I)=NMAX
0075       110 CONTINUE
       C
0076           NAM(MMAX,1) = NB
0077           NAM(MMAX,2) = NN1
0078           NAM(MMAX,3) = NMAX
       C
       C-------REORGANISE LINKS ON THE OTHER SIDE IF REQUIRED
0079           IF (IF2 .EQ. 0) GO TO 500
0080           MMA1 = MMAX-1
       C
0081           MAN(NMAX,3) = M2A
0082           MAN(NMAX,4) = MMA1
       C
```

```
FORTRAN IV-PLUS V02-04G          11:31:30     29-MAR-78          PAGE 3
REDUCE.FTN       /TR:BLOCKS/WR

0083            DO 120 I=1,IMAX
0084            IF (MAN(NB,I) .NE. M2A) GO TO 120
0085            MAN(NB,I) = MMA1
0086            GO TO 130
0087          120 CONTINUE
        C
0088          130 DO 140 I=1,IMAX
0089            IF (MAN(NN2,I) .NE. 0) GO TO 140
0090            MAN(NN2,I) = MMA1
0091            GO TO 150
0092          140 CONTINUE
        C
0093          150 DO 160 I=1,3
0094            IF (NAM(M2A,I) .EQ. NB) NAM(M2A,I)=NMAX
0095          160 CONTINUE
        C
0096            NAM(MMA1,1) = NB
0097            NAM(MMA1,2) = NMAX
0098            NAM(MMA1,3) = NN2
        C
0099          500 GO TO 1
        C
0100          501 DO 510 M=1,MMAX
0101            N1  = NAM(M,1)
0102            N2  = NAM(M,2)
0103            N3  = NAM(M,3)
0104            D1S = (X(N1)-X(N2))**2+(Y(N1)-Y(N2))**2
0105            D2S = (X(N2)-X(N3))**2+(Y(N2)-Y(N3))**2
0106            D3S = (X(N3)-X(N1))**2+(Y(N3)-Y(N1))**2
0107            IF (AMAX1(D1S,D2S,D3S) .LE. AMAXS) GO TO 510
0108            M1A = MOD(M1A,2)-1
0109            GO TO 1
0110          510 CONTINUE
        C
0111            PRINT 2003, (N,(MAN(N,I),I=1,IMAX),N=1,NMAX)
0112            PRINT 2004, (M,(NAM(M,J),J=1,3),M=1,MMAX)
0113            RETURN
0114         1001 FORMAT(8F10.0)
0115         2000 FORMAT(' MAXIMUM EDGE LENGTH ALLOWED ='1PE12.4)
0116         2003 FORMAT (///' ZONES SURROUNDING EACH NODE'//5X,'N',5X,'1',
                .        5X,'2',5X,'3',5X,'4',5X,'5',5X,'6',5X,'7',5X,'8',
                .        5X,'9',5X,'10',/(1116))
0117         2004 FORMAT (///' NODES SURROUNDING EACH ZONE'//5X,'M',5X,'1',
                .        5X,'2',5X,'3',/(4I6))
0118            END


PROGRAM SECTIONS

 NAME        SIZE                  ATTRIBUTES

$CODE1   003126   811              RW,I,CON,LCL
$IDATA   000322   105              RW,D,CON,LCL
$VARS    000066    27              RW,D,CON,LCL
$TEMPS   000022     9              RW,D,CON,LCL
CGRID    016374  3710              RW,D,OVR,GBL
```

```
FORTRAN IV-PLUS V02-04G          11:32:37     29-MAR-78          PAGE 1
NICE.FTN         /TR:BLOCKS/WR

0001           SUBROUTINE NICE
       C*****************************************************************
       C                                                                *
       C        SIMPLE MESH OPTIMIZATION                                *
       C                                                                *
       C*****************************************************************
0002           INCLUDE 'COMMON.FTN'
0003  *        REAL LAME1,LAME2
0004  *        COMMON /CGRID/ X(70),Y(70),XD(70),YD(70),GPM(70),
      *       .   XNPH(70),YNPH(70),
      *       .   XX(70),YY(70),XY(70),DXXFV(70),DYYFV(70),DXYFV(70),
      *       .   AM(70),RHO(70),LAME1(70),LAME2(70),COHES(70),TANPHI(70),
      *       .   TANPSI(70),MAN(70,10),NAM(70,3)
0005  *        COMMON /COTHR/ DT,NITER,ARAT,GRAV,ITER,NMAX,MMAX,IMAX,TIME,
      *       .   ZERO,NPRI,LCONT,IFLAG,KTOT,NBUF,NPL,NPLOT,ITPL(10),
      *       .   SCALEX,SCALEY
0006  *        COMMON /CPLOT/ BUF1(202),BUF2(202),BUF3(202)
0007  *        COMMON /CIMPC/ IMP(10),IMD(10),IFL(10),IMZ(10),IFG(10),
      *       .   IFR(10),N2M(10),N3M(10),MMM(10),N4M(10),TOLC(10),
      *       .   STNL,STNU,STSH,XMU,TOL,BDT,BDT1,CON1,CON2,TFRAC,
      *       .   FN(10),FS(10),DRN(10),DRS(10),DRST(10),XNC(10),YNC(10),DN
0008           COMMON /CNICE/ NTIMES,FACTOR,IBOUND(100)
       C
0009           DO 80 NTIME=1,NTIMES
0010           DO 90 N=1,NMAX
0011           IF (IBOUND(N) .EQ. 0) GO TO 90
0012           XSUM = 0
0013           YSUM = 0
0014           ANUM = 0
       C
0015           DO 15 I=1,IMAX
0016           M = MAN(N,1)
0017           IF (M .EQ. 0) GO TO 20
       C
0018           DO 10 J=1,3
0019           IF (NAM(M,J) .NE. N) GO TO 10
0020           JM = J+1
0021           IF (JM .EQ. 4) JM=1
0022           XSUM = XSUM+X(NAM(M,JM))
0023           YSUM = YSUM+Y(NAM(M,JM))
0024           ANUM = ANUM+1.
0025           PRINT 1000,N,I,M,J,JM,NAM(M,JM),ANUM,XSUM,YSUM,FACTOR,X(N),Y(N)
0026      1000 FORMAT(1X,6I4,6F6.2)
0027        10 CONTINUE
0028        15 CONTINUE
0029        20 X(N) = FACTOR*X(N)+(1.-FACTOR)*XSUM/ANUM
0030           Y(N) = FACTOR*Y(N)+(1.-FACTOR)*YSUM/ANUM
0031        90 CONTINUE
0032        80 CONTINUE
0033           RETURN
0034           END

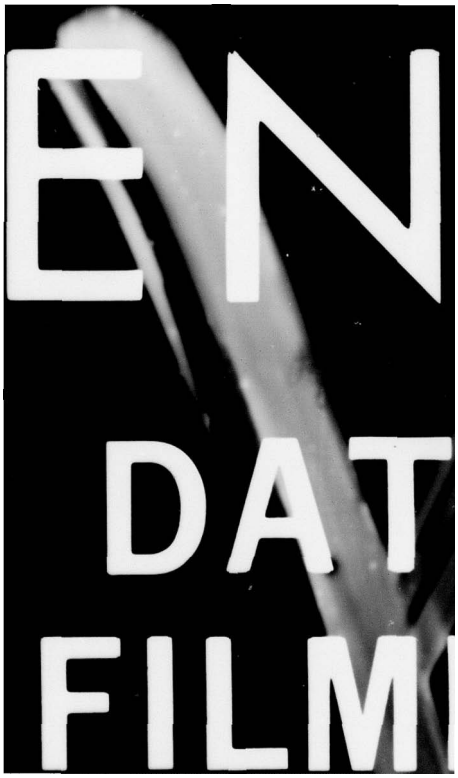

PROGRAM SECTIONS

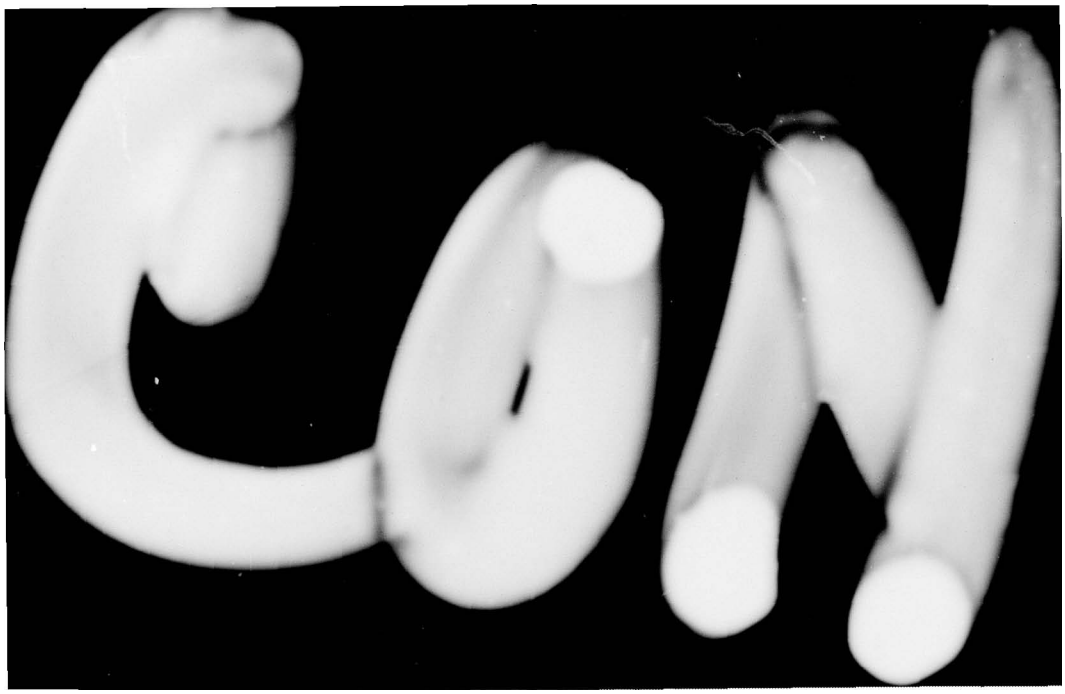
 NAME        SIZE                    ATTRIBUTES
```

In accordance with letter from DAEN-RDC, DAEN-ASI dated 22 July 1977, Subject: Facsimile Catalog Cards for Laboratory Technical Publications, a facsimile catalog card in Library of Congress MARC format is reproduced below.

Maini, Tidu
Computer modelling of jointed rock masses / by Tidu Maini ... [et al.], Dames and Moore, Los Angeles, Calif. Vicksburg, Miss. : U. S. Waterways Experiment Station ; Springfield, Va. : available from National Technical Information Service, 1978.
iii, 396 p. : ill. ; 27 cm. (Technical report - U. S. Army Engineer Waterways Experiment Station ; N-78-4)
Prepared for Defense Nuclear Agency, Washington, D. C., and Office, Chief of Engineers, U. S. Army, Washington, D. C., under Contract No. DACA39-77-C-0004.

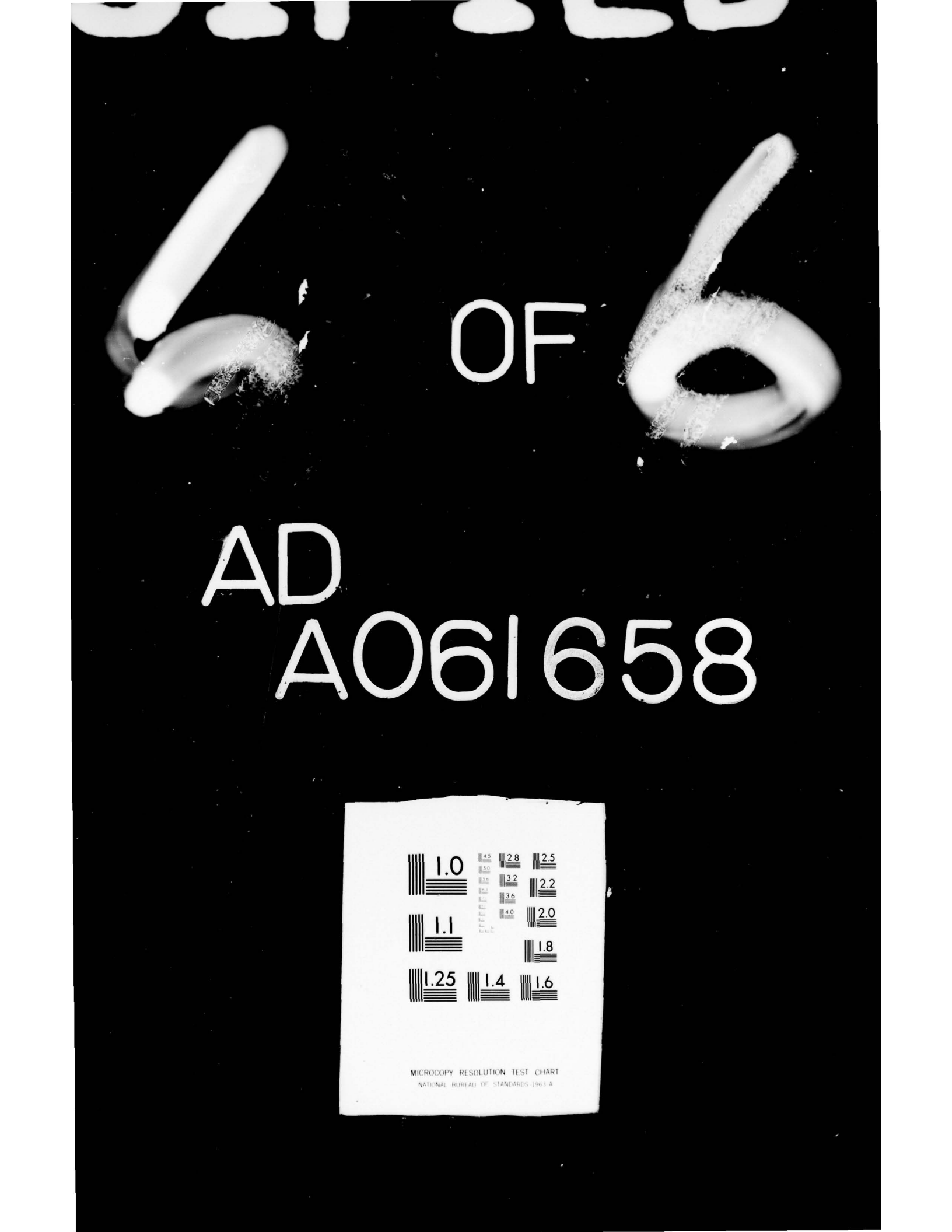
1. Computer programs. 2. Computerized models. 3. Dynamic loads. 4. Jointed rock. 5. Rock deformation. 6. Rock masses. I. Dames and Moore. II. Defense Nuclear Agency. III. United States. Army. Corps of Engineers. IV. Series: United States. Waterways Experiment Station, Vicksburg, Miss. Technical report ; N-78-4.
TA7.W34 no.N-78-4


EN
DAT
FILM


CON

AD-A061 658 DAMES AND MOORE LOS ANGELES CA F/G 8/7
COMPUTER MODELING OF JOINTED ROCK MASSES.(U)
AUG 78 P CUNDALL, J MARTI, P BERESFORD DACA39-77-C-0004
UNCLASSIFIED WES-TR-N-78-4 NL




OF
AD
A061658
1.0
1.1
1.25
1.4
1.6
2.8
2.5
3.2
2.2
3.6
2.0
4.0
1.8
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963-A

SUPPLEMENTARY

INFORMATION

AD-A061658

DEPARTMENT OF THE ARMY
WATERWAYS EXPERIMENT STATION, CORPS OF ENGINEERS
P. O. BOX 631
VICKSBURG, MISSISSIPPI 39180

IN REPLY REFER TO: WESSP

31 January 1979

Errata Sheet

No. 1

COMPUTER MODELLING OF JOINTED ROCK MASSES

Technical Report N-78-4
August 1978

1. The names of the authors on both the cover and in block 7 of the Form 1473 should read:

   Peter Cundall, Joaquin Marti, Peter Beresford, Nigel Last, Margaret Asgian

2. The first two lines of the facsimile catalog card, the page facing page 396, should read:

   Cundall, Peter
   Computer modelling of jointed rock masses / by Peter Cundall...