THE USE OF COMPUTERS IN ENGINEERING MECHANICS EDUCATION

Department of Engineering Mechanics
The University of Michigan

June 30, 1962

This material is distributed by The Ford Foundation Project on the Use of Computers in Engineering Education at The University of Michigan. This report appears in the library edition of the Final Report of the Project and is also issued as a separate booklet. Similar "Curriculum Reports" for other engineering disciplines are also available on request.

ABSTRACT

During the past two years, the faculty of the Engineering College of The University of Michigan has been exploring the integration of computers into the various undergraduate curricula. This report describes the Engineering Mechanics program at the University and discusses the role which the faculty of the Department feels computers should play in this curriculum. Digital computer solutions to engineering problems have been demonstrated in Departmental courses, although students have not been assigned digital computer problems as part of their homework.

The Department, which operates an analog laboratory of its own, uses analog computers extensively for both demonstration and laboratory purposes.

This report contains a selected set of three example problems with complete computer solutions prepared by a Departmental faculty member. These may be considered as a supplement to the 96 example engineering problems, including some related to Engineering Mechanics subject areas, which have been published previously by the Project.

Table of Contents

Tables

USE OF COMPUTERS IN ENGINEERING MECHANICS EDUCATION*

I. INTRODUCTION

The Ford Foundation Project on the Use of Computers in Engineering Education at The University of Michigan has enabled the faculty and students of the College of Engineering to test on a fairly large scale the role of the computer in undergraduate engineering. The present report attempts to set forth the views and efforts of the Department of Engineering Mechanics in this regard, in the hope that others can benefit in some measure from our experiences. It is recognized that the makeup and aims of the departments of mechanics vary to a large extent from school to school, and hence it is thought worthwhile to set forth our program and course offerings first, so that a frame of reference is in the reader's mind from the start. Our conclusions and opinions can then be modified to suit the conditions at the reader's institution. This report should be read in conjunction with the final report of the project. It is not intended to be self-contained. It should be appreciated also that due to changes which can be expected in the expanding fields of computer technology and numerical analysis, and the varying needs of science and industry in general, any and all of the ideas and conclusions presented here can become obsolete very rapidly. Whatever impression the reader receives as to the use of computers in the mechanics curriculum, it behooves him to be aware of the possibilities and potentialities as they develop, and to revise his ideas accordingly.

One special elaboration on these last two sentences is of particular note. Much of the present use made of computers by engineers is of the "slide rule" type. That is, the same methods are used on the high speed computer as were found effective on a desk calculator, and the only advantage gained is in speed and relief from tedium. Recently in several areas of science, users of the computer have broken away from utilizing just the arithmetical hardware of the computer and have made extensive use of the logical or decision-making side of it. In the long run, this is probably the most exciting side of the computer, and the one to watch most closely. The direct application of this to mechanics is not immediately clear, but one can perceive potentialities which could open doors now almost impenetrably closed (such fields as the study of turbulence, fatigue, crack propagation, to name but a few of the possibilities). Here, considerable research is still necessary before any tangible rewards are gained.

In the following, when the word computer is used without any modifying adjective, it is understood to mean digital computer.

---

* This report represents the group opinion of the Department. In particular, Professors S. K. Clark, W. R. Debler, R. A. Dodge, J. H. Enns, W. P. Graebel, R. M. Haythornthwaite, B. Herzog, and M. J. Kaldjian have made contributions to it. The report was prepared by W. P. Graebel with their assistance.

II. THE PROGRAM IN ENGINEERING MECHANICS

The undergraduate engineering student would normally elect the engineering mechanics program at the beginning of his sophomore year. He would start with the usual elementary sequence of courses (Table IFA). Following these courses, at the second semester of the junior or senior level he takes a series of intermediate courses (Table IFB) which lay the foundations and give an overall view of the main branches of mechanics. He is also required to select a minimum of seven credits of advanced subjects in engineering mechanics, in which he has considerable latitude (Table IFC). These courses are usually offered at least once a year, so that by proper planning, a student can elect a sequence of courses in one particular area of mechanics. In addition, the student can select seven to nine credits in any college of the University, and a group option of twelve or fourteen hours from any other program in the College of Engineering. (Frequent choices for the latter are aerodynamics, electronics, instrumentation, metallurgy, naval architecture, nuclear engineering, physics, structural engineering, etc.) It is not uncommon to find that a student obtaining a bachelor's degree in engineering mechanics will, with perhaps an additional semester's work, receive an additional B.S. degree in applied mathematics, engineering physics, or in various other curricula available to him in the College.

As is evident from the course outlines, the Engineering Mechanics Department does not offer any design courses as such. The student is required to take design courses in other departments, and can elect more in his group option if he so desires. The Department has felt that its principal contribution is not that of producing designers, but rather the training of the engineer in depth in the fundamentals of his field so that he is aware of the foundations and realizes the possibilities and limitations of mathematical models in his work. The student with such training at the bachelor's level receives ready acceptance from industries dealing with diverse technical interests. As might be expected, a large percentage of the students decide to continue their studies in graduate schools throughout the country, most continuing in mechanics, although a significant number go into allied fields such as mathematics, systems engineering, communications, physics, nuclear engineering, space sciences, etc. The student going to these other fields usually finds that his early training allows him to compete with his fellow students in these other disciplines at their level.

Since many of the students do enter graduate school upon completion of their bachelor's training, our course work is designed with this in mind, and a brief description of this program is perhaps necessary to give a complete overall picture. E.M. 412, 422, 441 (Table IFB), while intended as terminal courses at the B.S. level are beginning courses for students entering our our graduate program from other schools. Beyond these, no other specific mechanics courses are required at the graduate level and the student is free to specialize as he and his advisor see fit

TABLE IFA

	Elementary Course Work	Credits
E.M. 208 or 218	Statics	3
E.M. 210 or 219	Strength of Materials	4
E.M. 212 or 402	Laboratory in Strength of Materials	1 or 2
E.M. 343 or 345	Dynamics	3
E.M. 324 or 326	Fluid Mechanics	3 or 4

TABLE IFB

	Intermediate Course Work	Credits
E.M. 403	Experimental Mechanics	2
E.M. 412	Intermediate Mechanics of Materials	3
E.M. 422	Intermediate Mechanics of Fluids	3
E.M. 441	Intermediate Mechanics of Vibrations	3

TABLE IFC

Advanced Course Work in Mechanics on the Undergraduate-Beginning Graduate Level

		Credits
E.M. 411	Structural Mechanics	3
E.M. 413	Photoelasticity	2
E.M. 416	Stress Analysis	2
E.M. 514	Theory of Elasticity I	3
E.M. 515	Theory of Plates	3
E.M. 518	Theory of Elastic Stability I	3
E.M. 519	Theory of Plasticity I	3
Dynamics		
E.M. 542	Advanced Dynamics	3
E.M. 543	History of Dynamics	2
E.M. 544	Dynamics and Stability of Rotors	3
E.M. 545	Vibrations of Continuous Media	3
E.M. 547	Theory of Gyroscopes	2
Fluid Mechanics		
E.M. 522	Mechanics of Inviscid Fluids I	3
E.M. 523	Mechanics of Viscous Fluids I	3
E.M. 529	Advanced Laboratory in Mechanics of Fluids	2
Thermodynamics		
E.M. 527	Thermodynamics	2

TABLE IFD

Advanced Courses in Mechanics on the Graduate Level

Solid Mechanics		Credits
E.M. 714	Theory of Elasticity II	3
E.M. 715	Theory of Shells	3
E.M. 718	Theory of Elastic Stability II	3
E.M. 719	Theory of Plasticity II	3
Dynamics		
E.M. 714	Theory of Vibrations	2
E.M. 745	Wave Motion in Continuous Media	3
Fluid Mechanics		
E.M. 721	Mechanics of Inviscid Fluids II	3
E.M. 723	Mechanics of Viscous Fluids II	3
Other		
E.M. 707	Theory of Continuous Media	3

TABLE IFE

	Advanced Mathematics Courses Frequently Elected	Credits
Math. 749	Methods of Partial Differential Equations	3
Math. 750	Methods of Mathematical Physics I	3
Math. 751	Methods of Mathematical Physics II	3
Math. 757	Special Functions in Classical Analysis	3
Math. 777	Tensor Analysis	3

The Ph.D. candidate is, however, required to demonstrate proficiency in a written qualifying examination in the subjects covered in Tables IFA and IFB. He also must take at a later date, an oral examination in the area in which he majored plus two minor areas, one in the Engineering Mechanics Department, the other outside of the Department. The minor in the Department usually consists of two courses selected from one of the groups in Table IFC. The major consists of course selected from groups in Tables IFC and IFD, plus individual reading courses and advanced courses in other departments. Mathematics is heavily stressed, and the student usually finds it desirable to elect several graduate mathematics courses as well. Courses which have been elected most regularly in recent years are given in Table IFE.

III. THE USE OF COMPUTERS IN THE MECHANICS CURRICULUM

During the period when the Project on the Use of Computers in Engineering Education was in effect at The University of Michigan, several points involving the use of digital computers in undergraduate education became apparent. Since these apply to any student (and teacher) user of the computer, they will be set forth first, serving then as a basis of discussion when the role of computers in the mechanics curriculum is discussed. Discussions of other factors can be found in the various reports of the project.

The Computer Requires a Precise Statement of the Problem

Since communicating with the computer is somewhat similar to communicating with a three-year-old child, the student finds at an early stage that he must tell the computer exactly what he wants to do in order to obtain the correct results. This requires an understanding of the mathematics of the problem by the student and also a clear statement of the problem in mathematical language, desirable goals in the training of the engineer.

Of course, this clarity of problem statement should be the goal in all teaching, whether a computer is present or not. The teacher, however, as a human being, will usually accept the work if small errors are made. The computer is a much more stern taskmaster, and will produce satisfactory results only if the problem is presented to it letter perfect.

Like most good things in life, this need for preciseness is not a complete blessing. First of all, it tests only one facet of the student's knowledge, that is, his problem solving ability in a particular subject. It does not in general test his understanding of the fundamentals, since he may be led to a solution by the fact that the problem appeared at a particular part of the course. Also, after the first (usually relatively small) percentage of his time spent on this particular problem, the student's concern is not in furthering his understanding of the engineering involved, but rather with the details of computer programming. Even with a problem-oriented computer language such as MAD, considerable practice, experience, and care are needed to make sure that "punctuation" and similar details do not cause the computer to return undesirable results.

The Student Should Become Familiar with the Computer as a Mathematical Tool

Just as we expect the student to be familiar with techniques such as separation of variables, power series solutions of differential equations, etc., so he should also recognize that numerical methods are powerful tools in problem solving. Through experience he should learn when one method has advantages over another, and what factors play important roles in determining how accurate his answer is. This requires, then, that besides learning to program in a computer language, the student learn numerical analysis as well, including error analysis along with numerical techniques. In fact this knowledge of numerical analysis may be the most significant aspect in the long run. Knowledge of a computer language is necessary in that it provides laboratory experience, but it is the least permanent part of a student's education in that it is subject to rapid obsolescence.

The Computer Serves as a Laboratory

An "exact" solution to a complicated problem provides some inherent satisfaction to many people; however, it is not always possible to see just what role the various parameters play in governing the behavior of the system to which the solution applies. The computer, either through a numerical solution of the basic problem or through evaluation from the exact solution, can provide a graphical presentation of the solution which is frequently more easily understood. Thus a computer can serve much the same function as, say, an elementary laboratory in strength of materials does in the student's education.

Use of the Computer by Students Requires Considerable Availability of Knowledgeable Staff

An instructor considering the assignment of a problem for execution on the computer should first be sure that someone (preferably the instructor himself, or at least someone familiar with the engineering problems) is available to help the students interpret error returns, core dumps, and the like. If the degree-of-problem-difficulty times number-of-students is at all large, it should be anticipated that this help will be required for considerable periods of time over an interval of several weeks per problem. Instructor time, not computer time, is perhaps then the chief "cost" factor involved.

Student Problems on a Computer Require Days if not Weeks from Assignment to Completion

During the existence of the Computer Project, the approach to the computer at The University of Michigan was made relatively easy for the student. Carrier service from locations in the engineering buildings provided delivery of programs to and from the Computing Center twice a day. Keypunches were provided in the engineering buildings as well as at the Computing Center. Personnel were available at several locations for consultation in case of student or faculty difficulty. High speed printers and special services were provided at the Computing Center.

In spite of all these, 24 hours seemed to be the minimum time from delivery to receipt of the program. Breakdown of equipment and computer load at times lengthened this to as much as one week or more, so that average delivery-receipt time at the middle or end of the semester was probably at least 48 hours. Since the average number of passes through the computer for students with some acquaintance with programming, running reasonably short programs, was estimated to be in excess of three, it is easy to see the effect of this on the student. By the time the majority of the students in a class ran a simple problem on the computer, the class would have progressed to another topic, and hence the impact of the computer's contribution would be somewhat dulled. If, in addition, some of the students are unfamiliar with the particular language available, an appreciable amount of class time may be lost in an introduction to this languag

A Certain Number of Students Will Become Completely Engrossed in the Computer to the Point of Losing Interest in Engineering

The computer can be an engrossing object in the student's life, so much so that he feels compelled to devote much of his time and energy to this fascinating device, usually to the detriment of his engineering studies. This is perhaps due to the fact that in a reasonably brief period of time he may gain a high degree of proficiency and find himself in great demand by his teachers as well as his fellow students. This success quite often proves hazardous to his other studies and delays his progress towards a degree.

The Computer Solution Gives only the "Expected" Answer

While this is true to some degree of any method, there is a further danger when the computer is used. When a numerical method is decided upon, this pretty well fixes the type, or behavior, of the solution which will be found. An "Answer" based on this method will usually then be obtained, which, however, may have little bearing on the true physical problem. To illustrate, suppose a problem in the form of a differential equation is to be solved which in some region has either a boundary-layer type behavior (that is, the solution changes rapidly in almost a discontinuous manner) or perhaps the family of solutions become so close together in a region that the computer can jump easily from one solution to another. If the student is unaware of this, he might start off with a mesh size of, say, 0.1, on the next trial change this to 0.05, and next to 0.01. The solution may turn out to be relatively insensitive to this size of change; ergo, the student concludes that his method has converged to the solution. If, however, he would continue to reduce his mesh size until it was of the order of the boundary layer thickness (or the "distance" between radically differing solutions), he would find that his previous efforts were deceiving, and gave him false confidence in his "solution." This seems to emphasize again the need for proper training in error analysis and an understanding of the limitations of numerical computation.

The above could happen even without a computer, and perhaps best serves as an indictment of a cook book approach to numerical computation. The reason that this point is of particular danger when a computer is used, is because when one is performing a hand computation, the relative size of various terms in an equation quite often becomes apparent with little effort, and curiosity as to where various terms become important is easily aroused. In the computer, however, the inner intricacies of the computation are hidden, and unless the programmer is alert to the dangers beforehand, the lack of imagination in the machine gives the wrong solution.

Based on the above considerations and others mentioned in the various project reports, the Department of Engineering Mechanics presently makes only limited use of the digital computer in its existing undergraduate and graduate courses. For the courses in Table IFA which perform a service role besides giving an introduction to mechanics for our own students, there is simply not sufficient time to introduce any computer work without reducing the amount of mechanics taught in these courses. It may very well be desirable to use the computer in these courses, and if additional time were made available to us we would be willing to do so. We do not, however, believe it desirable to reduce the mechanics content of these courses below the level now being given.

For the other courses, instructors' computer solutions have been used to illustrate what various solutions look like when numerical computations are carried out, and also the digital computer has been used to replace some hand calculations which were previously used. So far it has been left optional to the student as to whether he would use a desk calculator or the IBM 709. Those familiar with programming choose the latter, and in the process, convince friends as to the desirability of learning programming. When more students come to these courses with the ability to program fairly involved problems, undoubtedly more use will be made of the digital computer. However, the content of the existing courses were set up before the Computer Project made the computer so accessible. The arrangement and content of these courses will soon come up for review and it is likely that new courses decided on will use the computer in a more integrated fashion.

In summary, then, we believe that our duty as a department is to train the student in the fundmentals of mechanics. We have adopted a wait and see attitude as to whether the computer can further these aims. It is becoming more apparent that it can do so, but at this time the advantages to be gained by widespread use in the undergraduate program do not seem to be economical timewise. However, through the aid made possible by the Computer Project, our future courses can expect to incorporate in varying degrees the use of the computer as a means of furthering our aims.

Analog Computers

The electronic differential analyzer (EDA) has been used by engineers long enough so that it is a familiar teaching tool. These computers are commonplace enough so that little need be said, but perhaps a brief comparison with the digital computer is in order.

The EDA is suited to a much smaller class of problems than the digital computer, namely, the solution of ordinary differential equations. This lack of versatility goes hand in hand with easier operation; hence the student can be introduced to the principles of EDA operations in a few minutes, and can learn to operate a particular machine in an afternoon. The nuances of how to handle infinities, two-point boundary value problems, and nonlinear terms takes more time, but available manuals make such learning relatively simple compared with the digital computer.

The quickness and ease of solution and presentation makes the EDA a tool well adapted to classroom demonstration. The Department of Engineering Mechanics presently owns two 10-amplifier computers plus a 24-amplifier model (the latter was provided through the Computer Project and is available to other departments in the College of Engineering). These are presently used to demonstrate to appropriate classes the behavior of various multi-mass systems, uniform vibrating strings and beams, flow in the boundary layer, the response of nonlinear materials, and other problems with only one independent variable. The students are also encouraged to use the EDA on their own for appropriate problems, and have relatively free access to it. Solutions on the EDA can be made to exhibit some of the dangers of "numerical" solutions in a simpler and more visual manner than on the digital computer. Thus, some of the benefits of the digital computer can be obtained with relatively little financial or time expense.

IV. EXAMPLE PROBLEMS

Included in this section are three problems and their solutions programmed in the MAD (Michigan Algorithm Decoder) language. They are listed in Table IIF.

TABLE IIF

List of Example Problems

Number*	Title	Author	Page
97	Numerical Solution of the Harmonic and Biharmonic Equations	W. P. Graebel	F11
98	Joukowski Airfoil	W. P. Graebel	F23
99	Principal Axes of a Second Order Tensor	W. P. Graebel	F27

* These problems may be considered as a supplement to problems 1 through 96 published in previous reports of the Project.

Example Problem No. 97

NUMERICAL SOLUTION OF THE HARMONIC AND BIHARMONIC EQUATIONS

by

William P. Graebel

Department of Engineering Mechanics

The University of Michigan

The harmonic and biharmonic equations appear many times in problems in mechanics. Potential flow, stress distribution, deflection of a membrane, steady-state temperature distribution are examples. The solution of these by relaxation methods is well known due to the efforts of Southwell and his followers, hence there seems to be little need in recapitulating the problem. The techniques used by these workers to speed the convergence now seem to be of incidental interest if machine computation is to be used, since the added complexity to the program would only be worthwhile if there was need for an extreme number of mesh points.

The arithmetic operations here are trivial; the main problem is to identify whether a point is interior, on the boundary, or exterior to the region of interest, and also how close are the neighbors of the point if any of the neighbors are boundary points. The present programs approach the problem by taking the simplest case, and then modifying this to meet more complex boundaries.

Case 1 (Program Relaxati)

Here we consider the case where all mesh points for a square grid lie on the boundary, hence all boundaries must be a series of straight line segments intersecting at prescribed mesh points. The information necessary to the program is simply the location of the boundary (given by the BOUND vector) and also the value of the function (PSI) at the boundary. For convenience, we also read in the maximum number of times we will allow a point value to be relaxed (NMAX) (to avoid a runaway program if something is wrong) as well as a means of stopping the program when we are content with the answer, that is, when the maximum change made in going through a complete iteration is less than ERRMAX. To avoid programming complexities, a code is used on the BOUND vector, and data is read in for every point on our net. The code used was a simple one, BOUND=0 for an interior point, 1 for points on the boundary, 2 for exterior points, and 3 for interior points which have at least one nearest net neighbor which is an exterior point. Actually, for Case 1, any integer other than zero will work instead of 1 or 2, and 3. The code is stated here for use in Case 3.

PSI is read in for every point, whether or not it is a boundary point. For nonboundary points, any value will do as it is never used.

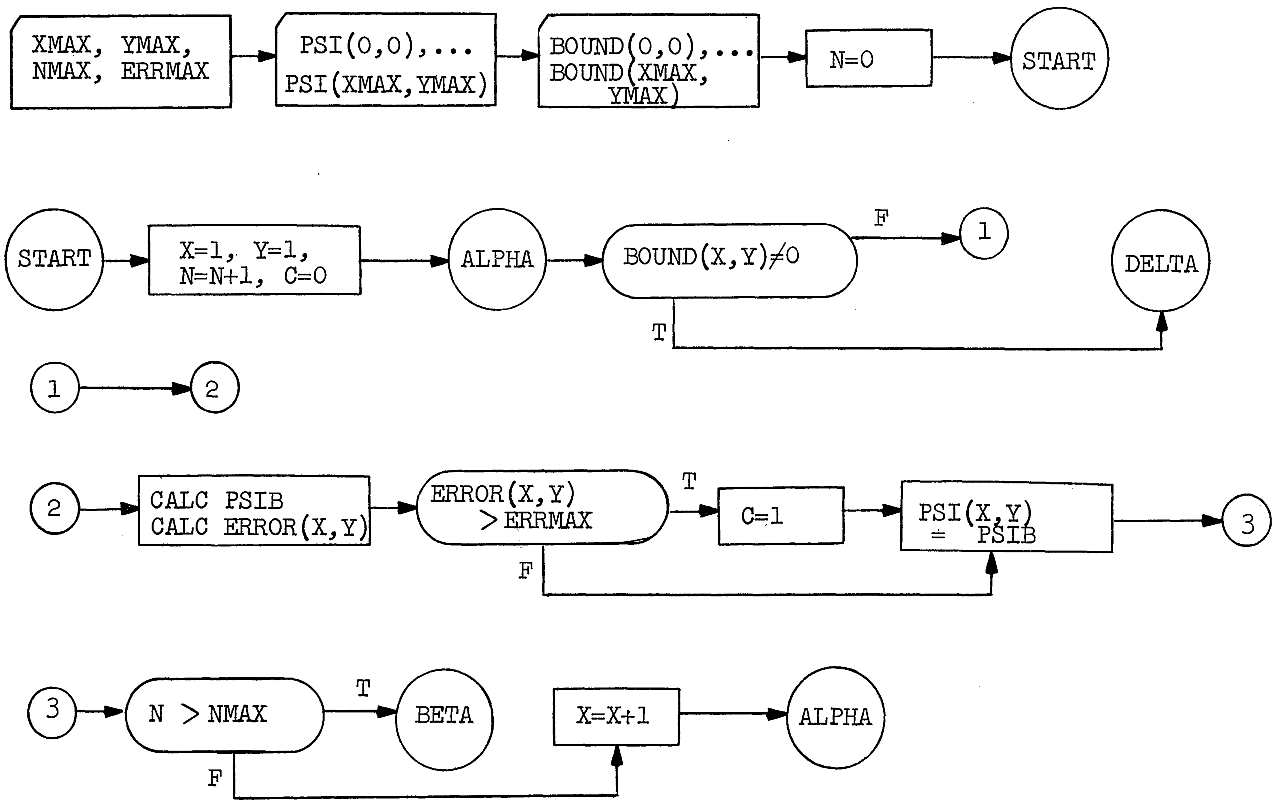
The program now goes through each point in a rectangle XMAX by YMAX in a straightforward manner, checking to see the nature of a point, then either going on if it is not an interior point or else taking the average value of its neighbors if it is an interior point. If NMAX relaxations are undergone without meeting the ERRMAX condition, the program prints out the last value of PSI at each point as well as the difference between this value and the one computed on the previous iteration.

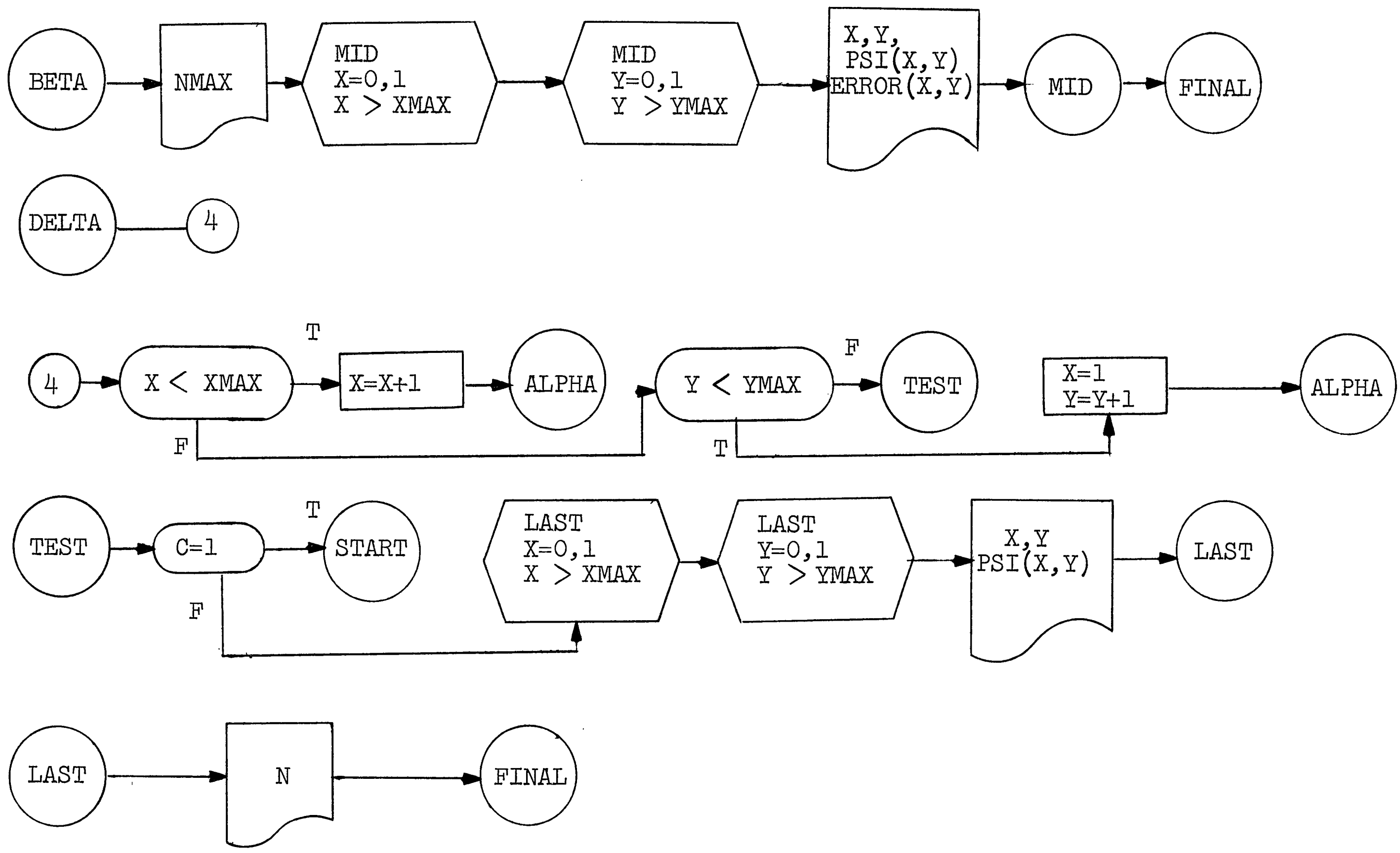
The problem presented represents the irrotational flow in a channel which suddenly increases in width.

Program variables for quantities not described in the text are listed below and are applicable to all four cases.

N	Number of iteration
NMAX	Maximum number of iterations allowed
C	Switch to stop iteration when change becomes small enough
PSI	Solution of harmonic or biharmonic equation
PSIB	New value of PSI
XMAX, YMAX	Size of rectangle enclosing the boundary
PSR, PSL, PSU, PSD	Value of PSI to right, left, above, and below (I, J)
NUM	Numerator of Equation (1)
DEM	Denominator of Equation (1)
T	Entry into random sequence
AVE	Average number of throws needed to get to boundary

A flow diagram for the program (RELAXATI) to solve problems of Case 1 is shown below.





The MAD program is shown below.

```
          RELAXATION METHOD FOR 2D LAPLACE EQUATION
          INTEGER N, C, X, Y, XMAX, YMAX, NMAX, BOUND
          READ FORMAT ABLE, XMAX, YMAX, NMAX, ERRMAX
          VECTOR VALUES ABLE = $3I5,F10.0*$
          DIMB(2) = YMAX + 3
          DIMB(3) = YMAX + 1
          READ FORMAT EASY, PSI(0,0)...PSI(XMAX,YMAX)
          VECTOR VALUES EASY=$(8F10.6)*$
          DIMENSION PSI(625, DIMB(1))
          READ FORMAT FOX, BOUND(0,0)...BOUND(XMAX,YMAX)
          VECTOR VALUES FOX=$(40I2)*$
          DIMENSION BOUND(625,DIMB(1)), DIMB(3)
          VECTOR VALUES DIMB(1) =  2,27,11
          N = 0
START     X = 1
          Y = 1
          N = N + 1
          C = 0
ALPHA     WHENEVER BOUND(X,Y) .NE.0, TRANSFER TO DELTA
          PSIB = (PSI(X - 1,Y) + PSI(X + 1,Y) + PSI(X,Y - 1) + PSI(X,Y
        1 + 1))/4.
          ERROR(X,Y) = .ABS.(PSIB - PSI(X,Y))
          DIMENSION ERROR(625, DIMB(1))
          WHENEVER ERROR(X,Y) .G. ERRMAX, C = 1
          PSI(X,Y) = PSIB
          WHENEVER N.G.NMAX, TRANSFER TO BETA
          X = X + 1
          TRANSFER TO ALPHA
BETA      PRINT FORMAT INTER, NMAX
          VECTOR VALUES INTER = $18H0 EXCEEDED NMAX = I5*$
          THROUGH MID, FOR X = 0,1, X.G.XMAX
          THROUGH MID, FOR Y = 0,1, Y.G.YMAX
MID       PRINT FORMAT ER, X, Y, ERROR(X,Y), X, Y, PSI(X,Y)
```

MAD Program, continued

```
          VECTOR VALUES ER = $S2,  6HERROR(I3,1H,I3,4H) = ,E11.4,S10,4H
         1PSI(,I3,1H,,I3,4H) = ,F10.6*$
          TRANSFER TO FINAL
DELTA     WHENEVER X.L.XMAX
          X = X + 1
          TRANSFER TO ALPHA
          END OF CONDITIONAL
          WHENEVER Y.L.YMAX
          X = 1
          Y = Y + 1
          TRANSFER TO ALPHA
          END OF CONDITIONAL
TEST      WHENEVER C.E.1, TRANSFER TO START
LAST1     THROUGH LAST, FOR X = 0,1,X.G.XMAX
          THROUGH LAST, FOR Y = 0,1,Y.G.YMAX
LAST      PRINT FORMAT OUT, X,Y,PSI(X,Y)
          VECTOR VALUES OUT = $1H0,S2,4HPSI(I5,1H,I5,2H)=F10.6*$
          PRINT FORMAT Q, N
          VECTOR VALUES Q = $1H0,S2,24HNUMBER OF RELAXATIONS = I5*$
FINAL     END OF PROGRAM
```

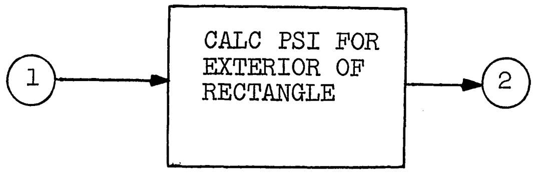
A typical set of computer output is shown below.

```
EXCEEDED NMAX =      70
ERROR(  0,   0) =   .1885E-36        PSI(  0,   0) =    .000000
ERROR(  0,   1) =   .1885E-36        PSI(  0,   1) =    .200000
ERROR(  0,   2) =   .1885E-36        PSI(  0,   2) =    .400000
ERROR(  0,   3) =   .1885E-36        PSI(  0,   3) =    .600000
ERROR(  0,   4) =   .1885E-36        PSI(  0,   4) =    .800000
ERROR(  0,   5) =   .1885E-36        PSI(  0,   5) =   1.000000
ERROR(  0,   6) =   .1885E-36        PSI(  0,   6) =   7.000000
ERROR(  0,   7) =   .1885E-36        PSI(  0,   7) =   7.000000
ERROR(  0,   8) =   .1885E-36        PSI(  0,   8) =   7.000000
ERROR(  0,   9) =   .1885E-36        PSI(  0,   9) =   7.000000
ERROR(  0,  10) =   .1885E-36        PSI(  0,  10) =   7.000000
ERROR(  1,   0) =   .1885E-36        PSI(  1,   0) =    .000000
ERROR(  1,   1) =   .2906E-06        PSI(  1,   1) =    .199967
ERROR(  1,   2) =   .4917E-06        PSI(  1,   2) =    .399946
ERROR(  1,   3) =   .4694E-06        PSI(  1,   3) =    .599946
ERROR(  1,   4) =   .2831E-06        PSI(  1,   4) =    .799967
ERROR(  1,   5) =   .1885E-36        PSI(  1,   5) =   1.000000
ERROR(  1,   6) =   .1885E-36        PSI(  1,   6) =   7.000000
ERROR(  1,   7) =   .1885E-36        PSI(  1,   7) =   7.000000
ERROR(  1,   8) =   .1885E-36        PSI(  1,   8) =   7.000000
ERROR(  1,   9) =   .1885E-36        PSI(  1,   9) =   7.000000
ERROR(  1,  10) =   .1885E-36        PSI(  1,  10) =   7.000000
ERROR(  2,   0) =   .1885E-36        PSI(  2,   0) =    .000000
ERROR(  2,   1) =   .6743E-06        PSI(  2,   1) =    .199922
ERROR(  2,   2) =   .1043E-05        PSI(  2,   2) =    .399873
ERROR(  2,   3) =   .1006E-05        PSI(  2,   3) =    .599874
ERROR(  2,   4) =   .5960E-06        PSI(  2,   4) =    .799922
ERROR(  2,   5) =   .1885E-36        PSI(  2,   5) =   1.000000
ERROR(  2,   6) =   .1885E-36        PSI(  2,   6) =   7.000000
ERROR(  2,   7) =   .1885E-36        PSI(  2,   7) =   7.000000
ERROR(  2,   8) =   .1885E-36        PSI(  2,   8) =   7.000000
ERROR(  2,   9) =   .1885E-36        PSI(  2,   9) =   7.000000
ERROR(  2,  10) =   .1885E-36        PSI(  2,  10) =   7.000000
```

Case 2 (Program Biharmon)

This program is essentially the same as the preceding, except it is used for solving the biharmonic equation with boundaries of the type given in Case 1. The main difference is in the arithmetic operation necessary in computing a new value of PSI, and also the boundary conditions, since the normal derivative of PSI is also given on the boundary. These data were incorporated by simply inserting data in the form of MAD statement cards (cards 19-38) which took care of these boundary conditions. All else is similar to the preceding case.

The example here was taken from Timoshenko and Goodier, Theory of Elasticity, McGraw-Hill, 1951, pp. 483-489. The flow diagram for Case 1 is to be used with the change indicated below going from 1 to 2.



The MAD program for Case 2 is shown below.

```
          RELAXATION METHOD FOR  2D BIHARMONIC EQUATION
          INTEGER N, C, X, Y, XMAX, YMAX, NMAX
          READ FORMAT ABLE, XMAX, YMAX, NMAX, ERRMAX
          VECTOR VALUES ABLE = $3I5,F10.0*$
          DIMB(2) = YMAX + 3
          DIMB(3) = YMAX + 1
          READ FORMAT EASY, PSI(0,0)...PSI(XMAX,YMAX)
          VECTOR VALUES EASY=$(8F10.6)*$
          DIMENSION PSI(625, DIMB(1))
          READ FORMAT FOX, BOUND(0,0)...BOUND(XMAX,YMAX)
          VECTOR VALUES FOX=$(40I2)*$
          DIMENSION BOUND(625,DIMB(1)), DIMB(3)
          VECTOR VALUES DIMB(1) =  2,27,11
          N = 0
START     X = 1
          Y = 1
          N = N + 1
          C = 0
ALPHA     WHENEVER BOUND(X,Y) .NE.0, TRANSFER TO DELTA
          PSI(2,0) = PSI(2,2)
          PSI(3,0) = PSI(3,2)
          PSI(4,0) = PSI(4,2)
          PSI(5,0) = PSI(5,2)
          PSI(6,0) = PSI(6,2)
          PSI(2,8) = PSI(2,6)
          PSI(3,8) = PSI(3,6)
          PSI(4,8) = PSI(4,6)
          PSI(5,8) = PSI(5,6)
          PSI(6,8) = PSI(6,6)
          PSI(0,2) = PSI(2,2) - 4.8
          PSI(0,3) = PSI(2,3) - 4.8
          PSI(0,4) = PSI(2,4) - 4.8
          PSI(0,5) = PSI(2,5) - 4.8
          PSI(0,6) = PSI(2,6) - 4.8
          PSI(8,2) = PSI(6,2) - 4.8
          PSI(8,3) = PSI(6,3) - 4.8
          PSI(8,4) = PSI(6,4) - 4.8
          PSI(8,5) = PSI(6,5) - 4.8
          PSI(8,6) = PSI(6,6) - 4.8
```

MAD Program, continued

```
               PSIB = 0.4*(PSI(X-1,Y) + PSI(X + 1,Y)+ PSI(X,Y -1) +PSI(X,Y+1
              1)) - 0.1*(PSI(X -1,Y -1) + PSI(X-1,Y + 1) +PSI(X+ 1,Y-1) +PSI
              2(X + 1,Y +1))-0.05*(PSI(X,Y -2) + PSI(X,Y +2)+PSI(X -2,Y) +PS
              3I(X +2,Y))
               ERROR(X,Y) = .ABS.(PSIB - PSI(X,Y))
               DIMENSION ERROR(625, DIMB(1))
               WHENEVER ERROR(X,Y) .G. ERRMAX, C = 1
               PSI(X,Y) = PSIB
               WHENEVER N.G.NMAX, TRANSFER TO BETA
               X = X + 1
               TRANSFER TO ALPHA
     BETA      PRINT FORMAT INTER, NMAX
               VECTOR VALUES INTER = $18H0 EXCEEDED NMAX = I5*$
               THROUGH MID, FOR X = 0,1, X.G.XMAX
               THROUGH MID, FOR Y = 0,1, Y.G.YMAX
     MID       PRINT FORMAT ER, X, Y, ERROR(X,Y),X,Y,PSI(X,Y)
               VECTOR VALUES ER = $S2, 6HERRROR(I3,1H,I3,4H) = E11.4,S10,4HP
              1SI(I3,1H,,I3,4H) = F10.6*$
               TRANSFER TO FINAL
     DELTA     WHENEVER X.L.XMAX
               X = X + 1
               TRANSFER TO ALPHA
               END OF CONDITIONAL
               WHENEVER Y.L.YMAX
               X = 1
               Y = Y + 1
               TRANSFER TO ALPHA
               END OF CONDITIONAL
     TEST      WHENEVER C.E.1, TRANSFER TO START
     LAST1     THROUGH LAST, FOR X = 0,1,X.G.XMAX
               THROUGH LAST, FOR Y = 0,1,Y.G.YMAX
     LAST      PRINT FORMAT OUT, X,Y,PSI(X,Y)
               VECTOR VALUES OUT = $1H0,S2,4HPSI(I5,1H,I5,2H)=F10.6*$
               PRINT FORMAT Q, N
               VECTOR VALUES Q = $1H0,S2,24HNUMBER OF RELAXATIONS = I5*$
     FINAL     END OF PROGRAM
```

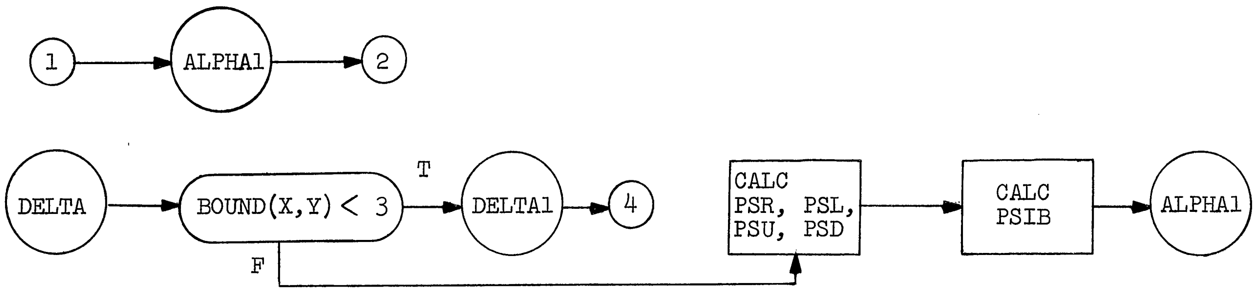
A typical set of computer output is shown below.

```
EXCEEDED NMAX =     70
ERROR(  0,  0) =  .1885E-36          PSI(  0,  0) =    .000000
ERROR(  0,  1) =  .1885E-36          PSI(  0,  1) =    .000000
ERROR(  0,  2) =  .1885E-36          PSI(  0,  2) =  -4.407767
ERROR(  0,  3) =  .1885E-36          PSI(  0,  3) =  -4.003612
ERROR(  0,  4) =  .1885E-36          PSI(  0,  4) =  -3.707660
ERROR(  0,  5) =  .1885E-36          PSI(  0,  5) =  -3.492551
ERROR(  0,  6) =  .1885E-36          PSI(  0,  6) =  -3.319117
ERROR(  0,  7) =  .1885E-36          PSI(  0,  7) =    .000000
ERROR(  0,  8) =  .1885E-36          PSI(  0,  8) =    .000000
ERROR(  1,  0) =  .1885E-36          PSI(  1,  0) =    .000000
ERROR(  1,  1) =  .1885E-36          PSI(  1,  1) =   -.720000
ERROR(  1,  2) =  .1885E-36          PSI(  1,  2) =   -.720000
ERROR(  1,  3) =  .1885E-36          PSI(  1,  3) =   -.720000
ERROR(  1,  4) =  .1885E-36          PSI(  1,  4) =   -.720000
ERROR(  1,  5) =  .1885E-36          PSI(  1,  5) =   -.720000
ERROR(  1,  6) =  .1885E-36          PSI(  1,  6) =   -.720000
ERROR(  1,  7) =  .1885E-36          PSI(  1,  7) =   -.720000
ERROR(  1,  8) =  .1885E-36          PSI(  1,  8) =    .000000
ERROR(  2,  0) =  .1885E-36          PSI(  2,  0) =    .392233
ERROR(  2,  1) =  .1885E-36          PSI(  2,  1) =    .000000
ERROR(  2,  2) =  .1412E-03          PSI(  2,  2) =    .392374
ERROR(  2,  3) =  .3363E-03          PSI(  2,  3) =    .796387
ERROR(  2,  4) =  .3916E-03          PSI(  2,  4) =   1.092340
ERROR(  2,  5) =  .2936E-03          PSI(  2,  5) =   1.307449
ERROR(  2,  6) =  .1171E-03          PSI(  2,  6) =   1.480883
ERROR(  2,  7) =  .1885E-36          PSI(  2,  7) =   1.600000
ERROR(  2,  8) =  .1885E-36          PSI(  2,  8) =   1.480883
```

Case 3 (Program Orelax)

This deals with the same problem as Case 1, except that arbitrary (but continuous) boundaries are now acceptable. The main problem now is getting the additional data on the curved boundaries read into the machine. This is done in the rather awkward but effective manner of inserting statement cards to take care of troublesome points indicated by 3 in the BOUND code. U, D, L, R data tell the distance (zero to one) of the neighboring points (mesh or boundary) above, below, left or right of the point where BOUND=3, and PSU, PSD, PSL, PSR give the values of PSI at these neighboring points. Statements must appear for all neighbors, to avoid further complications in input. Except for the other modification in the arithmetic computation of PSI at a point, everything else is as in Case 1.

The problem here is the same as for Case 1, except that the area changes more gradually, in that a quarter of a circle is substituted for the right angle corner. The flow diagram for Case 1 is applicable with the indicated changes shown below in going from 1 to 2 and DELTA to ALPHA1.



The MAD program for Case 3 is shown below.

```
W P GRAEBEL                      S146D              005    005    000        ORELAX
$COMPILE MAD, EXECUTE, DUMP
           RRELAXATION METHOD FOR 2D LAPLACE EQUATION
           R
           RDATA CARDS IN THE FOLLOWING ORDER
           R
           RXMAX(I5),YMAX(I5),NMAX(I5),ERRMAX(F10)
           R
           RPSI(0,0)...PSI(XMAX,YMAX)  (8F10)
           R
           RBOUNDARIES(0,0)...BOUNDARIES(XMAX,YMAX)(80I1)
           R(THIS LAST CARD SHOULD HAVE ZEROS FOR INTERIOR POINTS,  1 FOR
           RMESH POINTS WHICH LIE ON THE BOUNDARY, AND  2 FOR MESH POINTS
           ROUTSIDE OF THE BOUNDARY.  IF THE NEIGHBOR OF AN INTERIOR
           RPOINT LIES OUTSIDE THE BOUNDARY, THAT POINT SHOULD HAVE A  3
           RINSTEAD OF A ZERO.)
           R
           RU(X,Y), D(X,Y), L(X,Y), R(X,Y) * (DATA)
           R(L, R, U, D CARDS INDICATE DISTANCE FROM INTERIOR POINT TO
           RNEIGHBORS, WHEN AT LEAST ONE NEIGHBOR IS AN EXTERIOR POINT.
           RALL 4 CARDS MUST BE PRESENT FOR EACH SUCH POINT, EVEN IF THE
           RDISTANCE IS 1.)
           R
           RPSL(X,Y), PSR(X,Y), PSU(X,Y), PSD(X,Y) (MAD STATEMENTS)
           RPSL(X,Y), PSR(X,Y), PSU(X,Y), PSD(X,Y) INDICATE THE VALUES OF
           RPSI FOR THE POINTS DESCRIBED ON THE PREVIOUS DATA CARDS. THEY
           RBELONG IMMEDIATELY AFTER DELTA IN THE PROGRAM.)
           R
            DIMENSION PSL(1000, DIMB(1))
            DIMENSION PSR(1000, DIMB(1))
            DIMENSION PSU(1000, DIMB(1))
            DIMENSION PSD(1000, DIMB(1))
            DIMENSION R(1000, DIMB(1))
            DIMENSION U(1000, DIMB(1))
            DIMENSION D(1000, DIMB(1))
            DIMENSION L(1000, DIMB(1))
            DIMENSION PSI(1000, DIMB(1))
            DIMENSION ERROR(1000, DIMB(1))
            DIMENSION BOUND(1000, DIMB(1)), DIMB(3)
```

MAD Program, continued

```
            INTEGER N, C, X, Y, XMAX, YMAX, NMAX
            INTEGER BOUND
            READ FORMAT ABLE, XMAX, YMAX, NMAX, ERRMAX
            VECTOR VALUES ABLE = $3I5,F10.0*$
            5ECTOR VALUES DIMB(1) =  2,27,11
            DIMB(2) = YMAX + 3
            DIMB(3) = YMAX + 1
            READ FORMAT EASY, PSI(0,0)...PSI(XMAX,YMAX)
            5ECTOR VALUES EASY=$(8F10.6)*$
            READ FORMAT FOX, BOUND(0,0)...BOUND(XMAX,YMAX)
            VECTOR VALUES FOX=$(80I1)*$
            READ DATA U, D, L, R
            N = 0
START       X = 1
            8 = 1
            N = N + 1
            C = 0
ALPHA       WHENEVER BOUND(X,Y) .NE.0, TRANSFER TO DELTA
            PSIB = (PSI(X - 1,Y) + PSI(X + 1,Y) + PSI(X,Y - 1) + PSI(X,Y
           1 + 1))/4.
ALPHA1      ERROR(X,Y)=.ABS.(PSIB-PSI(X,Y))
            WHENEVER ERROR(X,Y) .G. ERRMAX, C = 1
            PSI(X,Y) = PSIB
            WHENEVER N.G.NMAX, TRANSFER TO BETA
            X = X + 1
            TRANSFER TO ALPHA
BETA        PRINT FORMAT INTER, NMAX
            VECTOR VALUES INTER = $18H0 EXCEEDED NMAX = I5*$
            3HROUGH MID, FOR X = 0,1, X.G.XMAX
            THROUGH MID, FOR Y = 0,1, Y.G.YMAX
MID         PRINT FORMAT ER, X, Y, ERROR(X,Y) , X, Y, PSI(X,Y)
            VECTOR VALUES ER = $S2,  6HERROR(I3,1H,I3,4H) = E11.4, S10, 4
           1HPSI(I3, 1H, I3, 4H) = F10.6*$
            TRANSFER TO FINAL
DELTA       6HENEVER BOUND(X,Y).L.3,TRANSFER TO DELTA1
           R
           RPSL, PSR, PSU, PSD CARDS HERE
            PSR(10,5)=PSI(11,5)
            PSL(10,5)=PSI(9,5)
            PSD(10,5)=PSI(10,4)
            PSU(10,5)=1.
            PSR(11,5)=PSI(12,5)
            PSL(11,5)=PSI(10,5)
            PSD(11,5)=PSI(11,4)
            PSU(11,5)=1.
            PSR(12,6)=PSI(13,6)
            PSL(12,6)=1.
            PSU(12,6)=PSI(12,7)
            PSD(12,6)=PSI(12,5)
            PSR(12,7)=PSI(13,7)
            PSL(12,7)=1.
            PSU(12,7)=PSI(12,8)
            PSD(12,7)=PSI(12,6)
           R
            PSIB=(PSL(X,Y)/L(X,Y)+PSR(X,Y)/R(X,Y)+PSD(X,Y)/D(X,Y)+PSU(X,Y
           1)/U(X,Y))/(1./L(X,Y)+1./R(X,Y)+1./U(X,Y)+1./D(X,Y))
            3RANSFER TO ALPHA1
DELTA1      WHENEVER X.L.XMAX
            X = X + 1
            TRANSFER TO ALPHA
            END OF CONDITIONAL
            WHENEVER Y.L.YMAX
            7 = 1
            Y = Y + 1
            TRANSFER TO ALPHA
            END OF CONDITIONAL
            6HENEVER C.E.1, TRANSFER TO START
            THROUGH LAST, FOR X = 0,1,X.G.XMAX
            3HROUGH LAST, FOR Y = 0,1,Y.G.YMAX
LAST        PRINT FORMAT OUT, X,Y,PSI(X,Y)
            VECTOR VALUES OUT = $1H0,S2,4HPSI(I3,1H,I3,4H) = F10.6*$
            PRINT FORMAT Q, N
            VECTOR VALUES Q = $1H0,S2,24HNUMBER OF RELAXATIONS = I5*$
FINAL       END OF PROGRAM
```

A typical set of results obtained from the computer program is shown below.

```
EXCEEDED NMAX =      70
ERROR( 0,  0) =  .1885E-36            PSI( 0,  0) =   .000000
ERROR( 0,  1) =  .1885E-36            PSI( 0,  1) =   .200000
ERROR( 0,  2) =  .1885E-36            PSI( 0,  2) =   .400000
ERROR( 0,  3) =  .1885E-36            PSI( 0,  3) =   .600000
ERROR( 0,  4) =  .1885E-36            PSI( 0,  4) =   .800000
ERROR( 0,  5) =  .1885E-36            PSI( 0,  5) =  1.000000
ERROR( 0,  6) =  .1885E-36            PSI( 0,  6) =  7.000000
ERROR( 0,  7) =  .1885E-36            PSI( 0,  7) =  7.000000
ERROR( 0,  8) =  .1885E-36            PSI( 0,  8) =  7.000000
ERROR( 0,  9) =  .1885E-36            PSI( 0,  9) =  7.000000
ERROR( 0, 10) =  .1885E-36            PSI( 0, 10) =  7.000000
ERROR( 1,  0) =  .1885E-36            PSI( 1,  0) =   .000000
ERROR( 1,  1) =  .5420E-06            PSI( 1,  1) =   .199896
ERROR( 1,  2) =  .9090E-06            PSI( 1,  2) =   .399831
ERROR( 1,  3) =  .8792E-06            PSI( 1,  3) =   .599830
ERROR( 1,  4) =  .5290E-06            PSI( 1,  4) =   .799895
ERROR( 1,  5) =  .1885E-36            PSI( 1,  5) =  1.000000
ERROR( 1,  6) =  .1885E-36            PSI( 1,  6) =  7.000000
ERROR( 1,  7) =  .1885E-36            PSI( 1,  7) =  7.000000
ERROR( 1,  8) =  .1885E-36            PSI( 1,  8) =  7.000000
ERROR( 1,  9) =  .1885E-36            PSI( 1,  9) =  7.000000
ERROR( 1, 10) =  .1885E-36            PSI( 1, 10) =  7.000000
ERROR( 2,  0) =  .1885E-36            PSI( 2,  0) =   .000000
ERROR( 2,  1) =  .1259E-05            PSI( 2,  1) =   .199754
ERROR( 2,  2) =  .1963E-05            PSI( 2,  2) =   .399600
ERROR( 2,  3) =  .1892E-05            PSI( 2,  3) =   .599598
ERROR( 2,  4) =  .1125E-05            PSI( 2,  4) =   .799751
ERROR( 2,  5) =  .1885E-36            PSI( 2,  5) =  1.000000
ERROR( 2,  6) =  .1885E-36            PSI( 2,  6) =  7.000000
ERROR( 2,  7) =  .1885E-36            PSI( 2,  7) =  7.000000
ERROR( 2,  8) =  .1885E-36            PSI( 2,  8) =  7.000000
ERROR( 2,  9) =  .1885E-36            PSI( 2,  9) =  7.000000
ERROR( 2, 10) =  .1885E-36            PSI( 2, 10) =  7.000000
```

Case 4 (Program Random)

An interesting variation to the relaxation type solution is the use of random walk techniques. By flipping a four-sided coin to decide which direction to move to next, the value of PSI at the starting point is given by

$$PSI = \frac{\sum_{i=1}^{MMAX} n_i \, PSI_i}{\sum_{i=1}^{MMAX} n_i} \qquad (1)$$

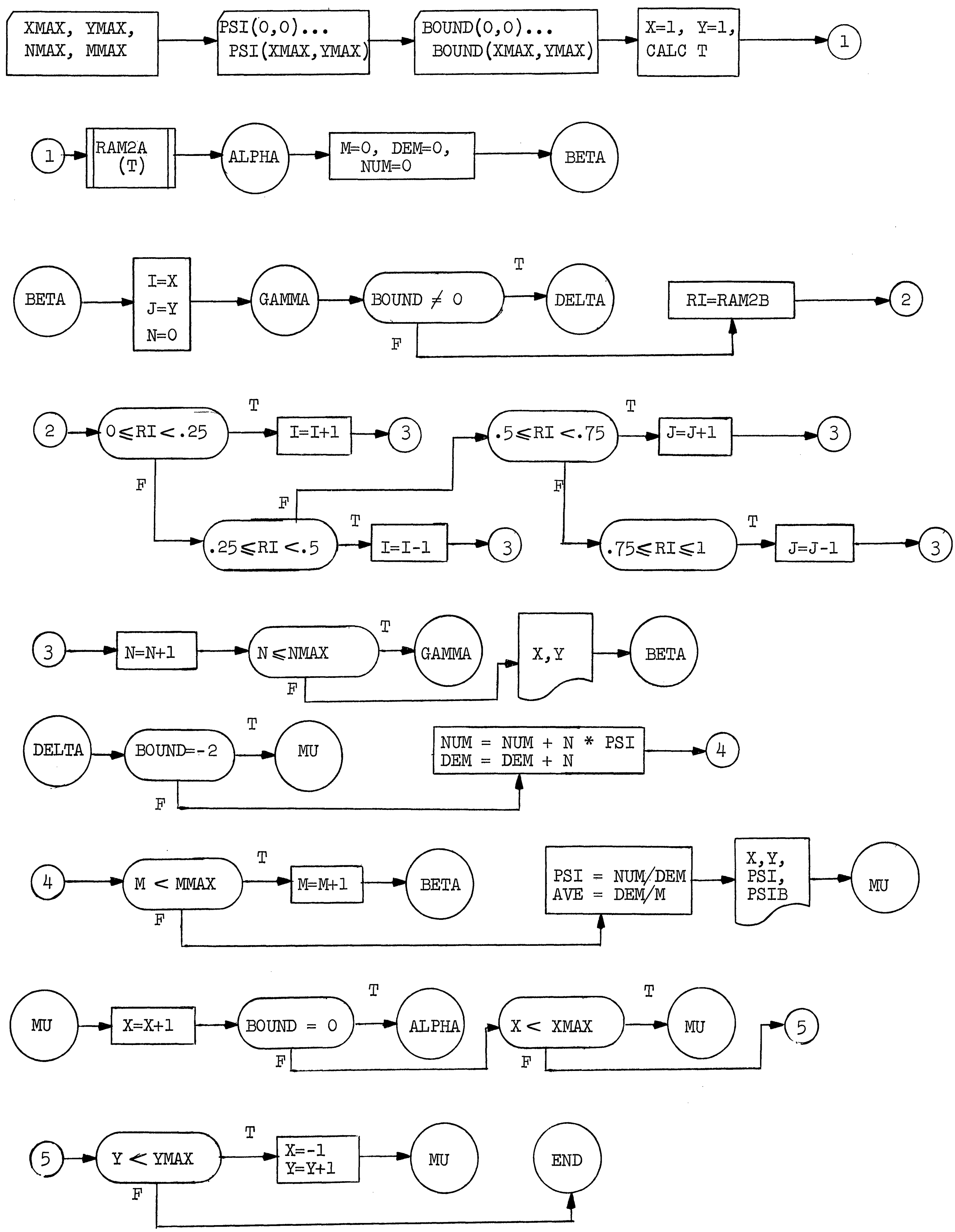
where n_i are the number of "flips" needed to reach the boundary, and PSI_i is the value of PSI at the boundary point reached.

The program given enters a random sequence by means of an entry point determined from the time clock. As a safeguard, if it takes more than NMAX "flips" to reach the boundary, the trial is restarted.

The program is suitable for the same boundaries as Case 1.

The flipping procedure is by no means as efficient as the relaxation approach, and probably is of merit only if information is needed in a small region of a very large domain.

A flow diagram of the program for Case 4 is shown below.



The MAD program is shown below.

```
              RANDOM WALK FOR 2D LAPLACE EQUATION
              NORMAL MODE IS INTEGER
              FLOATING POINT PSIB, PSI, RI, RAM2B., NUM, DEM
              READ FORMAT ABLE, NMAX, MMAX, XMAX, YMAX
              VECTOR VALUES ABLE=$4I5*$
              DIMB(2) =  YMAX + 3
              DIMB(3) =  YMAX + 1
              READ FORMAT EASY, PSI(0,0)...PSI(XMAX,YMAX)
              VECTOR VALUES EASY=$(8F10.6)*$                                    G010
              DIMENSION PSI(625, DIMB(1))
              READ FORMAT FOX, BOUND(0,0)...BOUND(XMAX,YMAX)
              VECTOR VALUES FOX=$(40I2)*$                                       G012
              DIMENSION BOUND(625,DIMB(1)), DIMB(3)
              VECTOR VALUES DIMB(1) =  2,27,11
              X=1                                                               G013
              Y=1                                                               G014
              T = TIME.(0)
              T = T*34359738367
              EXECUTE RAM2A.(T)
 ALPHA        M=0                                                               G015
              NUM=0                                                             G019
              DEM=0                                                             G020
 BETA         I=X                                                               G016
              J=Y                                                               G017
              N=0
 GAMMA        WHENEVER BOUND(I,J).NE.  0, TRANSFER TO DELTA
              RI= RAM2B.(0)
              WHENEVER RI .GE. 0. .AND. RI .L. 0.25
              I=I+1                                                             G024
              OR WHENEVER RI .GE. 0.25 .AND. RI .L. 0.50
              I=I-1
              OR WHENEVER RI .GE. 0.50 .AND. RI .L. 0.75
              J=J+1                                                             G028
              OR WHENEVER RI .GE. 0.75 .AND. RI .LE. 1.00
              J=J-1
              END OF CONDITIONAL                                                G031
              N=N+1                                                             G032
              WHENEVER N.LE. NMAX, TRANSFER TO GAMMA
              PRINT FORMAT DOG,X,Y
              VECTOR VALUES DOG=$22H0 EXCEEDED NMAX FOR X=I5, 7H AND Y=I5*$     G036
              TRANSFER TO BETA                                                  G037
 DELTA        WHENEVER BOUND(I,J) .E.-2, TRANSFER TO MU
              NUM=NUM+ N*PSI(I,J)
              DEM= DEM+N                                                        F039
              WHENEVER M.L. MMAX
              M=M+1
              TRANSFER TO BETA
              END OF CONDITIONAL
              PSI(X,Y)= NUM/DEM
              PSIB = DEM/M
              PRINT FORMAT CHUCK, X, Y, PSI(X,Y), PSIB
              VECTOR VALUES CHUCK=$1H0, S2, 4HPSI(I5,1H,I5,2H)=F10.6,S5,46H
            1 AVERAGE NUMBER OF THROWS TO REACH BOUNDARY = ,F10.6*$
 MU           X=X+1
              WHENEVER BOUND(X,Y).E. 0, TRANSFER TO ALPHA
              WHENEVER  X.L. XMAX, TRANSFER TO MU
              WHENEVER Y .L. YMAX
              X = -1
              Y=Y+1
              TRANSFER TO MU
              END OF CONDITIONAL
              END OF PROGRAM
```

A typical set of computer output is shown below.

```
PSI(   1,    1)=  .334311    AVERAGE NUMBER OF THROWS TO REACH BOUNDARY =   3.410000
PSI(   2,    1)=  .315789    AVERAGE NUMBER OF THROWS TO REACH BOUNDARY =   6.460000
PSI(   3,    1)=  .367710    AVERAGE NUMBER OF THROWS TO REACH BOUNDARY =   6.070000
PSI(   4,    1)=  .296160    AVERAGE NUMBER OF THROWS TO REACH BOUNDARY =   5.990000
PSI(   5,    1)=  .347952    AVERAGE NUMBER OF THROWS TO REACH BOUNDARY =   8.300000
PSI(   6,    1)=  .295181    AVERAGE NUMBER OF THROWS TO REACH BOUNDARY =   9.960000
PSI(   7,    1)=  .300597    AVERAGE NUMBER OF THROWS TO REACH BOUNDARY =   6.700000
PSI(   8,    1)=  .354658    AVERAGE NUMBER OF THROWS TO REACH BOUNDARY =   8.910000
PSI(   9,    1)=  .287466    AVERAGE NUMBER OF THROWS TO REACH BOUNDARY =   7.340000
EXCEEDED NMAX FOR X=   10 AND Y=    1
PSI(  10,    1)=  .308601    AVERAGE NUMBER OF THROWS TO REACH BOUNDARY =   7.790000
PSI(  11,    1)=  .476510    AVERAGE NUMBER OF THROWS TO REACH BOUNDARY =   8.940000
PSI(  12,    1)=  .392925    AVERAGE NUMBER OF THROWS TO REACH BOUNDARY =  10.460000
PSI(  13,    1)=  .425653    AVERAGE NUMBER OF THROWS TO REACH BOUNDARY =  10.720000
PSI(  14,    1)=  .418621    AVERAGE NUMBER OF THROWS TO REACH BOUNDARY =  11.600000
PSI(  15,    1)=  .399718    AVERAGE NUMBER OF THROWS TO REACH BOUNDARY =  14.160000
PSI(  16,    1)=  .275203    AVERAGE NUMBER OF THROWS TO REACH BOUNDARY =  14.760000
PSI(  17,    1)=  .361626    AVERAGE NUMBER OF THROWS TO REACH BOUNDARY =  14.020000
PSI(  18,    1)=  .405120    AVERAGE NUMBER OF THROWS TO REACH BOUNDARY =  12.890000
PSI(  19,    1)=  .238183    AVERAGE NUMBER OF THROWS TO REACH BOUNDARY =  12.440000
PSI(  20,    1)=  .383511    AVERAGE NUMBER OF THROWS TO REACH BOUNDARY =  10.310000
PSI(  21,    1)=  .445896    AVERAGE NUMBER OF THROWS TO REACH BOUNDARY =  14.010000
PSI(  22,    1)=  .260944    AVERAGE NUMBER OF THROWS TO REACH BOUNDARY =   6.990000
PSI(  23,    1)=  .265636    AVERAGE NUMBER OF THROWS TO REACH BOUNDARY =   5.820000
```

Case 3 has been used successfully in the Photoelasticity course for solving for the first stress invariant to enable separation of the stresses. The instructor's program was used by all students. None of the other programs have been class tested.

Example Problem No. 98

JOUKOWSKI AIRFOIL

by

William P. Graebel

Department of Engineering Mechanics

The University of Michigan

Course: Mechanics of Inviscid Fluids I   Credit Hours: 3   Level: Senior

This is a problem which appears in all of the books dealing with classical hydrodynamics, but has become of perhaps decreasing interest with the advent of high-speed aerodynamics. The approach used in such books as Milne-Thomson (Theoretical Hydrodynamics, MacMillan, 1960) and Streeter (Fluid Dynamics, McGraw-Hill, 1948), emphasizes the graphical method. The advantage accrued by use of the computer is to emphasize the fundamentals of the problem.

Basically, this is the problem of the mapping of a circle into a shape with one cusp point. For potential flow with circulation around a circular cylinder in the complex ξ plane, the complex potential is given by

$$= U(\xi e^{-i\alpha} + \frac{a^2}{\xi} e^{i\alpha}) + i \frac{\Gamma}{2\pi} \ln \xi \tag{1}$$

where U and α are the free stream speed and direction, a is the radius of the cylinder, and Γ is the magnitude of the circulation. Transforming this into the complex z plane by means of

$$z = \xi + am\, e^{i\delta} + a^2b^2/(\xi + am\, e^{i\delta}) \tag{2}$$

with

$$b^2 + 2\, mb \cos\theta + m^2 - 1 = 0, \tag{3}$$

the airfoil is obtained. For the cylinder of radius a, $\xi = a\, e^{i\theta}$ on the cylinder, hence

$$z/a = (\cos\theta + m\cos\delta)\,(1 + b^2/R) + i(\sin\theta + m\sin\delta)\,(1 - b^2/R), \tag{4}$$

$$R = (\cos\theta + m\cos\delta)^2 + (\sin\theta + m\sin\delta)^2 \tag{5}$$

The parameters governing the shape of the airfoil are then m and δ. Using equations (1), (2), and (3), the streamlines in the z plane could also be found.

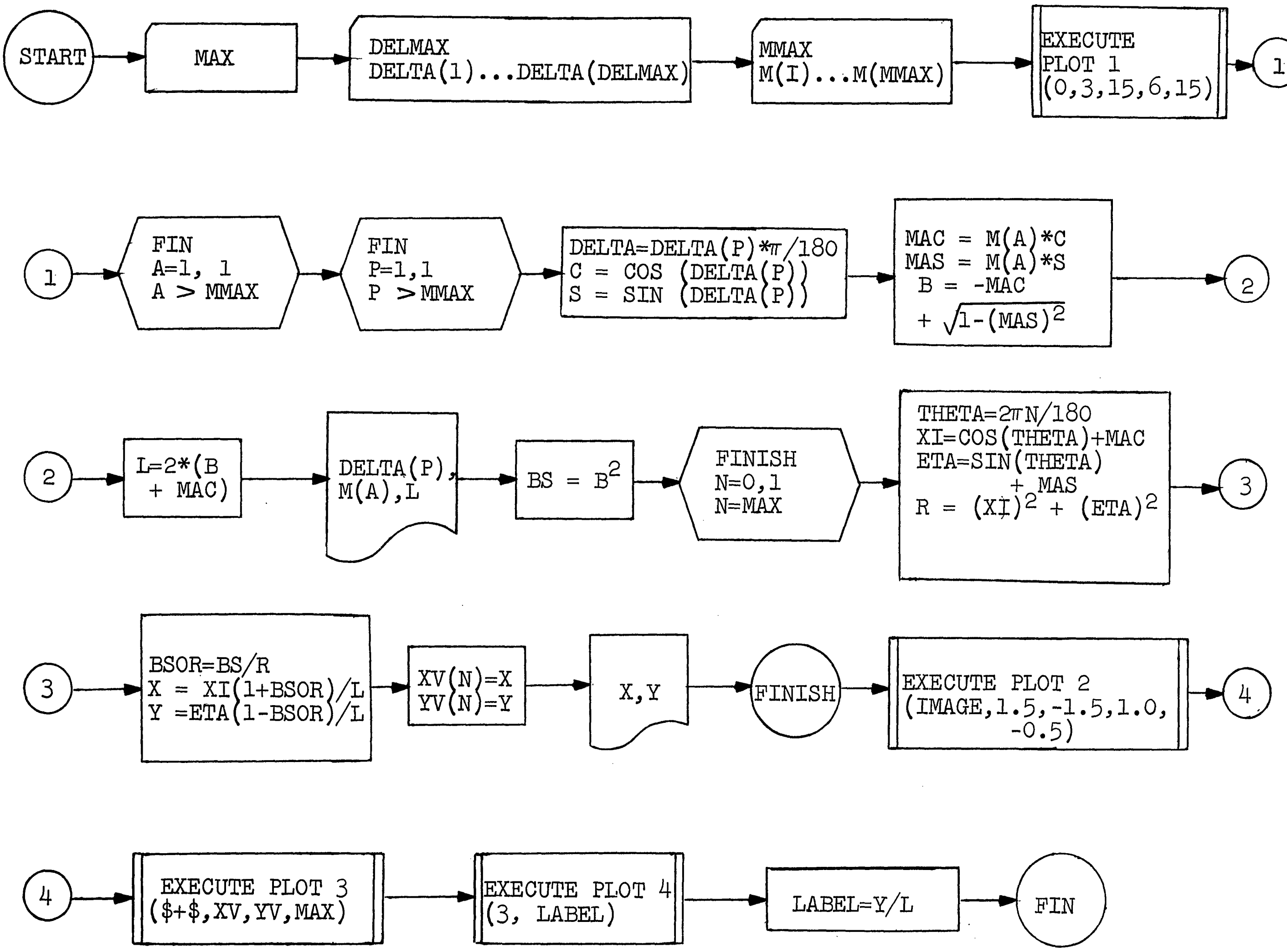
In the present program, we have contented ourselves with plotting just the airfoils. A given number (MAX) of θ's are used to carry out the computations given by equation (4), computing the airfoil shape at points 2π/MAX radians apart on the circle. For given m, δ, the values of these points are given numerically and then graphically, so that the computer can give the shapes of many airfoils in less time than the best draftsman can construct a single airfoil. (The plots are scaled by a factor $L = 2 \cdot (b + m\cos\theta)$, which estimates the chord length of the airfoil.) The effect of variation of m and δ is thus easily grasped.

The results of this program were shown in E.M. 522, and one student, experienced in programming, voluntarily developed his own program for the problem.

A list of variables appearing in the program but not described in the text is given below.

MAX	Number of points on airfoil to be computed
DELTA(I)	The angle delta in degrees
DELTA	The angle delta in radians
DELMAX	Number of DELTA's to be read in
M(I)	M in the transformation $(0 \geqslant M \geqslant 1)$
MMAX	Number of M's to be read in
B	Positive root of equation (3)
XI, ETA	Coordinates in zeta plane
XV, YV	Coordinates of points on airfoil in Z plane

A flow diagram for the program is shown below.



Example Problem No. 98

The MAD program is shown below.

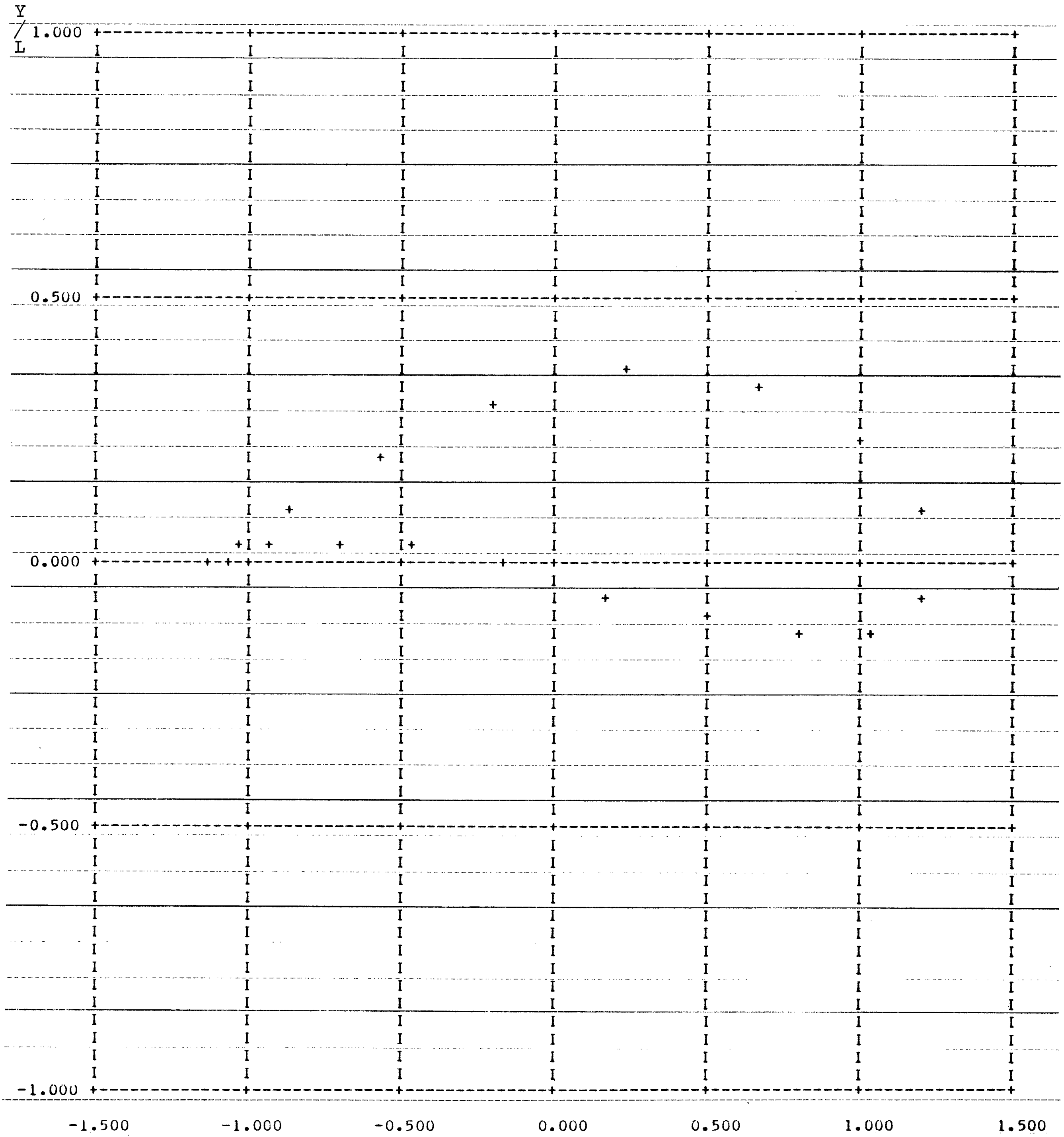
```
                    JOUKOWSKI AIRFOIL

                    READ FORMAT DATA1,MAX
                    READ FORMAT DATA2,DELMAX,DELTA(1)...DELTA(DELMAX)
                    READ FORMAT DATA2,MMAX,M(1)...M(MMAX)
                    INTEGER MAX,DELMAX,MMAX,A,P,N
                    VECTOR VALUES DATA1=$I2*$
                    VECTOR VALUES DATA2=$I2,15F5.5*$
                    DIMENSION DELTA(15),M(15)
                    DIMENSION XV(100), YV(100)
                    DIMENSION IMAGE(2000)
                    VECTOR VALUES PI=3.14159
                    VECTOR VALUES TWOPI=6.28318
                    EXECUTE PLOT1. (0,4,15,6,15)
                    PIO180=PI/180.
                    TPOMAX=TWOPI/MAX
                    THROUGH FIN    ,FOR A=1,1,A.G.MMAX
                    THROUGH FIN    ,FOR P=1,1,P.G.MMAX
                    DELTA=DELTA(P)*PIO180
                    C=COS.(DELTA(P))
                    S=SIN.(DELTA(P))
                    MAC=M(A)*C
                    MAS=M(A)*S
                    B=-MAC    +SQRT.(1.- MAS*MAS)
                    L=2.*(B + MAC)
                    PRINT FORMAT TITLE,DELTA(P),M(A)     ,L
                    VECTOR VALUES TITLE=$13H FOR DELTA = F10.5,16H DEGREES, M/A =
                   1 F10.5,51H THE FOLLOWING ARE OBTAINED FOR X/L AND Y/L. L/A =
                   2F10.7*$
                    BS=B*B
                    THROUGH FINISH,FOR N=0,1,N.E.MAX
                    THETA=N*TPOMAX
                    XI=COS.(THETA)+MAC
                    ETA=SIN.(THETA)+MAS
                    R=XI*XI+ETA*ETA
                    BSOR=BS/R
                    X=XI*(1.+BSOR)/L
                    Y=ETA*(1.-BSOR)/L
                    XV(N) = X
                    YV(N) = Y
 FINISH             PRINT FORMAT RESULT,X,Y
                    EXECUTE PLOT2. (IMAGE, 1.5, -1.5, 1.0, -1.0)
                    PRINT COMMENT $1$
                    EXECUTE PLOT3. ($+$, XV, YV, MAX)
                    EXECUTE PLOT4. (3, LABEL)
                    PRINT COMMENT $1$
                    VECTOR VALUES LABEL = $Y/L$
 FIN                CONTINUE
                    VECTOR VALUES RESULT =$1H S5,F10.6,S5,F10.6*$
                    END OF PROGRAM
```

A typical set of computer output is shown below.

FOR DELTA = 15.00000 DEGREES, M/A = .20000 THE FOLLOWING ARE OBTAINED FOR X/L AND Y/L. L/A = 1.9830129

1.187256	-.050904
1.036732	-.126794
.788810	-.137762
.487903	-.104112
.165991	-.052068
-.153671	-.003221
-.452478	.029137
-.714320	.040291
-.924972	.033028
-1.072219	.016012
-1.146066	.001777
-1.138655	.004253
-1.043833	.035555
-.856800	.101641
-.575282	.196586
-.205076	.296770
.227432	.360709
.661245	.345316
1.006939	.239799
1.188652	.085704



Example Problem No. 99

PRINCIPAL AXES OF A SECOND ORDER TENSOR

by

William P. Graebel

Department of Engineering Mechanics

The University of Michigan

The problem of determining principal axes occurs frequently in mechanics in connection with the stress, strain, and moment of inertia tensors. Using stress as an example, the law of transformation when axes are rotated is

$$\tau'_{ij} = \sum_{k=1}^{3} \sum_{\ell=1}^{3} a_i^k a_j^\ell \tau_{k\ell} \tag{1}$$

where the a's represent direction cosines. As a consequence of the orthogonality of axes,

$$\sum_{k=1}^{3} a_i^k a_k^j = \begin{matrix} 0 \text{ if } i \neq j \\ 1 \text{ if } i = j \end{matrix} \tag{2}$$

The problem is then to find a set of a's for which $\tau'_{ij} = 0$ when $i \neq j$. (This problem is the same as the problem of diagonalizing a matrix.) This is done for given $\tau_{k\ell}$ by solving

$$\sum_{k=1}^{3} \sum_{\ell=1}^{3} a_i^k a_j^\ell \tau_{k\ell} = 0, \; i \neq j \tag{3}$$

and any 3 of equation (2).

In the attached program, besides finding the a's, the τ'_{ij} are also solved for as well as the three principal invariants

$$\text{ONE} = \sum_{i=1}^{3} \tau_{ii} \tag{4}$$

$$\text{TWO} = 0.5\,(\text{ONE})^2 - \frac{1}{2} \sum_{i=1}^{3} \sum_{j=1}^{3} \tau_{ij} \tau_{if}$$

$$\text{THREE} = \det \tau_{ij}$$

The input data is

$XX = \tau_{xx}$, $XY = \tau_{xy} = \tau_{yx}$, $XZ = \tau_{xz} = \tau_{zx}$,

$YY = \tau_{yy}$, $YZ = \tau_{yz} = \tau_{zy}$, $ZZ = \tau_{zz}$,

read in that order on two data cards with the format 3E14.8. The invariants, τ'_{ij}, and a_j^i's are then found in a straightforward manner. τ'_{ij} is found as roots of the equation

$$\tau^3 - (\text{ONE})\,\tau^2 + (\text{TWO})\,\tau - (\text{THREE}) = 0 \tag{5}$$

As a check, the computed a_j^i's are then put into the remaining 6 of equation (2), and should give a number much less than 1 if the axes are orthogonal. A possibility of trouble arises when taking the square root, as the program always takes the positive root even though for some input data the negative root is required. No provision has been made to take care of this.

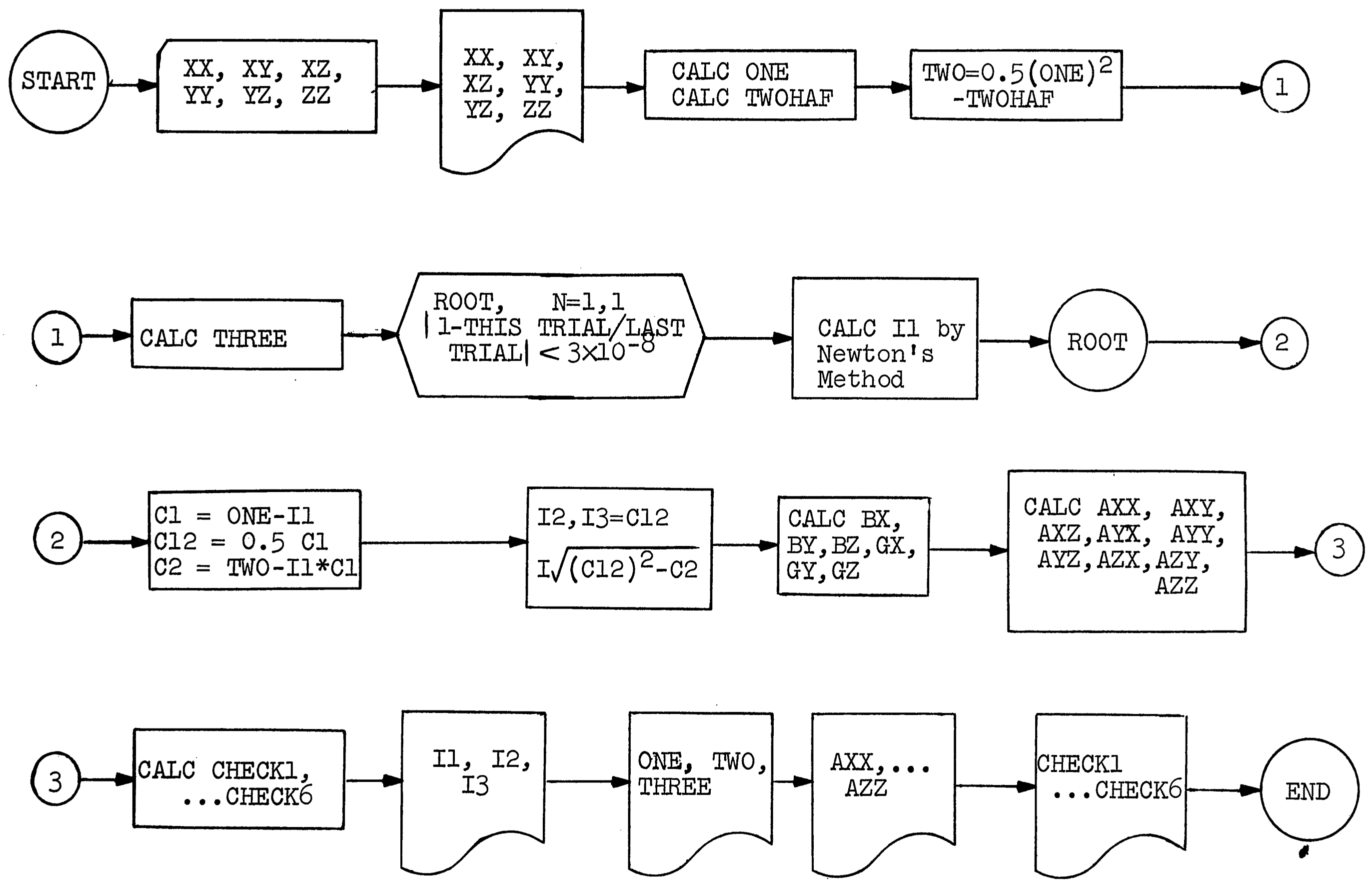
The program can be easily adapted to show the need for carrying sufficient figures. In the THROUGH statement ending at ROOT, the value of one of the roots of equation (5) is found by Newton's method to a greater number of significant places than is required for engineering work. If, however, the number of significant figures retained were only two or three, the check will show that the resulting principal axes are far from being orthogonal. This "extra" precision then is important for the computations which follow it.

This problem is suitable for introduction at the 412-422-542 level. It has not been used in the classroom.

A list of program variables for quantities not described in the text is given below.

AXX, AXY, AXZ, AYX, AYY, AYZ, AZX, AZY, AZZ	Direction cosines of principal axes.
TWOHAF	Second moment of the tensor.
I1, I2, I3	Three roots of cubic equation (5).
I, IN	Temporary names for I1.
CHECK 1,...,CHECK 6	Accuracy check for orthogonal axes. These should all be zero.
TEST	Switch to stop iteration on I1

A flow diagram for the program is shown below.



The MAD program is shown below.

```
            PRINCIPAL VALUES AND DIRECTIONS OF A SECOND ORDER SYMMETRIC
            TENSOR
            INTEGER N, TEST
            READ FORMAT INPUT,XX,XY,XZ,YY,YZ,ZZ
            PRINT FORMAT DATA,XX,XY,XZ,YY,YZ,ZZ
            ONE=XX+YY+ZZ
            TWOHAF=0.5*(XX*XX+YY*YY+ZZ*ZZ)+XY*XY+XZ*XZ+YZ*YZ
            TWO=0.5*ONE*ONE-TWOHAF
            THREE=XX*(YY*ZZ-YZ*YZ)+XY*(-ZZ*XY+YZ*XZ)+XZ*(XY*YZ-YY*XZ)
            WHENEVER ONE.NE.0.
            I=ONE
            OTHERWISE
            I=TWO
            END OF CONDITIONAL
            TEST=0
            THROUGH ROOT,FOR N=1,1,TEST.E.1
            IN=I-(I*(TWO+I*(I-ONE))-THREE)/(I*(3.0*I-2.0*ONE))
            WHENEVER .ABS.(1.0-IN/I).L..00000003, TEST=1
ROOT        I=IN
            I1=I
            C1=ONE-I1
            C12=0.5*C1
            C2=TWO-I1*C1
            R=SQRT.(C12*C12-C2)
            I2=C12+R
            I3=C12-R
            BX=(XZ*XY+YZ*(-XX+I1))/(XY*YZ-XZ*(YY-I1))
            BY=(XZ*XY+YZ*(-XX+I2))/(XY*YZ-XZ*(YY-I2))
            BZ=(XZ*XY+YZ*(-XX+I3))/(XY*YZ-XZ*(YY-I3))
            GX=((XX-I1)*(YY-I1)-XY*XY)/(XY*YZ-XZ*(YY-I1))
            GY=((XX-I2)*(YY-I2)-XY*XY)/(XY*YZ-XZ*(YY-I2))
            GZ=((XX-I3)*(YY-I3)-XY*XY)/(XY*YZ-XZ*(YY-I3))
            AXX=1.0/SQRT.(1.0+BX*BX+GX*GX)
            AYX=1.0/SQRT.(1.0+BY*BY+GY*GY)
            AZX=1.0/SQRT.(1.0+BZ*BZ+GZ*GZ)
            AXY = BX*AXX
            AYY=BY*AYX
            AZY = BZ*AZX
            AXZ = GX*AXX
            AYZ = GY*AYX
            AZZ=GZ*AZX
            CHECK1 = AXX*AYX + AXY*AYY + AXZ*AYZ
            CHECK2 = AXX*AZX + AXY*AZY + AXZ*AZZ
            CHECK3 = AYX*AZX + AYY*AZY + AYZ*AZZ
            CHECK4 = AXX*AXY + AYX*AYY + AZX*AZY
            CHECK5 = AXX*AXZ + AYX*AYZ + AZX*AZZ
            CHECK6 = AXY*AXZ + AYY*AYZ + AZY*AZZ
            PRINT FORMAT OUT1,I1,I2,I3
            PRINT FORMATOUT15,ONE,TWO,THREE
            PRINT FORMAT OUT2
            PRINT FORMAT OUT3,AXX,AXY,AXZ
            PRINT FORMAT OUT3,AYX,AYY,AYZ
            PRINT FORMAT OUT3,AZX,AZY,AZZ
            PRINT FORMAT OUT
            PRINT FORMAT OUT3, CHECK1, CHECK2, CHECK3
            PRINT FORMAT OUT3, CHECK4, CHECK5, CHFCK6
            VECTOR VALUES DATA=$1H0,S2,5H XX = E14.8,6H, XY = E14.8,6H, X
           1Z = E14.
           28,6H, YY = E14.8,6H, YZ = E14.8,6H, ZZ = E14.8*$
            VECTOR VALUES OUT1=$1H0,S2,42HTHE PRINCIPAL VALUES ARE, IN X,
           1Y,Z ORDER, E14.8,2H, E14.8,2H, E14.8*$
            VECTOR VALUES OUT15 =
           1                   $1H0,S2,45HTHE THREE INVARIANTS ARE, IN 1, 2,
           2 3, ORDER    3(E14.8,2H, ) *$
            VECTOR VALUES OUT2=$1H0,S2,70HTHE DIRECTION COSINES AXX, AXY,
           1 AXZ, AYX, AYY, AYZ, AZX, AZY, AZZ ARE                *$
            VECTOR VALUES OUT3=$1H0,S9,2(E14.8,2H, ),E14.8 *$
            VECTOR VALUES OUT=$1H0,S2,52HTHE CHECK ON ORTHOGONALITY OF TH
           1E PRINCIPAL AXES IS
            VECTOR VALUES INPUT =$3E14.8*$
            END OF PROGRAM
```

A typical set of computer output is shown below.

```
XX = .10880952E 02, XY = .25714280E 01, XZ = .42857140E 00,
YY = .57380951E 01, YZ = .85714280E 00, ZZ = .13738095E 02  OE
THE PRINCIPAL VALUES ARE, IN X,Y,Z ORDER,  .11696239E 02,  .14031550E 02,  .46293523E 01

THE THREE INVARIANTS ARE, IN 1, 2, 3, ORDER   .30357143E 02,  .28321937E 03,  .75975236E 03,

THE DIRECTION COSINES AXX, AXY, AXZ, AYX, AYY, AYZ, AZX, AZY, AZZ ARE

       .88412448E 00,  .33466290E 00, -.32607464E 00

       .27816949E 00,  .18368832E 00,  .94280449E 00

       .37541855E 00, -.92426053E 00,  .69306377E-01

THE CHECK ON ORTHOGONALITY OF THE PRINCIPAL AXES IS

      -.14510006E-04,  .19716099E-05, -.35120174E-05

      -.44070184E-05, -.12225704E-04, -.60535967E-07
```