MCR 76-217
Contract NAS8-31376

Volume I

Programming Manual

May 1976

Expansion and Improvement of the FORMA System for Response and Load Analysis

[illegible]
R. [illegible]
[illegible]
[illegible]

MARTIN MARIETTA

MCR-76-217

Contract NAS8-31376

EXPANSION AND IMPROVEMENT OF THE FORMA SYSTEM FOR RESPONSE AND LOAD ANALYSIS

Volume I - Programming Manual

May 1976

Author:

Richard L. Wohlen

Approved:

Richard L. Wohlen

Richard L. Wohlen
Program Manager

Prepared for: National Aeronautics and Space Administration
George C. Marshall Space Flight Center
Huntsville, Alabama 35812

MARTIN MARIETTA CORPORATION
Denver Division
Denver, Colorado 80201

FOREWORD

This report presents results of the expansion and improvement of the FORMA system for response and load analysis. The acronym FORMA stands for FORTRAN Matrix Analysis. The study, performed from 16 May 1975 through 17 May 1976 was conducted by the Analytical Mechanics Department, Martin Marietta Corporation, Denver Division, under the contract NAS8-31376. The program was administered by the National Aeronautics and Space Administration, George C. Marshall Space Flight Center, Huntsville, Alabama under the direction of Dr. John R. Admire, Structural Dynamics Division, Systems Dynamics Laboratory.

This report is published in seven volumes:

Volume I - Programming Manual,
Volume IIA - Listings, Dense FORMA Subroutines,
Volume IIB - Listings, Sparse FORMA Subroutines,
Volume IIC - Listings, Finite Element FORMA Subroutines,
Volume IIIA - Explanations, Dense FORMA Subroutines,
Volume IIIB - Explanations, Sparse FORMA Subroutines, and
Volume IIIC - Explanations, Finite Element FORMA Subroutines.

CONTENTS

ABSTRACT

This report presents techniques for the solution of structural dynamic systems on an electronic digital computer using FORMA (FORTRAN Matrix Analysis).

FORMA is a library of subroutines coded in FORTRAN IV for the efficient solution of structural dynamics problems. These subroutines are in the form of building blocks that can be put together to solve a large variety of structural dynamics problems. The obvious advantage of the building block approach is that programming and checkout time are limited to that required for putting the blocks together in the proper order.

The FORMA method has advantageous features such as:

1. subroutines in the library have been used extensively for many years and as a result are well checked out and debugged;
2. method will work on any computer with a FORTRAN IV compiler;
3. incorporation of new subroutines is no problem;
4. basic FORTRAN statements may be used to give extreme flexibility in writing a program.

Two programming techniques are used in FORMA: dense and sparse.

ACKNOWLEDGMENTS

The editor expresses his appreciation to those individuals whose assistance was necessary for the successful completion of this report. Dr. John R. Admire was instrumental in the definition of the program scope and contributed many valuable suggestions. Messrs. Carl Bodley, Wilcomb Benfield, Darrell Devers, Richard Hruda, Roger Philippus, and Herbert Wilkening, all of the Analytical Mechanics Department, Denver Division of Martin Marietta Corporation, have contributed ideas, as well as subroutines, in the formulation of the FORMA library.

The editor also expresses his appreciation to those persons who developed FORTRAN, particularly the subroutine concept of that programming tool.

SUMMARY

The formulation and solution of most structural dynamics problems involves the use of matrix analysis and an electronic digital computer. Matrix analysis is used because it allows complicated arithmetical operations to be formulated systematically and provides a compact form of bookkeeping. The electronic digital computer is used in the solution of the problem because of its low cost per calculation.

After the analyst has formulated a problem in matrix notation, he is faced with the practical consideration of obtaining numerical answers using numerical input to the equation. The analyst must therefore translate (i.e., program) the equations into a form recognizable by the computer. Two computer programming approaches are available to the analyst. One is to program the computer to solve a specific type problem using a basic programming language such as ALGOL or FORTRAN. This approach can yield a very efficient computer program but the development of such a program is very time consuming. Thus, such an approach is practical only if the program will be used extensively. The second approach involves a library of matrix analysis operations in subroutine form that allows the analyst to set up his own program using a "building block" concept. This second approach allows the acquisition of quick results from problems of quite different types and is the approach considered in this report.

The validity of the second approach becomes evident from a study of structural dynamic analysis methods. This study reveals that for most types of problems, the mathematical operations required for solutions are limited in number. Thus, these mathematical operations can be programmed in the form of computer subroutines resulting in a library of "building blocks" that can be put together to solve a large variety of structural dynamics problems. The obvious advantage of the building block approach is that the only programming and checkout time required is putting the necessary blocks together in the proper order.

The building block approach described in this report uses FORTRAN call statements with subroutines from a library of subroutines entitled FORMA (FORTRAN Matrix Analysis). Development of subroutines in the FORMA library was started in 1964 by engineers in the Dynamics and Loads Section of Martin Marietta Corporation, Denver Division, to solve a wide variety of structural dynamics analyses of aerospace vehicles such as the Titan booster and Skylab orbiting laboratory. These subroutines were programmed specifically for the solution of small and medium size structural dynamics problems of up to approximately 150 degrees of freedom. Since this beginning, the FORMA library has been expanded to include the solution of large size structural dynamics problems of up to approximately 6000 degrees of freedom. These subroutines for the analysis of large size structures have been used by engineers in the Dynamics

and Loads Section in the analysis of Viking and Space Shuttle. The FORMA library as included here consists of over 200 subroutines. Listing and explanations of these subroutines are given in Volumes II and III respectively. A division is made in those two volumes for dense programming logic subroutines, sparse programming logic subroutines and finite element subroutines.

The FORMA library includes subroutines for mass matrix calculations, stiffness matrix calculations, vibration modal solutions, time response solutions as well as the basic matrix algebra subroutines. A list of available subroutines is given in Appendices A, B, and C of this volume.

The subroutines in this library have been used extensively and as a result are well checked out and debugged. The FORMA method has advantageous features such as:

1. method will work on any computer with a FORTRAN IV compiler. It has been used on the IBM 7044, IBM 7094, GE 625/635, CDC 6400/6500, and UNIVAC 1108 with only minor modifications;

2. computer times are reasonable;

3. incorporation of new subroutines is no problem;

4. basic FORTRAN statements may be used to give extreme flexibility in writing a program;

5. an analyst can program relatively complex problems with very little programming experience; and

6. the method of programming is closely related to the manner of the mathematical formulation of the physical problem.

In conclusion, this report expands and improves the FORMA system for response and loads analysis by combing existing and adding new dense, sparse and finite element subroutines to the FORMA library. Modifications for MSFC requirements are included where necessary

I. INTRODUCTION

This volume presents the programming techniques and summarizes the subroutines available in the FORMA library that will enable an analyst to convert his matrix equations into a computer program. It is assumed that the analyst has a basic knowledge of Fortran.

Using the FORMA method, a computer program is coded using CALL statements for the desired subroutines from the FORMA library. Two programming techniques (dense and sparse) are utilized to describe the matrices. In dense programming, all elements of the matrix, both zero and non-zero, are used. The maximum size of a matrix is, thus, limited by the core size of the computer. For example, with an available computer core size of 50,000, the maximum square matrix size is approximately 150 (when two matrices are used). To get around this size restriction, a sparse programming technique was devised. In sparse programming (subroutines begin with the letter "Y") only the non-zero matrix elements are used. In the sparse technique, the matrix size is nominally unlimited because partitions of a matrix are stored on disk.

A list of available subroutines is included in Appendix A (dense), Appendix B (sparse), and Appendix C (finite element) grouped according to function (e.g., input, output, algebraic calculation, etc.).

As with all skills, the more experienced and skillful the analyst is, the "better" the FORMA program he will code. A "better" program is defined to be one that has the maximum possible matrix sizes, checks the input data for mistakes (where possible), and uses the least computer time. Probably the best means of improving FORMA skills is by becoming familiar with Fortran capabilities through reading of a Fortran coding manual. It should be emphasized, however, that any FORMA program will work, some programs are just "better" than others.

II. PROGRAMMING TECHNIQUE (DENSE PROGRAMMING LOGIC)

1. Transfer of Data

Transfer of matrix data to and from the subroutines is made by subroutine arguments. Transfer of page heading data is made by a labeled COMMON block as explained in subroutine START.

Input matrix data for programs using dense FORMA subroutines are read using Subroutine READ for real numbers (a Fortran term for numbers with a decimal point) or Subroutine READIM for integer numbers. A special-purpose subroutine (READO is available but is not needed for most programs. The only other subroutines that read input data are (a) Subroutine START 3 cards for (1) runs number, and user's name, (2) title card 1, and (3) title card 2 ; (b) Subroutine COMENT for comment cards; (c) Subroutine UPDATE for tape updating data; (d) Subroutine RBTTAB for data defining degrees of freedom and coordinate locations for a structural system. No other subroutines read input data.

Printed output data for programs using dense FORMA subroutines are generally obtained by using Subroutine WRITE for real numbers or Subroutine WRITIM for integer numbers. Exceptions to this are the time response subroutines and frequency response subroutines. Here the volume of calculated data is too great to be transferred out of the subroutine and is automatically printed in the subroutine. Other exceptions are CKMAS1, CKSTF1 and RBTTAB which provide specially formatted output.

In the development of FORMA it was recognized that the matrix sizes and row dimensions could be eliminated from the subroutine arguments to give simpler CALL statements. However, by doing this, considerable programing skill is then required by the analyst if in his program he wishes to refer to a particular element of a matrix. Considering the advantages and disadvantages of (1) more arguments in CALL statement but easy matrix element referral in main program against (2) less arguments in CALL statement but difficult matrix element referral in main program, it was decided to use the first approach.

2. Coding Procedure - Sample Problem 1

Perhaps the best means of demonstrating the use of FORMA is through a sample problem. Assume the matrix equation

$$[Z]_{N1xN3} = \left(3.\ [P]_{N1xN2} + [Q]_{N1xN2}\right) [R]_{N2xN3} \qquad (1)$$

is to be programed. Matrices [P], [Q], and [R] are to be input data to the program. The answer matrix [Z] is to be printed. The maximum sizes expected are N1 = 50, N2 = 45, and N3 = 60. However, the particular sizes of N1, N2, and N3 will be determined at run time and could be any value between 1 and the maximum size expected.

The following steps are used to code the program. The program will be written on a sheet of coding paper to facilitate keypunching the information to cards. A typical coding sheet with the steps listed below is shown in Figure 1.

The names for data in a program are alphanumeric, but the first character must be alphabetic. A first letter of I, J, K, L, M, or N indicates an integer, while the rest of the alphabet in the first letter indicates a real number.

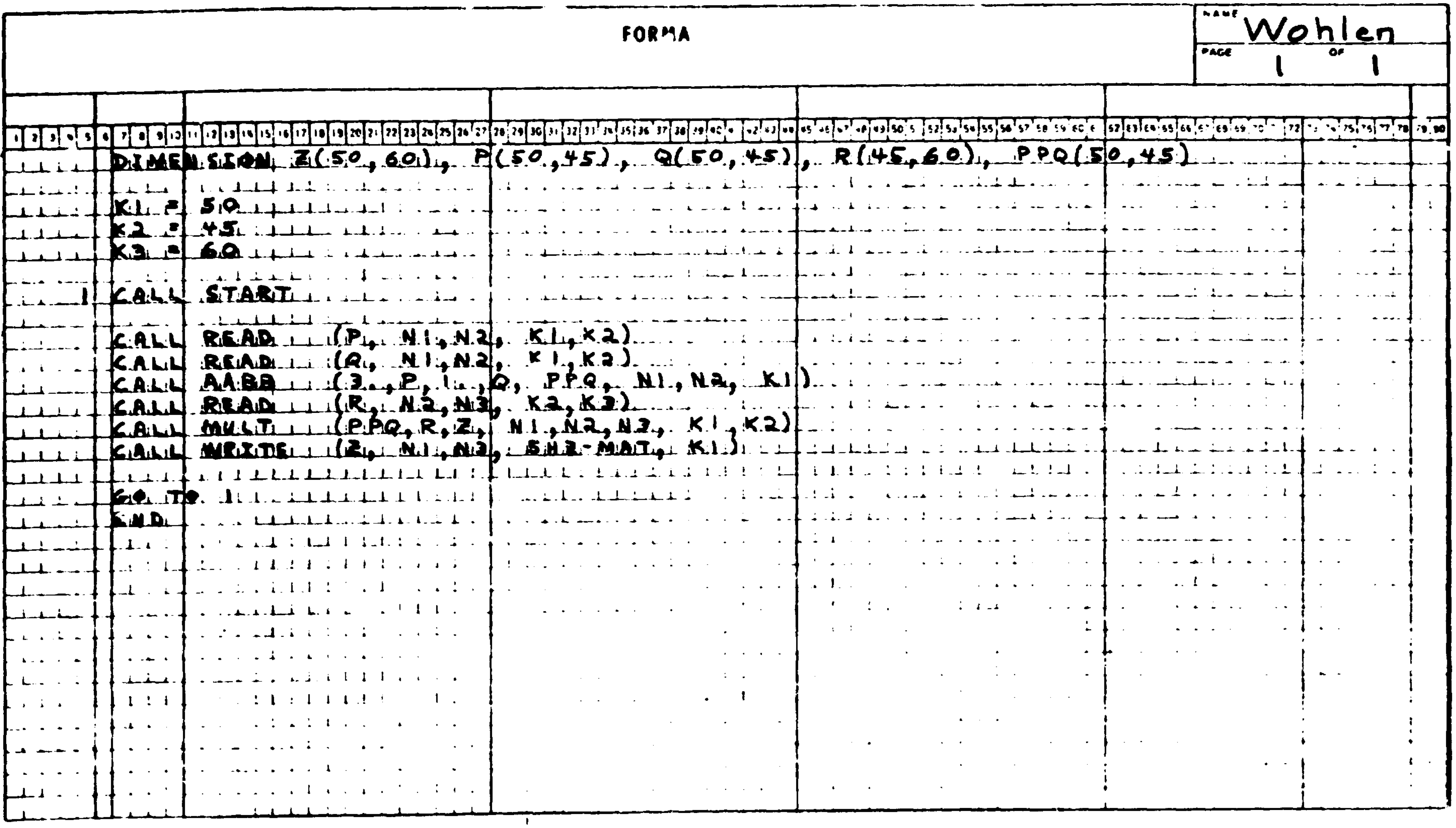
Step (1) - Call Subroutine START to read three input data cards for (1) run number and user's name, (2) title card 1, and (3) title card 2.

Step (2) - Write the CALL statements based on the above equation (1) using the subroutines listed in Appendix A. This is shown in Figure 1 where K1 is a symbol used to designate the maximum size expected for N1. Similarly for K2,N2 and K3,N3.

Step (3) - Write the DIMENSION statements for the matrices. This indicates the maximum size expected for each matrix. Note that an intermediate matrix [PPQ] = 3. [P] + [Q] is formed in Subroutine AABB and must be dimensioned. The numerical values for K1, K2, and K3 are also defined.

Step (4) - Shift back to Subroutine START by using the Fortran statement GO TO 1. This procedure allows for "stacked" problems. The run is terminated by a STOP data card (see Subroutine START writeup) after the data of the last problem.

Step (5) - The end of the Fortran source deck is indicated with the Fortran statement END.



FORMA

NAME Wohlen
PAGE 1 OF 1

```
Step (3)  {       DIMENSION Z(50,60), P(50,45), Q(50,45), R(45,60), PPQ(50,45)
          {
          {       K1 = 50
          {       K2 = 45
          {       K3 = 60

Step (1)        1 CALL START

Step (2)  {       CALL READ   (P, N1,N2, K1,K2)
          {       CALL READ   (Q, N1,N2, K1,K2)
          {       CALL AABB   (3.,P,1.,Q, PPQ, N1,N2, K1)
          {       CALL READ   (R, N2,N3, K2,K3)
          {       CALL MULT   (PPQ,R,Z, N1,N2,N3, K1,K2)
          {       CALL WRITE  (Z, N1,N3, SHZ-MAT, K1)

Step (4)          GO TO 1
Step (5)          END
```

FIGURE 1. FORMA COMPUTER PROGRAM FOR SAMPLE PROBLEM 1

The input data to the sample problem is also written on a coding form and is shown in Figure 2. The input matrices are assumed to be:

$$[P]_{2x3} = \begin{bmatrix} 1. & 2. & 3. \\ 4. & 5. & 6. \end{bmatrix},$$

$$[Q]_{2x3} = \begin{bmatrix} 7. & 8. & 9. \\ 0. & 0. & 0. \end{bmatrix},$$

$$[R]_{3x6} = \begin{bmatrix} 10. & 11. & 12. & 13. & 14. & 15. \\ 0. & 0. & 0. & 0. & 0. & 26. \\ 31. & 0. & 33. & 0. & 35. & 0. \end{bmatrix}.$$

The first three cards of input data contain the following information:

Card 1: Run number in columns 1-6. User's name in columns 11-28.

Card 2: Title 1 in columns 1-72.

Card 3: Title 2 in columns 1-72.

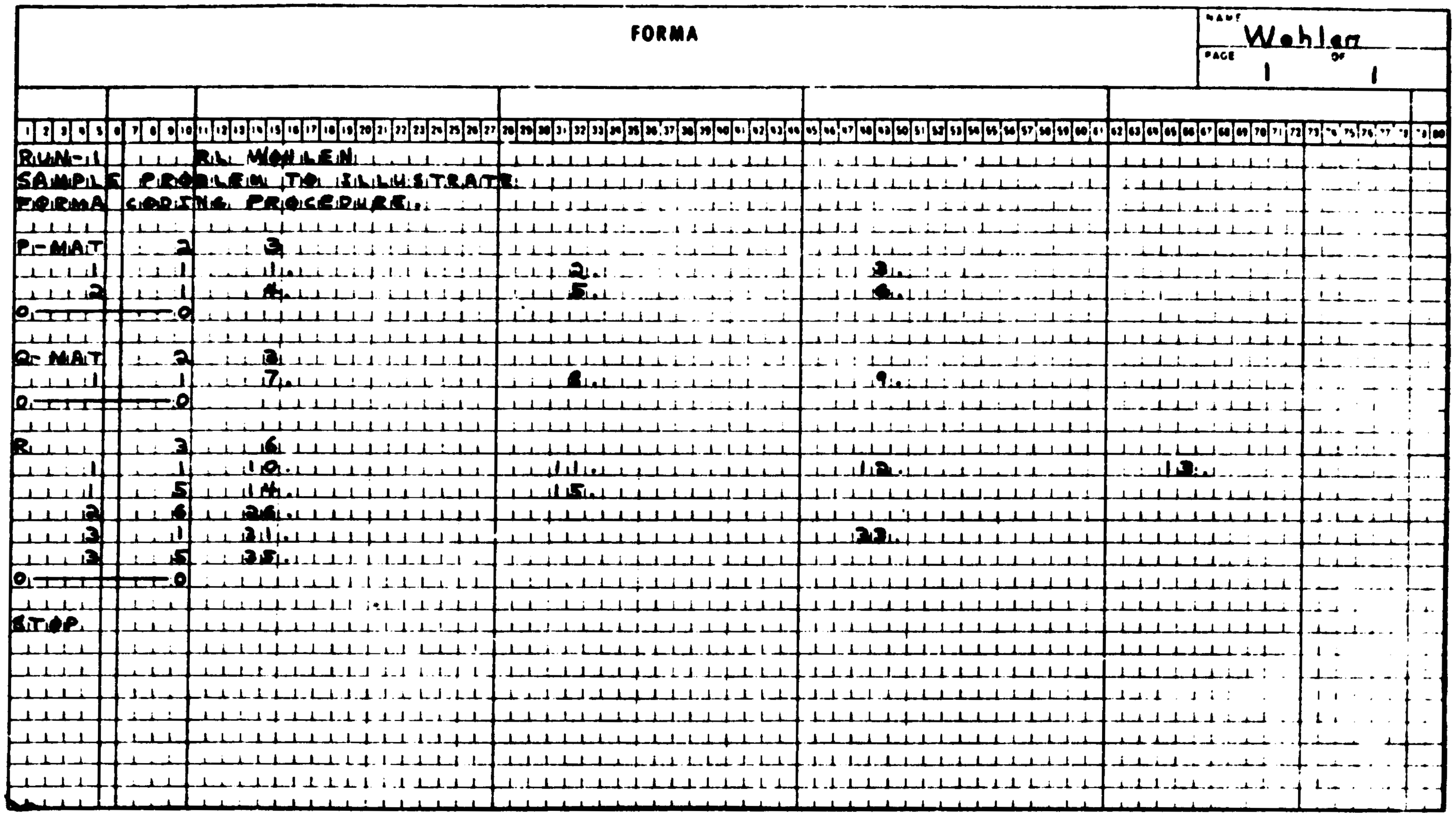
The input form for each matrix is:

First Card: Matrix name in columns 1-6. Matrix row size in columns 7-10 (right justified). Matrix column size in columns 11-15 (right justified).

Middle Cards: Matrix row number in columns 1-5 (right justified) of data. Matrix column number in columns 6-10 (right justified) of data in next field. Matrix data in four fields in columns 11-27, 28-44, 45-61, and 62-78.

Last Card: Ten zeros in columns 1-10.

The last input card is STOP in columns 1-4.



FORMA

NAME Wohlen

PAGE 1 OF 1

```
RUN-1     RL WOHLEN
SAMPLE PROBLEM TO ILLUSTRATE
FORMA CODING PROCEDURE.

P-MAT    2    3
    1    1    1.               2.               3.
    2    1    4.               5.               6.
0--------0

Q-MAT    2    3
    1    1    7.               8.               9.
0--------0

R        3    6
    1    1   10.              11.              12.              13.
    1    5   14.              15.
    2    6   26.
    3    1   31.                               33.
    3    5   35.
0--------0

STOP
```

FIGURE 2. INPUT DATA FOR SAMPLE PROBLEM 1

3. Coding a Better Program

If the analyst is satisfied with the program he has coded, this section can be skipped. However, if large size matrices are to be used or if it is desired to check the sizes of the input matrices then this section should be consulted.

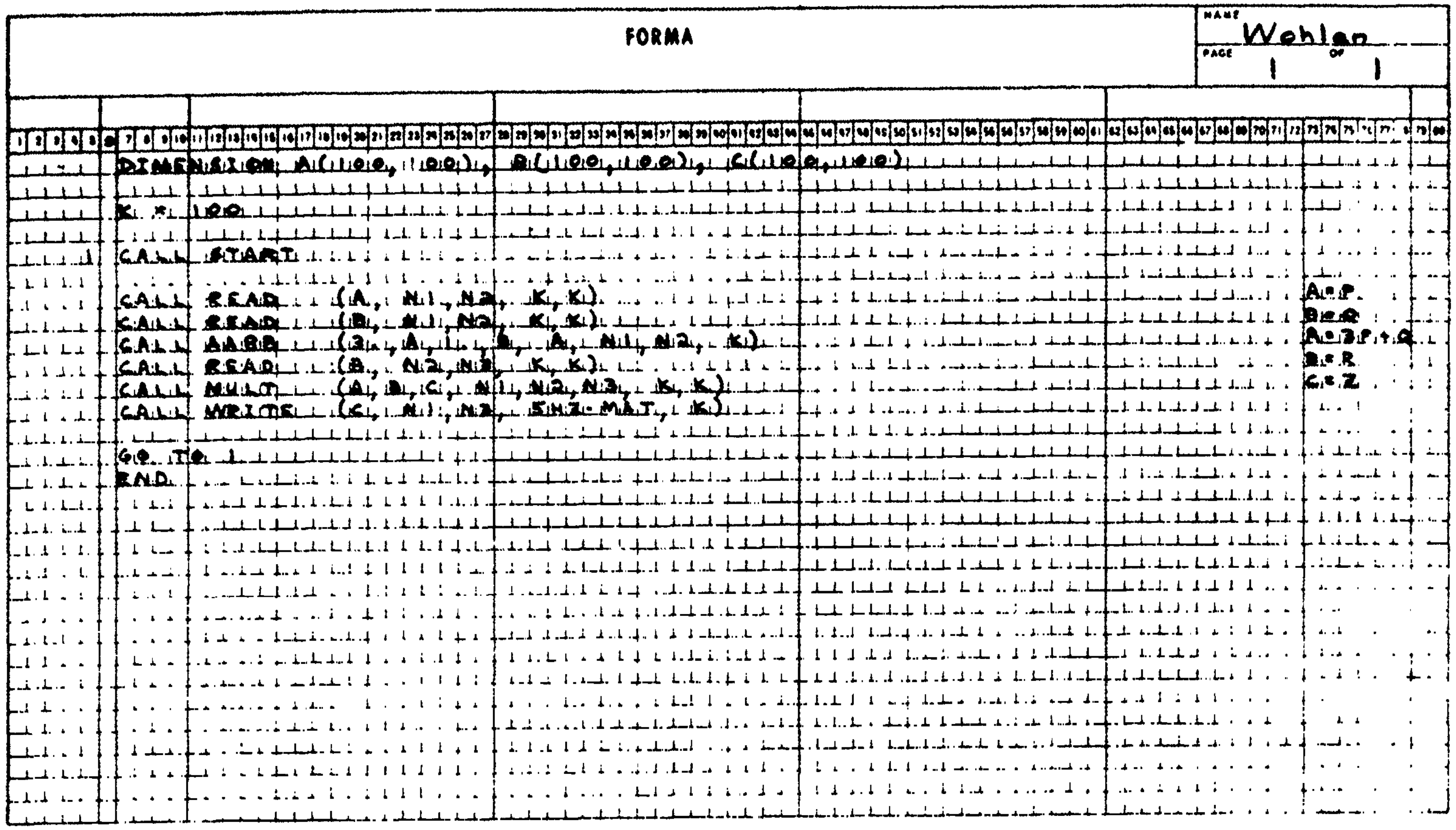
Equivalence - If the analyst wished to increase the maximum expected sizes in the program of Figure 1 to K1 = K2 = K3 = 100, then 50,000 core locations would be required for the matrices alone. If this size requirement exceeds the capacity of the computer being used, then core locations will have to be shared between matrices where possible. This is accomplished by using Fortran EQUIVALENCE. Equivalencing is a very sensitive operation because it is easy to wipe out numbers of a matrix before being finished with the matrix. Mistakes of this type will not stop the running of the problem and can only be noticed (hopefully!) by "incorrect-looking" answers.

There are several methods of equivalencing. In the first method, the various matrices are equivalenced to locations in a large dummy matrix. In the second method, the various matrices are equivalenced to each other. The third method is a "manual equivalencing" procedure and is recommended over the other two methods because it is easier to code and understand. In this manual equivalencing method, only two or three matrices are dimensioned [e.g., DIMENSION A(100,100), B(100,100), C(100,100)]. The entire program is coded using only the names A, B, or C. The particular meaning of A, B, or C should then be given in Columns 73 thru 80. By this third method, the EQUIVALENCE statements are kept to a minimum and may not be needed at all.

It is advisable to equivalence only the larger matrices of a program. The possibility of mistake introduced by equivalencing scalars and the smaller matrices is not worth the small amount of core that will be saved.

To demonstrate the manual equivalencing method, assume the dimensions of the program of Figure 1 are to be increased to K1 = K2 = K3 = 100. Assume that the resulting 50,000 core locations exceeds the capacity of the computer being used.

In the first example of manual equivalencing, Subroutine MULT is retained in the program. Thus a minimum of three matrices will be needed. The resulting program is shown in Figure 3. The core requirements for this program are only 30,000 for the matrices. The input data of Figure 2 is still the same. The same results as the program shown in Figure 1 will still be obtained.



FORMA

NAME Wehlan

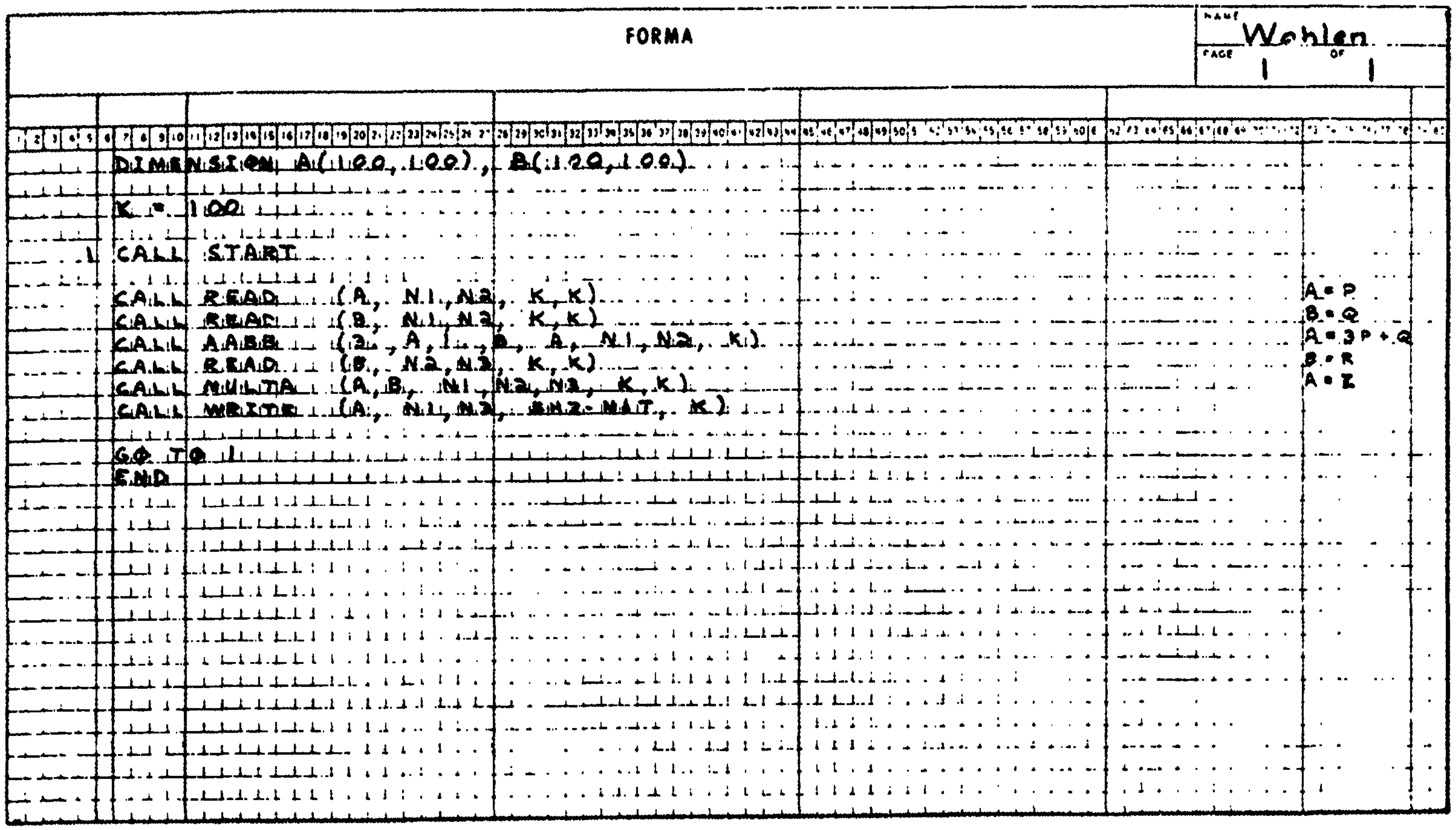
PAGE 1 OF 1

```
      DIMENSION A(100,100), B(100,100), C(100,100)

      K = 100

    1 CALL START

      CALL READ   (A, N1, N2, K, K)                                     A=P
      CALL READ   (B, N1, N2, K, K)                                     B=Q
      CALL AABB   (3., A, 1., B, A, N1, N2, K)                          A=3P+Q
      CALL READ   (B, N2, N3, K, K)                                     B=R
      CALL MULT   (A, B, C, N1, N2, N3, K, K)                           C=Z
      CALL WRITE  (C, N1, N3, 5HZ-MAT, K)

      GO TO 1
      END
```

FIGURE 3. FORMA COMPUTER PROGRAM FOR SAMPLE PROBLEM 1
THREE MATRICES DIMENSIONED

In the second example of manual equivalencing, the program of Figure 3 is modified to use only two matrices. This requires that Subroutine MULT be replaced with either Subroutine MULTA or MULTB. The resulting program is shown in Figure 4. The core requirements for this program are only 20,000 for the matrices. The input data of Figure 2 is still the same. The same results as the program shown in Figure 1 will still be obtained.

Size Check - Any of the three programs just coded will run even if there is a mistake in the matrix sizes in the input data. The modifications to the program of Figure 4 to check the sizes of the input matrix data are shown in Figure 5. This is the "best" program for the sample problem given by equation (1). The input data of Figure 2 is still the same. The same results as the program shown in Figure 1 will still be obtained.

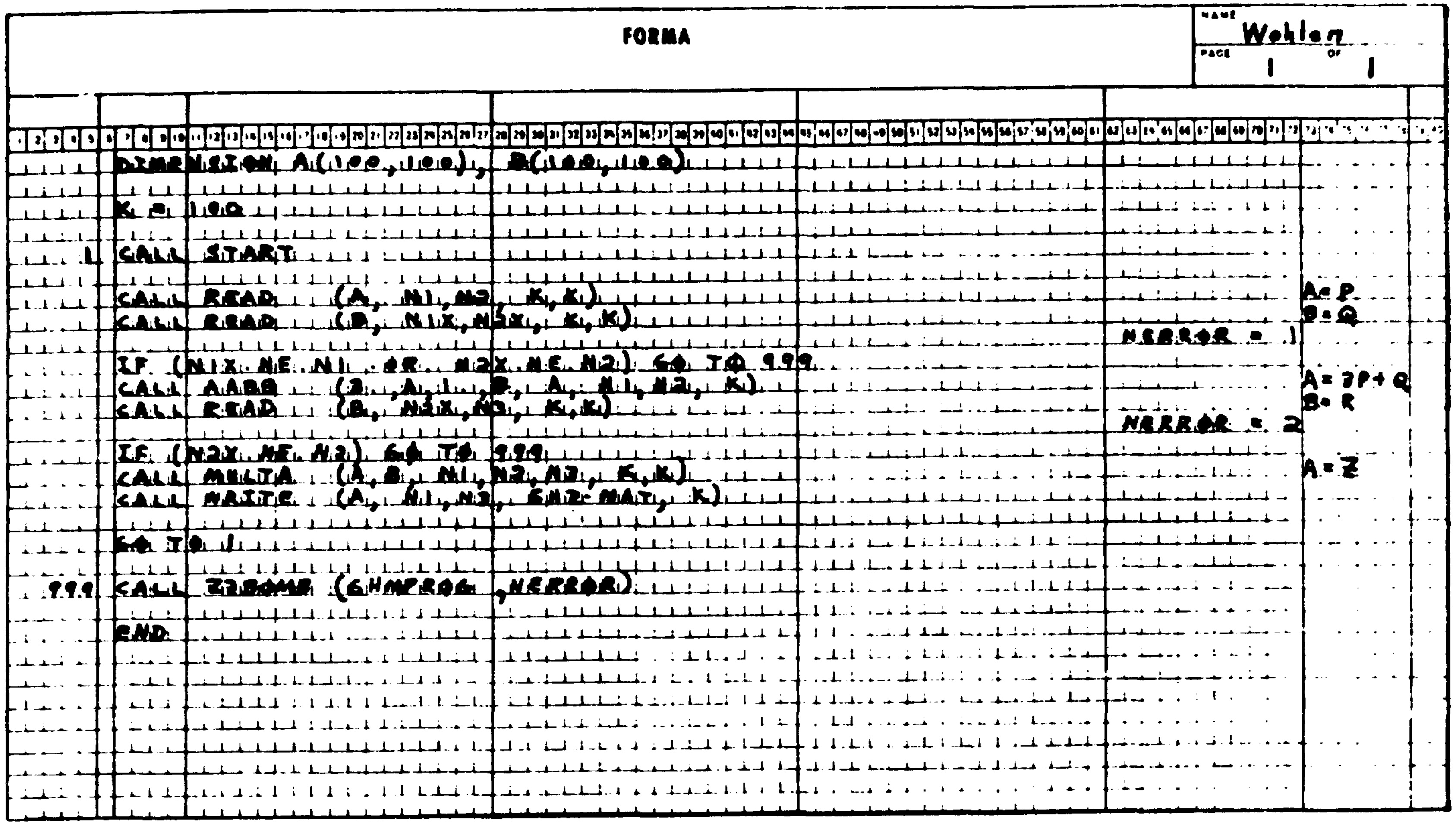

FORMA

NAME Wohlen

PAGE 1 OF 1

```
      DIMENSION A(100,100), B(100,100)

      K = 100

    1 CALL START

      CALL READ   (A, N1,N2, K,K)                                       A = P
      CALL READ   (B, N1,N2, K,K)                                       B = Q
      CALL AABB   (2., A, 1., B, A, N1,N2, K)                           A = 3P + Q
      CALL READ   (B, N2,N3, K,K)                                       B = R
      CALL MULTA  (A,B, N1,N2,N3, K,K)                                  A = Z
      CALL WRITE  (A, N1,N3, ENZ-MAT, K)

      GO TO 1
      END
```

FIGURE 4. FORMA COMPUTER PROGRAM FOR SAMPLE PROBLEM 1
TWO MATRICES DIMENSIONED



FORMA

NAME Wohlen

PAGE 1 OF 1

```
      DIMENSION A(100,100), B(100,100)

      K = 100

    1 CALL START

      CALL READ   (A, N1,N2, K,K)                              A=P
      CALL READ   (B, N1X,N2X, K,K)                            B=Q
                                                   NERRØR = 1
      IF (N1X.NE.N1 .ØR. N2X.NE.N2) GØ TØ 999
      CALL AABB   (3.,A,1.,B, A, N1,N2, K)                     A=3P+Q
      CALL READ   (B, N2X,N3, K,K)                             B=R
                                                   NERRØR = 2
      IF (N2X.NE.N2) GØ TØ 999
      CALL MULTA  (A,B, N1,N2,N3, K,K)                         A=Z
      CALL WRITE  (A, N1,N3, 6H2-MAT, K)

      GØ TØ 1

  999 CALL ZZBOMB (6HMPROG ,NERRØR)

      END
```

FIGURE 5. FORMA COMPUTER PROGRAM FOR SAMPLE PROBLEM 1
TWO MATRICES DIMENSIONAL
SIZE CHECKS INCORPORATED

4. Sample Problem 2

To further illustrate the use of FORMA, a second sample problem is coded in this section. The coding techniques described in Section 4 to obtain a "better" program will be used.

In this problem, the "free-free" mode shapes and frequencies of the beam shown in Figure 6 are to be calculated and printed. Two degrees of freedom, translation and rotation, are assumed at each of the five panel points (also called collocation points). The input data to the computer program are the beam panel points, the beam weight distribution, and the beam stiffness distribution. For this sample problem the distributed rotary inertia, any concentrated weights, and the shear stiffness are ignored.

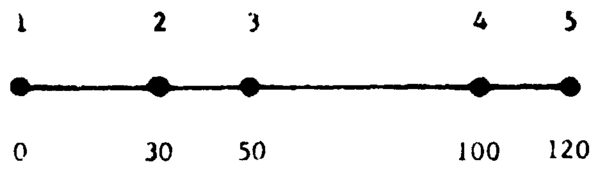
The following steps are used to code the computer program. The program will be written on a sheet of coding paper to facilitate keypunching the information to cards. A typical coding sheet with the steps listed below is shown in Figure 7.

As mentioned previously, the names for data in a program are alphanumeric, but the first character must be alphabetic. A first letter of I, J, K, L, M, or N indicates an integer, while the rest of the alphabet in the first letter indicates a real number.

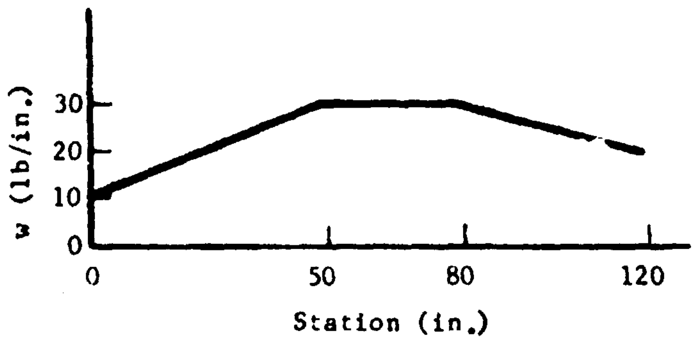
Step (1) - Call Subroutine START to read three input data cards for (1) run number and user's name, (2) title card 1, and (3) title card 2.

Step (2) - Write the CALL statements to read in the panel points, weight distribution, and stiffness distribution. Checks on the column size are made. Write the CALL statements to calculate and write the mass and stiffness matrices, and to calculate and write the mode shapes and frequencies. K1 is a symbol used to designate the maximum number of degrees of freedom allowed. K2 is a symbol used to designate the maximum number of panel points allowed. K3 is a symbol used to designate the maximum number of rows of distributed data.

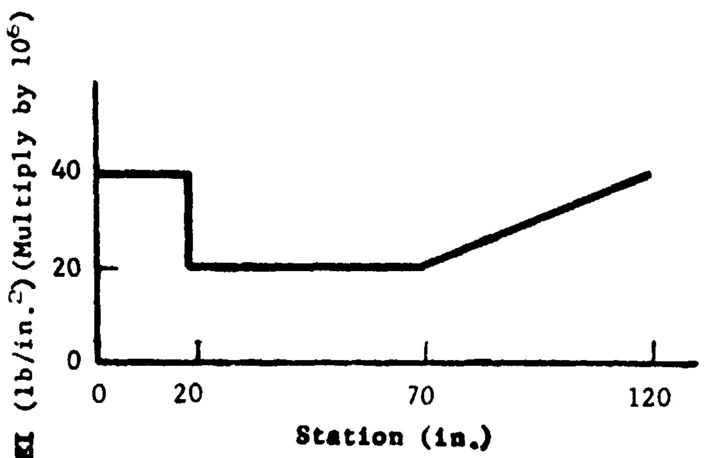
Step (3) - Write the DIMENSION statements for the matrices. This indicates the maximum size expected for each matrix. Even though this sample problem has five panel points, the computer program is written assuming that there could be as many as 50 panel points and thus 100 degrees of freedom. Also, a maximum of 40 rows of distributed data is allowed by the dimension given to the matrix D. The corresponding values for K1, K2, and K3 are defined.



(a) Panel Point Arrangment

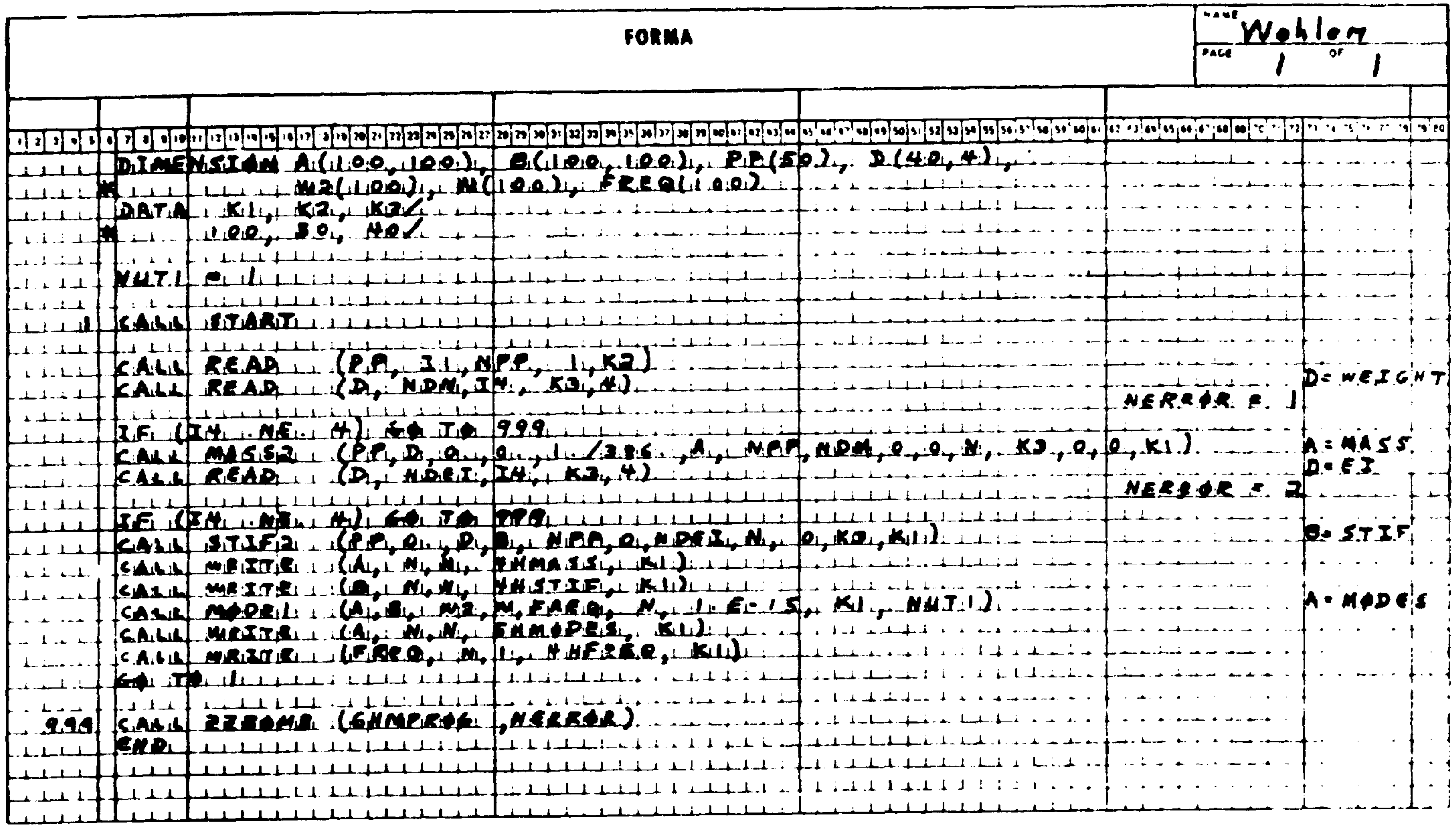


(b) Weight Distribution



(c) Stiffness Distribution

Figure 6. Beam for Sample Problem 2



FORMA

NAME Wohlen
PAGE 1 OF 1

```
      DIMENSIØN A(100,100), B(100,100), PP(50), D(40,4),
     *          M2(100), M(100), FREQ(100)
      DATA  K1, K2, K3/
     *      100, 30, 40/

      NUT1 = 1

    1 CALL START

      CALL READ   (PP, 31, NPP, 1, K2)
      CALL READ   (D, NDM, I4, K3, 4)                              D= WEIGHT
                                                   NERRØR = 1
      IF (I4 .NE. 4) GØ TØ 999
      CALL MASS2  (PP, D, 0., 0., 1./386., A, NPP, NDM, 0, 0, N, K3, 0, 0, K1)   A=MASS
      CALL READ   (D, NDEI, I4, K3, 4)                             D=EI
                                                   NERRØR = 2
      IF (I4 .NE. 4) GØ TØ 999
      CALL STIF2  (PP, 0., D, B, NPP, 0, NDEI, N, 0, K3, K1)       B= STIF
      CALL WRITE  (A, N, N, 8HMASS, K1)
      CALL WRITE  (B, N, N, 8HSTIF, K1)
      CALL MØDE1  (A, B, M2, M, FREQ, N, 1.E-15, K1, NUT1)         A= MØDES
      CALL WRITE  (A, N, N, 5HMØDES, K1)
      CALL WRITE  (FREQ, M, 1, 4HFREQ, K1)
      GØ TØ 1

  999 CALL ZZBOMB (6HMPROG ,NERRØR)
      END
```

FIGURE 7. FORMA COMPUTER PROGRAM FOR SAMPLE PROBLEM 2

Step (4) - Shift back to Subroutine START by using the Fortran statement GO TO 1. This procedure allows for "stacked" problems. The run is terminated by a STOP data card (see Subroutine START writeup) after the data of the last problem.

Step (5) - The end of the Fortran source deck is indicated with the Fortran statement END.

The input data to sample problem 2 is also written on a coding form as shown in Figure 8. The first three cards of input data contain the following information:

Card 1: Run number in columns 1-6. User's name in columns 11-28.

Card 2: Title 1 in columns 1-72.

Card 3: Title 2 in columns 1-72.

The input form for each matrix is:

First Card: Matrix name in columns 1-6. Matrix row size in columns 7-10 (right justified). Matrix column size in columns 11-15 (right justified).

Middle Cards: Matrix row number in columns 1-5 (right justified) of data. Matrix column number in columns 6-10 (right justified) of data in next field. Matrix data in four fields in columns 11-27, 28-44, 45-61, and 61-78.

Last Card: Ten zeros in columns 1-10.

The matrix data consists of:

1) Matrix of panel point stations from Figure 6(a).

2) Matrix of end point coordinates of the line segments representing the distributed weight from Figure 6(b).

3) Matrix of end point coordinates of the line segments representing the distributed bending stiffness from Figure 6(c).

The end point coordinates of each nonvertical straight line for the distributed data is given as a row in the matrix of distributed data. Each row has the form:

Matrix column 1 - station x_i, i.e., the abscissa of the line segment originating point.

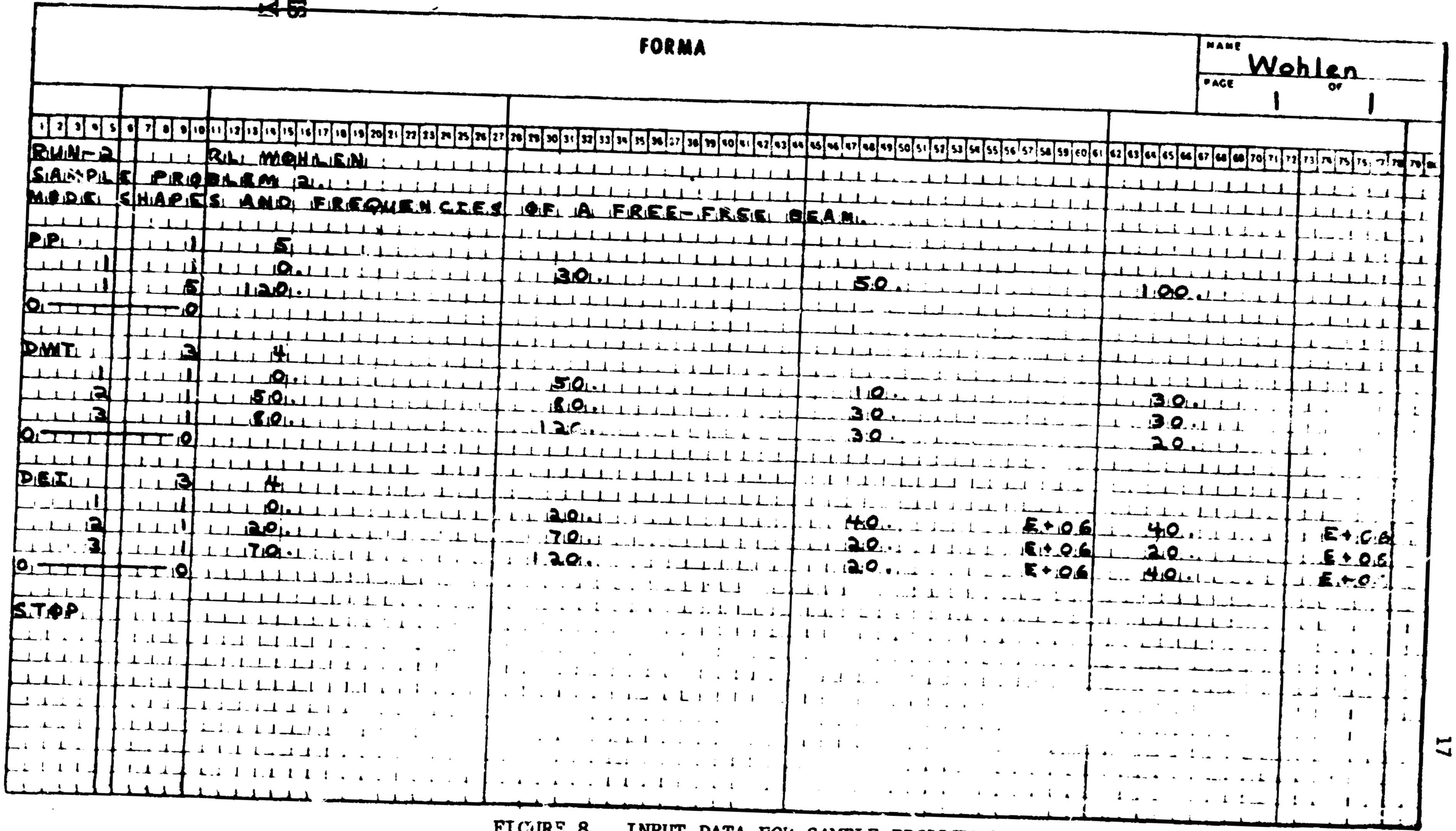
Matrix column 2 - station x_{i+1}, i.e., the abscissa of the line segment terminating point.

Matrix column 3 - value at $x_i(+)$, i.e., the ordinate of the line segment originating point.

Matrix column 4 - value at $x_{i+1}(-)$, i.e., the ordinate of the line segment terminating point.

The last input card is STOP in card columns 1-4.

ORIGINAL PAGE IS
OF POOR QUALITY



FORMA

NAME Wohlen
PAGE 1 OF 1

```
RUN-2     RL WOHLEN
SAMPLE PROBLEM 2.
MODE SHAPES AND FREQUENCIES OF A FREE-FREE BEAM.
PP       1    5
    1    1    0.            30.              50.             100.
    1    5    120.
0--------0

DMT      3    4
    1    1    0.           50.              10.               30.
    2    1    50.          80.              30.               30.
    3    1    80.         120.              30.               20.
0--------0

DEI      3    4
    1    1    0.           20.              40.        E+06   40.        E+06
    2    1    20.          70.              20.        E+06   20.        E+06
    3    1    70.         120.              20.        E+06   40.        E+0
0--------0

STOP
```

FIGURE 8. INPUT DATA FOR SAMPLE PROBLEM 2

III. PROGRAMMING TECHNIQUE (SPARSE PROGRAMMING LOGIC)

1. Transfer of Data

Matrix data for the sparse subroutines is stored on disk with a disk number(representing a matrix) being transferred to and from the subroutines by argument. Transfer of page heading data is made by a labeled COMMON block as explained in subroutine START.

Input matrix data is read using subroutine YREAD and printed output data is obtained using subroutine YWRITE.

2. Coding Procedure

The same example will be used here that was used for sample problem 1 for the dense programming logic. The example is repeated from page 3:

$$\begin{bmatrix} Z \end{bmatrix}_{N1xN3} = \left(3 \begin{bmatrix} P \end{bmatrix}_{N1xN2} + \begin{bmatrix} Q \end{bmatrix}_{N1xN2}\right) \begin{bmatrix} R \end{bmatrix}_{N2xN3} \qquad (1)$$

As before, matrices $[P]$, $[Q]$, and $[R]$ are to be input data to the program. The answer $[Z]$ is to be printed.

The following steps are used to code the program. The program is written on a sheet of coding paper to facilitate key punching the information to cards. A typical coding sheet with the steps listed below is shown in Figure 9.

Step (1) - Dimension workspaces V and LV at least 3 times the largest row or column size expected. The larger the dimension size the faster the computer time. Indicate the dimension size with KV.

Step (2) - Set the tape names to numbers.

Step (3) - Call subroutine START to read three input data cards for (1) run number and user's name, (2) title card 1, and (3) title card 2.

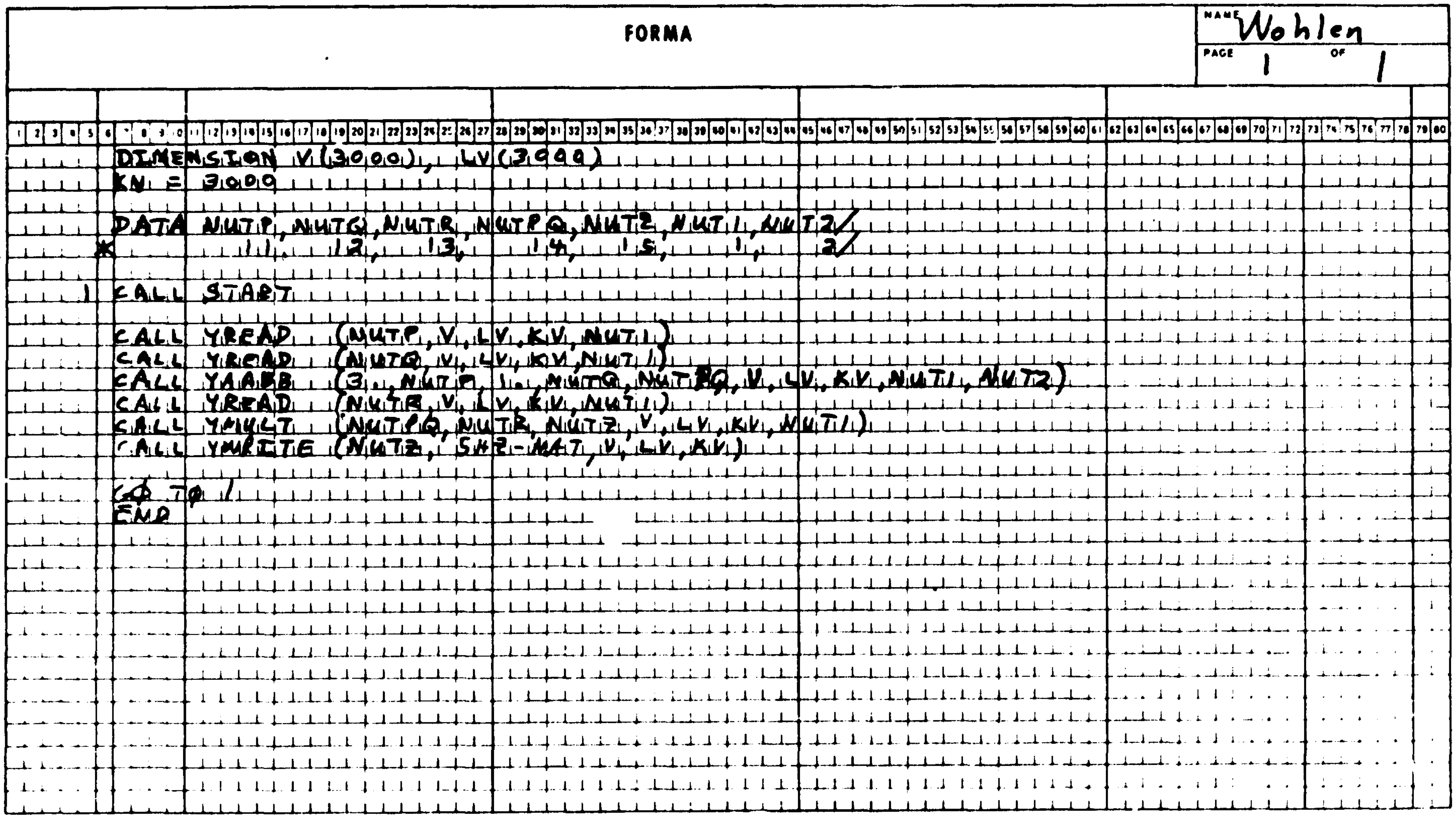
Step (4) - Write the CALL statements based on the above equation (1) using the subroutines listed in Appendix B.

Step (5) - Shift back to subroutine START by using the Fortran statement GO TO 1. This procedure allows for "stacked" problems. The run is terminated by a STOP data card (see subroutine START writeup) after the data of the last problem.

Step (6) - The end of the Fortran source deck is indicated with the Fortran statement END.

The input data for this sparse program is identical to the input data previously s own on pages 5 and 6 (Figure 2) for the dense program.

The techniques of pages 7 through 11 for coding a better dense program are not pertinent for a sparse program because equivalence is not needed. The size checks could be made with the sparse program but is not shown here.



FORMA

NAME Wohlen

PAGE 1 OF 1

```
Step (1)        DIMENSION V(3000), LV(3000)
                KN = 3000

Step (2)        DATA NUTP, NUTQ, NUTR, NUTPQ, NUT2, NUT1, NUT2/
               *        11,   12,   13,    14,   15,    1,    2/
Step (3)
             1  CALL START

Step (4)        CALL YREAD  (NUTP, V, LV, KV, NUT1)
                CALL YREAD  (NUTQ, V, LV, KV, NUT1)
                CALL YAABB  (3., NUTP, 1., NUTQ, NUTPQ, V, LV, KV, NUT1, NUT2)
                CALL YREAD  (NUTR, V, LV, KV, NUT1)
                CALL YMULT  (NUTPQ, NUTR, NUTZ, V, LV, KV, NUT1)
                CALL YWRITE (NUTZ, 5HZ-MAT, V, LV, KV)

(5)             GO TO 1
(6)             END
```

FIGURE 9. FORMA COMPUTER PROGRAM, SPARSE PROGRAMMING LOGIC

APPENDIX A. SUMMARY OF CALLING INSTRUCTIONS - DENSE FORMA SUBROUTINES.

IN THE ARGUMENTS OF THE SUBROUTINES BELOW IT IS ASSUMED THAT THERE IS CORRESPONDENCE IN SIZE AND ROW DIMENSION OF COMPATIBLE MATRICES. FOR INSTANCE IN SUBROUTINE MULT, NRA=NRZ, NCA=NRB, NCB=NCZ, KA=KZ

A 6H ARGUMENT MAY ALSO BE A VARIABLE READ WITH AN A6 FORMAT OR OBTAINED WITH A DATA STATEMENT.

*** LIST OF SYMBOLS ***

A	= INPUT MATRIX
B	= INPUT MATRIX
ALPHA	= INPUT SCALAR
BETA	= INPUT SCALAR
AVEC	= INPUT VECTOR (ROW OR COLUMN MATRIX)
IVEC	= INPUT VECTOR (ROW OR COLUMN MATRIX), INTEGER
TAB	= INPUT TABLE (MATRIX WITH INCOMPLETE COLUMNS IN SOME ROWS)
Z	= RESULT MATRIX
ZVEC	= RESULT VECTOR
N	= SIZE (FOR VECTOR OR SQUARE MATRIX)
NR	= NUMBER OF ROWS
NC	= NUMBER OF COLUMNS
K	= ROW DIMENSION
KR	= ROW DIMENSION
KC	= COLUMN DIMENSION
V	= VECTOR WORK SPACE
LV	= VECTOR WORK SPACE, INTEGER
KV	= V,LV DIMENSION
NUTI	= LOGICAL NUMBER OF ITH UTILITY TAPE
NTAPE	= SYMBOLIC NUMBER OF TAPE. FOR EXAMPLE, 1

A SINGLY DIMENSIONED VARIABLE IS REFERRED TO AS A VECTOR IN THIS REPORT.
A DOUBLY DIMENSIONED VARIABLE IS REFERRED TO AS A MATRIX IN THIS REPORT.
A VECTOR MAY BE HANDLED AS EITHER A ROW OR COLUMN MATRIX.
THE ROW DIMENSION OF A VECTOR IN THE ARGUMENTS OF THE CALL STATEMENTS IS ANALOGOUS TO THE ROWS OF THE VECTOR. THAT IS,
IF THE VECTOR IS HANDLED AS A ROW, THEN KR=1
IF THE VECTOR IS HANDLED AS A COL, THEN KR=DIMENSION SIZE

```
    EXAMPLE        DIMENSION A(10)
                   CALL READ (A, N1,N2,  1,10)
                OR CALL READ (A, N2,N1, 10, 1)
```

PRECEDING PAGE BLANK NOT FILMED

```
.01     MISCELLANEOUS

.01.01     PROGRAM INITIALIZATION
              CALL START

.01.02     PROGRAM PAGE HEADING
              CALL PAGEHD

.01.03     PROGRAM BOMBOUT
              CALL ZZBOMB (6HSUBNAM,NERROR)

.01.04     PROGRAM COMMENTS
              CALL COMENT

.01.05     CONVERSION
              CALL YDTOS   (A,NUTZ,NR,NC,KR,KC,V,LV,KV,NUTI)
                 USES YIN,YINI,YOUT,YOUTI,YPART,ZZBOMB.

.01.06     MATRIX ELEMENT COMPARISON
              CALL COMPAR (A,REF,NR,NC,NDIG,GTOL,6HANAME-,6HREFNAM,
                           KA,KREF)

.01.07     TIME CHECK
              CALL TIMCHK (6HNAMCHK)

.01.08     ORDERING
              CALL XLORD   (V,LV,LAS,NNZA)
              CALL ORDALP (IMAT,NR,NC,NCAL,IWMAT,KRI,KCW)

.01.09     MERGE NAME AND NUMBER (FUNCTION)
              NAME    (6HNAME   ,NUM)
```

```
.02     INPUT

.02.01     REAL NUMBERS
              CALL READ    (Z,*NR,*NC,KR,KC)
                 USES INTAPE,LTAPE,PAGEHD,PTAPE,WRITE,WTAPE,ZZBOMB.
                *NR,NC WILL BE DEFINED BY INPUT. USE SYMBOLS.
                                            CARD INPUT
                 FIRST CARD     - ZNAME,NR,NC WITH A6,I4,I5 FORMAT.
                                  REMARKS IN COLUMNS 16-69.
                                  $ IN COL 72 FOR WRITE TAPE INITIALIZE.
                                  BLANK,REWIND,LIST,OR TAPEID (FOR WRITE
                                  TAPE, IN COLUMNS 73-78.
                                  NWTAPE IN COLUMNS 79-80.
                 MIDDLE CARDS - DATA WITH 2I5,4E17 FORMAT.
                 LAST CARD      - TEN ZEROES IN COLUMNS 1-10.
                                            TAPE INPUT
                 ONE CARD       - ZNAME,0 OR -LOCATION,NRTAPE(WITH - FOR
                                  NO WRITE OUT),ZRUNNO WITH
                                  A6,I4,I5,A6 FORMAT.
                                  BLANK,REWIND,OR LIST (FOR READ TAPE)
                                  IN COLUMNS 22-27.
                                  REMARKS IN COLUMNS 28-69.
                                  $ IN COL 72 FOR WRITE TAPE INITIALIZE.
                                  BLANK,REWIND,LIST,OR TAPEID (FOR WRITE
                                  TAPE) IN COLUMNS 73-78.
                                  NWTAPE IN COLUMNS 79-80.

.02.02     INTEGER NUMBERS
              CALL READIM (IZ,*NR,*NC,KR,KC)
                 USES INTAPE,LTAPE,PAGEHD,PTAPE,WRITIM,WTAPE,ZZBOMB.
                *NR,NC WILL BE DEFINED BY INPUT. USE SYMBOLS.
                                            CARD INPUT
                 FIRST CARD     - IZNAME,NR,NC WITH A6,I4,I5 FORMAT.
                                  REMARKS IN COLUMNS 16-69.
                                  $ IN COL 72 FOR WRITE TAPE INITIALIZE.
                                  BLANK,REWIND,LIST,OR TAPEID (FOR WRITE
                                  TAPE) IN COLUMNS 73-78.
                                  NWTAPE IN COLUMNS 79-80.
                 MIDDLE CARDS - DATA WITH 2I5,14I5 FORMAT.
                 LAST CARD      - TEN ZEROES IN COLUMNS 1-10.
                                            TAPE INPUT
                                  (SAME AS SUBROUTINE READ ABOVE).

.02.03     OCTAL NUMBERS
              CALL READO   (Z,*NR,*NC,KR,KC)
                *NR,NC WILL BE DEFINED BY INPUT. USE SYMBOLS.
                 FIRST CARD     - ZNAME,NR,NC WITH A6,I4,I5 FORMAT.
                                  REMARKS IN COLUMNS 16-69.
                 MIDDLE CARDS - DATA WITH 2I5,3(3X,O20) FORMAT.
                 LAST CARD      - TEN ZEROES IN COLUMNS 1-10.
```

```
.02      INPUT (CONTINUED)

.02.04      ALPHA-NUMERIC CHARACTERS
               CALL READAN (IZ,*NR,*NC,KP,KC)
                  USES INTAPE,LTAPE,PAGEHD,RTAPE,WRITAN,WTAPE,ZZBOMB.
                 *NR,NC WILL BE DEFINED BY INPUT. USE SYMBOLS.
                                         CARD INPUT
                  FIRST CARD    - IZNAME,NR,NC WITH A6,I4,I5 FORMAT.
                                  REMARKS IN COLUMNS 16-69.
                                  $ IN COL 72 FOR WRITE TAPE INITIALIZE.
                                  BLANK,REWIND,LIST,OR TAPEID (FOR WRITE
                                  TAPE) IN COLUMNS 73-78.
                                  NWTAPE IN COLUMNS 79-80.
                  MIDDLE CARDS  - DATA WITH 2I5,10A6 FORMAT.
                  LAST CARD     - TEN ZEROES IN COLUMNS 1-10.
                                         TAPE INPUT
                                  (SAME AS SUBROUTINE READ ABOVE).
```

```
.03      OUTPUT

.03.01      PRINT REAL NUMBERS
               CALL WRITE   (A,NR,NC,6HANAME-,K)

.03.02      PRINT INTEGER NUMBERS
               CALL WRITIM (IA,NR,NC,6HIANAME,K)

.03.03      PRINT ALPHA-NUMERIC CHARACTERS
               CALL WRITAN (IA,NR,NC,6HIANAME,K)

.03.04      PLOT
               CALL PLOT1   (XVEC,YMAT,NRY,NCY,XSTART,XDELTA,6HXNAME-,
                             YNAME,PTITLE,IFSAME,IFCURV,IFLIFT, K)
                  USES PLOTSS.
                  HAVE -CALL BPLT (0,2HLC)- IN MP USED ONCE/RUN.
                  MAX NCY = 3
               CALL PLOT2   (XVEC,YMAT,NRY,NCY,6HXNAME-,6HYNAME-,
                             PTITLE,IPLOT,IYS, KY)
                  MAX NCY=10
               CALL PLOT3   (CLOC,MLOC,COELOC,VPLOC,PANGLE,CANGLE,
                             EEDIST,IEJNUM,LREYE,NVIEW,IEFA,PTITLE,
                             NC,NM,KC,KM)
                  USES VCROSS,VDOT.
               CALL PLOTSS (YMAX,YMIN,YTOP,TBOT)

.03.05      PUNCH
               CALL PUNCAN  (IA,NR,NC,6HIANAME,K)
               CALL PUNCH   (A,NR,NC,6HANAME-,K)
               CALL PUNCHD  (A,NR,NC,6HANAME-,K)
               CALL PUNCIM  (IA,NR,NC,6HIANAME,K)
```

```
.04      TAPE OPERATIONS

          REWIND NTAPE AT BEGINNING OF MAIN PROGRAM.

.04.01      INITIALIZATION
               CALL INTAPE  (NTAPE,6HTAPEID)

.04.02      READING
               CALL RTAPE   (6HZRUNNO,6HZNAME-,Z,*NR,*NC,KR,KC,NTAPE)
                 *NR,NC WILL BE DEFINED BY TAPE DATA. USE SYMBOLS.
                  USES LTAPE.

.04.03      WRITING
               CALL WTAPE   (A,NR,NC,6HANAME-,K,NTAPE)

.04.04      LISTING OF HEADERS
               CALL LTAPE   (NTAPE)

.04.05      UPDATING
               CALL UPDATE

.04.06      CORE/TAPE DATA TRANSFER
               CALL IN      (NTAPE,Z,N)
               CALL OUT     (NTAPE,A,N)
               CALL RWND    (NTAPE)
               CALL SKPR    (NTAPE,NREC)
```

.05 GENERATION

.05.01 BASIC

```
        CALL ONES    (Z,NR,NC,K)
        CALL SIGMA   (Z,N,K)
        CALL UNITY   (Z,N,K)
        CALL ZERO    (Z,NR,NC,K)
```

```
.06    ALGEBRA

.06.01     SCALAR PRODUCT
              CALL ALPHAA  (ALPHA,A,Z,NR,NC,KAZ)
              CALL PA      (P,A,Z,NR,NC,KA,KZ)

.06.02     ADDITION, SUBTRACTION
              CALL AAPB    (ALPHA,A,BETA,B,Z,NR,NC,KABZ)
              CALL PAQB    (P,A,Q,B,Z,NR,NC,KA,KB,KZ)

.06.03A    MULTIPLICATION - GENERAL
              CALL MULT    (A,B,Z,NRA,NRB,NCB,KAZ,KB)
              CALL MULTA   (AZ,B,NRA,NRB,NCB,KAZ,KB)
                 MAX NRB = 500
              CALL MULTB   (A,BZ,NRA,NRB,NCB,KA,KBZ)
                 MAX NRB = 500

.06.03B    MULTIPLICATION - SPECIAL
              CALL AB1     (A,B,Z,NRA,NCA,NCB,KA,KB,KZ)
                 MAX NCA = 500
              CALL AB2     (A,B,Z,NRA,NCA,NCB,KA,KB,KZ)
                 MAX NCA = 500
              CALL ABC1    (A,B,C,Z,NRA,NCA,NCB,KA,KB,KC,KZ)
                 MAX NCA = 500
              CALL ABC2    (A,B,C,Z,NRA,NCA,NCB,KA,KB,KC,KZ)
                 MAX NCA = 500
              CALL ATB1    (A,B,Z,NRA,NCA,NCB,KA,KB,KZ)
                 MAX NRA = 500
              CALL ATB2    (A,B,Z,NRA,NCA,NCB,KA,KB,KZ)
                 MAX NRA = 500
              CALL ATBC1   (A,B,C,Z,NRA,NCA,NCB,KA,KB,KC,KZ)
                 MAX NRA = 500
              CALL ATBC2   (A,B,C,Z,NRA,NCA,NCB,KA,KB,KC,KZ)
                 MAX NRA = 500
              CALL ATXBA1  (AZ,B,NRB,NCB,KAZ,KB)
                 MAX NCB = 500
              CALL ATXBB   (A,BZ,NRAT,NRB,NCB,KA,KBZ)
                 MAX NRAT = 500
              CALL ATXBB1  (A,BZ,NRB,NCB,KA,KBZ)
                 MAX NRB = 500
              CALL ATXBB2  (A,BZ,NRB,NCB,KA,KBZ)
                 MAX NCB = 500
              CALL AXBA1   (AZ,B,NRA,NCA,KAZ,KB)
                 MAX NCA = 500
              CALL AXBA2   (AZ,B,N,KAZ,KB)
                 MAX N = 500
              CALL AXBA3   (AZ,B,NRB,NCB,KAZ,KB)
                 MAX NRB = 500
              CALL DB1     (D,B,Z,NRB,NCB,KB,KZ)
```

```
.06       ALGEBRA (CONTINUED)

.06.04A   TRIPLE MATRIX PRODUCT - GENERAL
             CALL BABT    (A,B,Z,NRB,NCB,KA,KBZ)
                MAX NCB = 500 (SIZE OF A)
             CALL BABTA   (AZ,B,NRB,NCB,KAZ,KB)
                MAX NCB = 500 (SIZE OF A)
             CALL BTAB    (A,B,Z,NRB,NCB,KAB,KZ)
                MAX NRB = 500 (SIZE OF A)
             CALL BTABA   (AZ,B,NRB,NCB,KAZ,KB)
                MAX NRB,NCB = 500 (SIZE OF MATRICES A,Z)

.06.04B   TRIPLE MATRIX PRODUCT - SPECIAL
             CALL BTAB1   (A,B,Z,NRB,NCB,KA,KB,KZ)
                MAX NRB = 500 (SIZE OF MATRIX A)
             CALL BTABA2  (AZ,B,N,KA)
                MAX N = 500 (SIZE OF MATRICES A,B,Z)
             CALL BTABC1  (A,B,C,Z,NRB,NCB,KA,KB,KC,KZ)
                MAX NRB = 500
             CALL BTDB1   (D,B,Z,NRB,NCB,KRB,KZ)
                MAX NRB = 500
             CALL BTDBC1  (D,B,C,Z,NRB,NCB,KB,KC,KZ)
             CALL UTAU1   (A,U,Z,N,KA,KU,KZ)
                MAX N = 500
             CALL UTAUC1  (A,U,C,Z,N,KA,KU,KC,KZ)
                MAX N = 500
.06.05    INVERSION
             CALL INV1    (A,Z,N,KAZ)
                MAX N = 250
             CALL INV2    (A,Z,N,KAZ)
                MAX N = 250
             CALL INV3    (A,Z,N,KAZ)
                USES DCOM1,INV4.
                MAX N = 250
             CALL INV4    (A,Z,N,KAZ)

.06.06    SIMULTANEOUS EQUATIONS
             CALL SMEQ1   (*A,*BVEC,ZVEC,N,KA)
               *A,BVEC ARE DESTROYED.

.06.07    EIGENVALUE, EIGENVECTOR
             CALL EIGN1   (*A,ZVAL,ZVEC,N,ECD,KAZ)
               *A IS DESTROYED.
             CALL EIGN1A  (*A,ZVAL,ZVEC,NIN,CTV,KAZ)
               *A IS DESTROYED.
```

```
.06     ALGEBRA (CONTINUED)

.06.08     DECOMPOSITION
              CALL DCOM1   (A,Z,N,KAZ)

.06.09     ROW, COLUMN OPERATIONS
              CALL ROWMLT (AVEC,B,Z,NR,NC,KBZ)
              CALL COLMLT (AVEC,B,Z,NR,NC,KBZ)

.06.10     VECTOR OPERATIONS
              CALL VDOT    (VA,VB,PRODCT,VAMAG,VBMAG,COSAB)
              CALL VCROSS (VA,VB,VZ,VAMAG,VBMAG,VZMAG,SINAB)
```

```
.07      MODIFICATION

.07.01      ASSEMBLY
               CALL ASSEM   (A,IZ,JZ,*Z,NRA,NCA,NRZ,NCZ,KA,KZ)
                 *BE SURE Z IS DEFINED. (EG CALL ZERO TO CLEAR Z).

.07.02      DISASSEMBLY
               CALL DISA    (A,IA,JA,Z,NRA,NCA,NRZ,NCZ,KA,KZ)

.07.03      REVISION
               CALL REVADD (ALPHA,A,IVEC,JVEC,*Z,NRA,NCA,NRZ,NCZ,KA,KZ)
                 *BE SURE Z IS DEFINED. (EG CALL ZERO TO CLEAR Z).
               CALL REVIJ   (AZ,IVEC,JVEC,NRA,NCA,NRZ,NCZ,KRAZ)

.07.04      TRANSPOSE
               CALL AT1     (A,Z,NRA,NCA,KA,KZ)
                  MAX NCA = 500
               CALL TRANS   (A,Z,NRA,NCA,KA,KZ)

.07.05      SYMMETRIZE
               CALL SYMLH   (AZ,N,K)
               CALL SYMUH   (AZ,N,K)

.07.06      NULLIFY
               CALL ZEROLH (AZ,N,K)
               CALL ZEROUH (AZ,N,K)

.07.07      DIAGONALIZE
               CALL DIAG    (AVEC,Z,N,KZ)
```

```
.08     INTERPOLATION, DIFFERENTIATION

.08.01     INTERPOLATION
              CALL TERP1  (XA,XZ,A,Z,NXA,NXZ,NCA,KA,KZ)
              CALL TERP2  (XA,XZ,A,Z,NXA,NXZ,NCA,KA,KZ)

.08.02     DIFFERENTIATION
              CALL DIFF1  (XA,XZ,A,Z,NXA,NXZ,NCA,KA,KZ)
              CALL DIFF2  (XA,XZ,A,Z,NXA,NXZ,NCA,KA,KZ)
```

```
.09     AIRLOAD

.09.01     LATERAL
              CALL ALOD1   (PP,DIST,CONC,CONVRT,ZVEC,NPP,ND,NC,KD,KC)

.09.02     AXIAL
              CALL ALOD2   (PP,DIST,CONC,CONVRT,ZVEC,NPP,ND,NC,KD,KC)
```

```
.10     MASS

.10.01     ROD
              CALL MASS1   (PP,DMASS,DRIN,CONC,CONVRT,Z,NPP,NDM,NDI,NC,
                            KDM,KDI,KC,KZ)

.10.02     BEAM
              CALL MASS2   (PP,DMASS,DRIN,CONC,CONVRT,Z,NPP,NDM,NDI,NC,
                            *NZ,KDM,KDI,KC,KZ)
                *NZ=2NPP WILL BE DEFINED BY MASS2. USE SYMBOL.

.10.03     FLUID
              CALL MASS2A  (PP,DMASS,EQSM,FLEVEL,CONVRT,Z,NPP,NDM,*NZ,
                            KDM,KZ)
                *NZ=2NPP+1 WILL BE DEFINED BY MASS2A. USE SYMBOL.
```

```
.11      STIFFNESS

.11.01      ROD
               CALL STIF1   (PP,DAE,Z,NPP,NDAE,KDAE,KZ)

.11.02      BEAM
               CALL STIF2   (PP,DKAG,DEI,Z,NPP,NDKAG,NDEI,*NZ,KDKAG,
                             KDEI,KZ)
                 *NZ=2NPP WILL BE DEFINED BY STIF2. USE SYMBOL.

.11.03      REDUCTION
               CALL SRED1   (A,R,T,N,NR,IFT,KART)
               CALL SRED2   (A,R,T,N,NR,IFT,KART)
               CALL SRED3   (A,IV,R,T,N,NR,IFT,KART)
```

```
.12      MODAL

.12.01      JACOBI
              CALL MODE1   (*AMASS,**STIF,W2,W,FREQ,N,FOD,KAS,NUTAPE)
                *AMASS IS REPLACED BY MODE SHAPES.
               **STIF IS DESTROYED.
                 USES BTABA,DCOM1,EIGN1,INV4.
                 MAX N = 500
              CALL MODE1A  (*AMASS,**STIF,W2,W,FREQ,N,FOD,KAS,NUTAPE)
                *AMASS IS REPLACED BY MODE SHAPES.
               **STIF IS DESTROYED.
                 USES BTABA,DCOM1,EIGN1,INV4.
                 MAX N = 500
              CALL MODE1B  (*AMASS,**FLEX,W2,W,FREQ,N,FOD,KAF,NUTAPE)
                *AMASS IS REPLACED BY MODE SHAPES.
               **FLEX IS DESTROYED.
                 USES BTABA,DCOM1,EIGN1,INV4,MULTA.
                 MAX N = 500
```

```
.13     RIGID BODY MODES

.13.01     CALCULATION
              CALL RBTG1   (XYZ,XYZREF,JDOF,JVEC,Z,NNODE,*NRZ,*NCZ,KXJ,
                            KZ)
                *NRZ,NCZ WILL BE DEFINED BY RBTG1. USE SYMBOL.
                 USES REVADD.
              CALL RBTG2   (XRT,XYZREF,JDOF,JVEC,Z,NNODE,*NRZ,*NCZ,KXJ,
                            KZ)
                *NRZ,NCZ WILL BE DEFINED BY RBTG2. USE SYMBOL.
                 USES REVADD.

.13.02     ORTHO-NORMALIZATION
              CALL ONRBM   (RBMODE,AMASS,N,NRBMOD,KRA)
                 USES BTAB,EIGN1,MULTA.
                 MAX N      = 250
                 MAX NRBMOD =   6
```

.14 INERTIAL TRANSFORMATION

```
        CALL UMAM1   (AMASS,RBMODE,Z,N,NRBMOD,KARZ)
           USES PART,BTAB,INV1,MULTB
           MAX N       = 250
           MAX NRBMOD =    6
```

```
.15     INTERNAL LOADS

.15.01      BEAM - SHEAR, MOMENT LOADS
               CALL VM1     (XVEC,DIST,CONC,AMP1,AMP2,CONVRT,ZVECV,ZVECM,
                             NX,ND,NC,NA1,NA2,KD,KC,KA1,KA2)

.15.02      BEAM - SHEAR, MOMENT TRANSFORMATION
               CALL VMTR1   (PP,Z,NPP,*NZ,KZ)
                 *NZ=2NPP WILL BE DEFINED BY VMTR1. USE SYMBOL.
```

```
.16     TIME RESPONSE

.16.01      FORCE IS OBTAINED BY LINEAR INTERPOLATION USING TABT,TABF
               CALL TRSP1   (*A,*B,*C,*D,TABT,TABF,XDO,XO,STARTT,DELTAT,
                             ENDT,NWRITE,NX,NRTAB,NCTAB,6HXNAME ,KARCD,
                             KTAB,NXTAPE,NUT1)
                  *MATRICES A,B,C,D ARE DESTROYED.
                   USES INV1,MULTB.
                   MAX NX     = 250
                   MAX NRTAB = 500
               CALL TRSP1B  (B,C,D,TABT,TABF,XDO,XO,STARTT,DELTAT,ENDT,
                             NWRITE,NX,NRTAB,NCTAB,6HXNAME ,KBCD,KTAB,NXTAPE)
                   MAX NX     = 250
                   MAX NRTAB = 500
               CALL TRSP2   (*A,*B,*C,*D,TABT,TABF,XDO,XO,STARTT,DELTAT,
                             ENDT,BETA,NWRITE,NX,NRTAB,NCTAB,6HXNAME ,KARCD,
                             KTAB, NXTAPE,NUT1)
                  *MATRICES A,B,C,D ARE DESTROYED.
                   USES INV1,MULT,MULTB.
                   MAX NX     = 250
                   MAX NRTAB = 250
               CALL TRSP3   (*AVEC,*BVEC,*CVEC,*D,TABT,TABF,*XDO,*XO,
                             STARTT,DELTAT,ENDT,NWRITE,NX,NRTAB,
                             NCTAB,6HXNAME ,KD,KTAB,NXTAPE)
                  *VECTORS AVEC,BVEC,CVEC,XDO,XO, MATRIX D ARE DESTROYED.
                   MAX NX     = 250
                   MAX NRTAB = 500
                   MAXIMUM UNIQUE TIMES PAST STARTT IN TABT = 250.

.16.02      FORCE IS CALCULATED AS (1 - COS)/2
               CALL TRSP1A  (*A,*B,*C,*D,FMAG,FPP,VEL,GL,XDO,XO,STARTT,
                             DELTAT,ENDT,NWRITE,NX,NF,6HXNAME ,KARCD,
                             NXTAPE,NUT1)
                  *MATRICES A,B,C,D ARE DESTROYED.
                   USES INV1,MULTB.
                   MAX NX = 250
                   MAX NF = 500
               CALL TRSP1C  (B,C,D,FMAG,FPP,VEL,GL,XDO,XO,STARTT,DELTAT,
                             ENDT,NWRITE,NX,NF,6HXNAME ,KBCD,NXTAPE)
                   MAX NX = 250
                   MAX NF = 500
               CALL TRSP2A  (*A,*B,*C,*D,FMAG,FPP,VEL,GL,XDO,XO,STARTT,
                             DELTAT,ENDT,BETA,NWRITE,NX,NF,6HXNAME ,KARCD,
                             NXTAPE,NUT1)
                  *MATRICES A,B,C,D ARE DESTROYED.
                   USES INV1,MULT,MULTB.
                   MAX NX = 250
                   MAX NF = 250
```

```
.16      TIME RESPONSE (CONTINUED)

.16.03      ADDITIONAL EQUATIONS
               CALL TRAE2  (6HXRUNNO,6HXNAME ,IFA,A,IFB,B,IFC,C,IFD,D,
                            IFE,E,ZTMM,STARTT,ENDT,MLTXTP,NWRITE,*ZIDENT,
                            **STA,6HZNAME ,NZ,KZ,NXTAPE,NZTAPE,STOREZ)
                  *Z HEADING (12A6 FORMAT).
                 **STATIONS (A6 FORMAT).
                   MAX NX = 250

.16.04      TIME RESPONSE MAX-MINS
               CALL TRMM    (6HXRUNNO,6HXNAME ,XTMM,STARTT,ENDT,*NX,
                            KX,NXTAPE)
                  *NX IS DEFINED IN SUBROUTINE, USE SYMBOL.
                   MAX NX = 250

.16.05      POWER SPECTRAL DENSITY OF ADDITIONAL EQUATIONS
               CALL TRPSD  (6HXRUNNO,6HXNAME ,IRAE,IEXP,STARTT,MLTXTP,
                            ZPSD,NFREQ,TIMPER,NXTAPE,WRKV)
```

.17 FREQUENCY RESPONSE

```
.17.01     RESPONSE SOLUTION
              CALL FR1     (*A,*B,*C,*D,TAPW,TABF,OMEGA,NX,NOMEGA,
                            NRTAB,NCTAB,KABCD,KTAB,WRKM,NTAPE)
                 *MATRICES A,B,C,D ARE DESTROYED.
                  B MUST BE NON-SINGULAR.
                  USES INV1,MULT,MULTF.
                  MAX NX     = 50
                  MAX NRTAB = 80

.17.02     ADDITIONAL EQUATIONS
              CALL FRAE1 (A,*STA,**ZIDENT,NZ,NX,NOMEGA,KA,NXTAPE,
                            NZTAPE)
                 *STATIONS (A6 FORMAT).
                **Z HEADING (12A6 FORMAT).
                  MAX NZ = 80
                  MAX NX = 50
```

APPENDIX B. SUMMARY OF CALLING INSTRUCTIONS - SPARSE FORMA SUBROUTINES.

*** LIST OF SYMBOLS ***

A	= INPUT MATRIX
ALPHA	= INPUT SCALAR
BETA	= INPUT SCALAR
NUTA	= LOGICAL NUMBER OF UTILITY TAPE CONTAINING INPUT MATRIX A
NUTB	= LOGICAL NUMBER OF UTILITY TAPE CONTAINING INPUT MATRIX B
NUTZ	= LOGICAL NUMBER OF UTILITY TAPE CONTAINING OUTPUT MATRIX Z
AVEC	= INPUT VECTOR (ROW OR COLUMN MATRIX)
IVEC	= INPUT VECTOR (ROW OR COLUMN MATRIX), INTEGER
Z	= RESULT MATRIX
ZVEC	= RESULT VECTOR
W	= MATRIX WORK SPACE
NR	= NUMBER OF ROWS
NC	= NUMBER OF COLUMNS
KR	= ROW DIMENSION
KC	= COLUMN DIMENSION
V	= VECTOR WORK SPACE
LV	= VECTOR WORK SPACE, INTEGER
KV	= V,LV DIMENSION
NUTI	= LOGICAL NUMBER OF ITH UTILITY TAPE
NTAPE	= LOGICAL NUMBER OF FORMA RESERVE TAPE

```
.01      MISCELLANEOUS

.01.01      CONVERSION
               CALL YSTOD   (NUTA,Z,*NR,*NC,KR,KC,V,LV,KV,NUT1)
                 *NR,NC WILL BE DEFINED BY YSTOD. USE SYMBOLS.
                  USES YIN,YIN1,ZZBOMB.

.01.02      ORDER MATRIX NON-ZERO ELEMENT LOCATIONS ROWWISE
               CALL YLORD   (NUTA,V,LV,KV,NUT1,NUT2)
                  USES YIN,YIN1,YOUT,YOUT1,YPART,ZZBOMB.

.01.03      ELIMINATE ZERO ELEMENTS
               CALL YNOZER  (NUTA,V,LV,KV,NUT1)
                  USES YIN,YIN1,YOUT,YOUT1,YPART,ZZBOMB.

.01.04      REPARTITION
               CALL YPART   (NUTA,V,LV,KV,NUT1)
                  USES YIN,YIN1,YOUT,YOUT1,ZZBOMB.

.01.05      COPY TO ANOTHER TAPE
               CALL EQUAL   (NUTA,NUTZ,V,LV,KV)
```

```
.02     INPUT

.02.01     REAL NUMBERS
              CALL YREAD   (NUTZ,V,LV,KV,NUT1)
                 USES INTAPE,LTAPE,PAGEHD,YIN,YIN1,YOUT,YOUT1,YPART,
                      YRTAPE,YWRITE,YWTAPE,ZZBOMB.
                           CARD INPUT (SAME AS READ EXCEPT SHAPE)
                 FIRST CARD    - ZNAME,NR,NC WITH A6,I4,I5 FORMAT.
                                 MATRIX SHAPE IN COLUMNS 16-21.
                                 REMARKS IN COLUMNS 22-69.
                                 $ IN COL 72 FOR WRITE TAPE INITIALIZE.
                                 BLANK,REWIND,LIST,OR TAPEID (FOR WRITE
                                 TAPE) IN COLUMNS 73-78.
                                 NWTAPE IN COLUMNS 79-80.
                 MIDDLE CARDS  - DATA WITH 2I5,4E17 FORMAT.
                 LAST CARD     - TEN ZEROES IN COLUMNS 1-10.
                                          TAPE INPUT (SAME AS READ)
                 ONE CARD      - ZNAME,0 OR -LOCATION,NRTAPE(WITH - FOR
                                 NO WRITE OUT),ZRUNNO WITH
                                 A6,I5,A6 FORMAT
                                 BLANK,REWIND,OR LIST (FOR READ TAPE)
                                 IN COLUMNS 22-27.
                                 REMARKS IN COLUMNS 28-69.
                                 $ IN COL 72 FOR WRITE TAPE INITIALIZE.
                                 BLANK,REWIND,LIST,OR TAPEID (FOR WRITE
                                 TAPE) IN COLUMNS 73-78.
                                 NWTAPE IN COLUMNS 79-80.
```

```
.03      OUTPUT

.03.01      PRINT REAL NUMBERS
               CALL YWRITE (NUTA,6H ANAME,V,LV,KV)
                  USES PAGEHD,YIN,YIN1,ZZBOMB.

.03.02      PUNCH REAL NUMBERS
               CALL YPUNCH (NUTA,6H ANAME,V,LV,KV)
                  USES YIN,YIN1,ZZBOMB.
```

```
.04     TAPE OPERATIONS

         REWIND NRTAPE,NWTAPE AT BEGINNING OF MAIN PROGRAM.
         SEE APPENDIX A FOR TAPE INITIALIZING, LISTING, AND UPDATING.

.04.01     READING
              CALL YRTAPE (6HZRUNNO,6H ZNAME,NUTZ,V,LV,KV,*NRTAPE,NUTI)
                *NRTAPE MUST BE A READ,WRITE TAPE ONLY.
                 USES LTAPE,YIN,YINI,YOUT,YOUTI,ZZBOMB.
              CALL YIN     (NUTA,ZVEC,LS,LE)
              CALL YINI    (NUT,IA,NS,NE)

.04.02     WRITING
              CALL YWTAPE (NUTA,6H ANAME,V,LV,KV,*NWTAPE)
                *NWTAPE MUST BE A READ,WRITE TAPE ONLY.
                 USES YIN,YINI,ZZBOMB.
              CALL YOUT    (NUTA,AVEC,LS,LE)
              CALL YOUTI   (NUT,IA,NS,NE)
```

```
.05    GENERATION

.05.01     BASIC
              CALL YUNITY  (NUTZ,NR,V,LV,KV)
                 USES YOUT,YOUTI.
              CALL YZERO   (NUTZ,NR,NC)
                 USES YOUT,YOUTI.

.05.02     RAYLEIGH VECTORS
              CALL YRVI    (NUTZ,N,NU,V,LV,KV,NUT1,NUT2,NUT3)
                 USES YIN,YINI,YLORD,YOUT,YOUTI,YPART,YTRANS,ZZBOMP.
```

```
.06     ALGEBRA

.06.01      SCALAR PRODUCT
               CALL YAA     (ALPHA,NUTA,NUTZ,V,LV,KV,NUT1,NUT2)
                  USES YIN,YINI,YLORD,YNOZER,YOUT,YOUTI,YPART,ZZBCMB.

.06.02      ADDITION, SUBTRACTION
               CALL YAABB   (ALPHA,NUTA,BETA,NUTB,NUTZ,V,LV,KV,NUT1,
                             NUT2)
                  USES XLORD,YIN,YINI,YLORD,YNOZER,YOUT,YOUTI,YPART,
                       YSYMLH,YSYMUH,ZZBCMB.

.06.03      MULTIPLICATION - BASIC
               CALL YMULT   (NUTA,NUTB,NUTZ,V,LV,KV,NUT1)
                  USES YIN,YINI,YLORD,YNOZER,YOUT,YOUTI,YPART,YSYMLH,
                       YSYMUH,ZZBCMB.

.06.04      MULTIPLICATION - SPECIAL
               CALL YMULT1  (NUTA,NUTB,NUTZ,V,LV,KV,NUT1)
                  USES YIN,YINI,YLORD,YNOZER,YOUT,YOUTI,YPART,YSYMLH,
                       YSYMUH,ZZBCMB.
               CALL YMULT2  (NUTA,NUTB,NUTZ,W1,W2,V,LV,KV,KW,NUT1)
                  USES YDTOS,YIN,YINI,YLORD,YNOZER,YOUT,YOUTI,YPART,
                       YSYMLH,YSYMUH,ZZBCMB.
               CALL YMULT4  (NUTA,B,NUTZ,W,V,LV,KV,KW,NUT1)
                  USES YIN,YINI,YLORD,YOUT,YOUTI,YPART,ZZBCMB.

.06.05      TRIPLE MATRIX PRODUCT
               CALL YBTAB   (NUTA,NUTB,NUTZ,V,LV,KV,NUT1,NUT2)
                  USES YIN,YINI,YLORD,YMULT,YNOZER,YOUT,YOUTI,YPART,
                       YSYMLH,YSYMUH,ZZBCMB.

.06.06      DECOMPOSITION
               CALL YDCM3A  (NUTA,NUTU,NUTD,V,LV,KV,NUT1,NUT2)
                  USES YIN,YINI,YLORD,YOUT,YOUTI,YPART,YTRANS,ZZBCMB.
               CALL YDCOM3  (NUTA,NUTU,NUTD,V,LV,KV,NUT1,NUT2)
                  USES YIN,YINI,YLORD,YOUT,YOUTI,YPART,YTRANS,ZZBCMB.

.06.07      BACK SOLUTION
               CALL YBSL3A  (NUTU,NUTD,NUTB,NUTZ,V,LV,KV,NUT1,NUT2)
                  USES YIN,YINI,YLORD,YOUT,YOUTI,YPART,YTRANS.
```

```
.07     MODIFICATION

.07.01     ASSEMBLY
              CALL YASSEM (NUTA,IRZ,JCZ,*NUTZ,V,LV,KV,NUT1,NUT2,NUT3)
                 *BE SURE Z IS DEFINED. (CALL YZERO TO CLEAR NUTZ).
                  USES YIN,YINI,YLORD,YOUT,YOUTI,YPART,YSYMLH,YSYMUH,
                        ZZBOMB.

.07.02     DISASSEMBLY
              CALL YDISA  (NUTA,IRA,JCA,NUTZ,NRZ,NCZ,V,LV,KV,NUT1)
                  USES YIN,YINI,YLORD,YOUT,YOUTI,YPART,ZZBOMB.

.07.03     REVISION
              CALL YREVAD (ALPHA,NUTA,IVEC,JVEC,*NUTZ,V,LV,KV,NUT1,
                             NUT2,NUT3,NUT4)
                 *BE SURE Z IS DEFINED. (CALL YZERO TO CLEAR NUTZ).
                  USES XLORD,YAABB,YIN,YINI,YLORD,YNOZER,YOUT,YOUTI,
                        YPART,YSYMLH,YSYMUH,ZZBOMB.
              CALL YREVAD1 (ALPHA,A,IJVEC,*NUTZ,NRA,V,LV,KV,KA,NUT1,
                             NUT2,NUT3,NUT4)
                 *BE SURE Z IS DEFINED. (CALL YZERO TO CLEAR NUTZ).
                  USES XLORD,YAABB,YIN,YINI,YLORD,YNOZER,YOUT,YOUTI,
                        YPART,YSYMLH,YSYMUH,ZZBOMB.
              CALL YREVAD2 (NUTA,NUTZ,NRZ,W,KW,V,LV,KV,NUT1,NUT2,NUT3)
                  USES YIN,YINI,YOUT,YOUTI,YPART,ZZBOMB.
              CALL YREVAD3 (NUTA,NUTZ,NRZ,NCZ,W,KW,V,LV,KV,NUT1,NUT2,NUT3)
                  USES YIN,YINI,YOUT,YOUTI,YPART,ZZBOMB.
              CALL YRVIS1 (A,JVEC,NUTZ,NRAZ,NCA,NCZ,V,LV,KV,KA)
                  USES XLORD,YOUT,YOUTI,ZZBOMB.
              CALL YRVTOD (NUTA,IVEC,JVEC,Z,NRZ,NCZ,V,LV,KV,KZ)
                  USES YIN,YINI,ZZBOMB.

.07.04     TRANSPOSE
              CALL YTRANS (NUTA,NUTZ,V,LV,KV,NUT1,NUT2)
                  USES YIN,YINI,YLORD,YOUT,YOUTI,YPART,ZZBOMB.

.07.05     SYMMETRIZE
              CALL YSYMLH (NUTAZ,V,LV,KV,NUT1,NUT2)
                  USES YIN,YINI,YLORD,YNOZER,YOUT,YOUTI,YPART.
              CALL YSYMUH (NUTAZ,V,LV,KV,NUT1,NUT2)
                  USES YIN,YINI,YLORD,YNOZER,YOUT,YOUTI,YPART.

.07.06     NULLIFY
              CALL YZERLH (NUTAZ,V,LV,KV,NUT1,NUT2)
                  USES YIN,YINI,YLORD,YOUT,YOUTI,YPART,ZZBOMB.
              CALL YZERUH (NUTAZ,V,LV,KV,NUT1,NUT2)
                  USES YIN,YINI,YLORD,YOUT,YOUTI,YPART,ZZBOMB.

.07.07     DIAGONALIZE
              CALL YDIAG  (NUTA,NUTZ,V,LV,KV)
                  USES YIN,YINI,YOUT,YOUTI,ZZBOMB.
```

```
.08     STIFFNESS

.08.01     REDUCTION
              CALL YSRED2 (NUTA,NUBR,NUTT,NR,IFT,V,LV,KV,NUT1,NUT2,
                           NUT3,NUT4)
                 USES YASSEM,YDISA,YIN,YINI,YLORD,YNOZER,YOUT,YOUT1,
                      YPART,YSYMLH,YSYMUH,YTRANS,YUNITY,YZERO,ZZBOMB.
```

.09   MODAL

.09.01   ITERATIVE RAYLEIGH-RITZ

```
          CALL YMOD2A (NUTM,NUTK,NUTZ,W2,W,FREQ,NW,NU,SHIFT,TOLZ,
                       TOLW2,MAXIT,IFPRNT,V,LV,A,S,KVIN,KA,
                       NUT1,NUT2,NUT3,NUT4,NUT5,NUT6)
             USES BTABA2,EIGN1A,INV4,MODEIX,NAME,PAGEHD,TIMCHK,
             WRITF,WRITIM,XLORD,YAABB,YBSL3A,YDCM3A,YDTOS,YIN,
             YIN1,YLORD,YMULT1,YMULT2,YMULT4,YNOZER,YOUT,YOUT1,
             YPART,YRV1,YSTOD,YSYMLH,YSYMUH,YTRANS,YWRITE,ZZBOMB.
```

APPENDIX C. SUMMARY OF CALLING INSTRUCTIONS - FINITE ELEMENT ROUTINES.

A 6H ARGUMENT MAY ALSO BE A VARIABLE READ WITH AN A6 FORMAT OR OBTAINED WITH A DATA STATEMENT.

*** LIST OF SYMBOLS ***

NUTB	= LOGICAL NUMBER OF UTILITY TAPE ON WHICH ELEMENT UNIT LOAD BUCKLING MATRICES AND IVECS ARE OUTPUT. NUTB MAY BE ZERO IF BUCKLING MATRICES ARE NOT FORMED. USES FORTRAN READ, WRITE.
NUTEL	= LOGICAL NUMBER OF TAPE CONTAINING ELEMENT INPUT DATA FOR THIS SUBROUTINE AND SUBROUTINES AXIAL, ETC GIVEN BY NAMEL. IF NUTEL = 5, DATA WILL BE READ FROM CARDS.
NUTK	= LOGICAL NUMBER OF UTILITY TAPE ON WHICH ASSEMBLED STIFFNESS MATRIX IS OUTPUT IN SPARSE NOTATION. NUTK MAY BE ZERO IF STIFFNESS MATRIX IS NOT FORMED. USES FORMA YIN, YOUT.
NUTKX	= LOGICAL NUMBER OF UTILITY TAPE ON WHICH ELEMENT STIFFNESS MATRICES (SAME AS GLOBAL LOADS TRANSFORMATION MATRICES) AND IVECS ARE STORED. NUTKX MAY BE ZERO IF STIFFNESS MATRIX IS NOT FORMED. USES FORTRAN READ, WRITE.
NUTLT	= LOGICAL NUMBER OF UTILITY TAPE ON WHICH ELEMENT LOCAL LOAD TRANSFORMATION MATRICES AND IVECS ARE OUTPUT. NUTLT MAY BE ZERO IF LOAD TRANSFORMATIONS ARE NOT FORMED. USES FORTRAN READ, WRITE.
NUTM	= LOGICAL NUMBER OF UTILITY TAPE ON WHICH ASSEMBLED MASS MATRIX IS OUTPUT IN SPARSE NOTATION. NUTM MAY BE ZERO IF MASS MATRIX IS NOT FORMED. USES FORMA YIN, YOUT.
NUTMX	= LOGICAL NUMBER OF UTILITY TAPE ON WHICH ELEMENT MASS MATRICES AND IVECS ARE STORED. NUTMX MAY BE ZERO IF MASS MATRIX IS NOT FORMED. USES FORTRAN READ, WRITE.
NUTST	= LOGICAL NUMBER OF UTILITY TAPE ON WHICH ELEMENT STRESS TRANSFORMATION MATRICES AND IVECS ARE OUTPUT. NUTST MAY BE ZERO IF STRESS TRANSFORMATIONS ARE NOT FORMED. USES FORTRAN READ, WRITE.

```
.01     MISCELLANEOUS

.01.01     DIRECTION COSINES
              CALL DCOS1A (CJ,EJ,Z,KCJ,KEJ,KZ)
                 USES EULER,MULTB,ZZBOMB.
              CALL DCOS1B (CJ,EJ,Z,KCJ,KEJ,KZ)
                 USES EULER,MULTB,ZZBOMB.
              CALL DCOS2  (CJ,EJ,Z,KCJ,KEJ,KZ)
                 USES EULER,MULTB,ZZBOMB.
              CALL DCOS3C (CJ,EJ,Z,KCJ,KEJ,KZ)
                 USES EULER,MULTB,ZZBOMB.

.01.02     EULER ANGLES
              CALL EULER  (E,R,KR)

.01.03     TETRAHEDRON GEOMETRY
              CALL TEGEOM (CJ,JM,VL,DV,KCJ,IFBAD)
                 USES VCROSS,VDOT.
```

```
.02    FINITE ELEMENT MASS, STIFFNESS, BUCKLING, LOAD TRANSFORMATION,
                     STRESS TRANSFORMATION

          CALL FINEL   (XYZ,JDOF,EUL,NUTEL,NJ,NUTM,NUTK,NUTLT,
                        NUTST,NUTB,V,LV,KV,KRX,KRJ,KRE,NUTMX,NUTKX,
                        NUT1,NUT2,NUT3)
             USES AXIAL,BAR,FLUID,GRAVTY,PAGEHD,QUAD,RECTSP,TRNGL,
                  REVAD2,ZZBOMB.

          CALL AXIAL   (XYZ,JDOF,EUL,NUTEL,NJ,NUTMX,NUTKX,NUTLT,
                        NUTST,W,T,S,KX,KJ,KE,KW)
             USES ATXBA1,ATXBB,DCOS1A,EULER,K1A1,M1A1,M1A2,MAS1A,
                  MULTA,MULTB,PAGEHD,STF1A,ZZBOMB.

          CALL BAR     (XYZ,JDOF,EUL,NUTEL,NJ,NUTMX,NUTKX,NUTBX,
                        NUTLT,NUTST,W,T,S,KX,KJ,KE,KW)
             USES ATXBA1,B1A1,B1A2,BTABA,BUC1B,DCOS1B,EULER,K1A1,
                  K1B1,K1C1,M1A1,M1A2,M1B1,M1B2,M1C1,M1C2,MAS1B,
                  MULTA,MULTB,PAGEHD,STF1B,ZZBOMB.

          CALL FLUID   (XYZ,JDOF,EUL,NUTEL,NJ,NUTMX,NUTKX,NUTLT,NUTST,
                        W,T,S,KX,KJ,KE,KW)
             USES TEGEOM,VCROSS,VDOT,ZZBOMB.

          CALL GRAVTY  (XYZ,JDOF,EUL,NUTEL,NJ,NUTKX,W,T,S,KX,KJ,KE,KW)
             USES KGRAV,MULTA,MULTB,VCROSS,ZZBOMB.

          CALL QUAD    (XYZ,JDOF,EUL,NUTEL,NJ,NUTMX,NUTKX,NUTBX,
                        NUTLT,NUTST,W,T,S,KX,KJ,KE,KW)
             USES ATXBA1,BTABA,DCOS2,EULER,K2A1,K2B1,M2A1,
                  M2A2,M2B1,M2B2,MAS2,MAS3,MULTA,MULTB,PAGEHD,
                  REVADD,STF2,STF3,ZZBOMB.

          CALL RECTSP  (XYZ,JDOF,EUL,NUTEL,NJ,NUTMX,NUTKX,NUTLT,NUTST,
                        W,T,S,KX,KJ,KE,KW)
             USES MAS3A,PAGEHD,STF3A,ZZBOMB.

          CALL TRNGL   (XYZ,JDOF,EUL,NUTEL,NJ,NUTMX,NUTKX,NUTBX,
                        NUTLT,NUTST,W,T,S,KX,KJ,KE,KW)
             USES ATXBA1,BTABA,DCOS2,EULER,K2A1,K2B1,M2A1,
                  M2A2,M2B1,M2B2,MAS2,MULTA,MULTB,PAGEHD,STF2,ZZBOMB.
```

```
.03     MASS

.03.01     ROD
              CALL MAS1A    (CJ,EJ,A1,A2,RO,6HNAMEM ,KCJ,KEJ,KZ)
                 USES ATXBB,EULER,M1A1,M1A2,ZZBOMB.
              CALL M1A1     (A1,A2,RL,RO,Z,KZ)
              CALL M1A2     (A1,A2,RL,RO,Z,KZ)
              CALL M1C1     (PI1,PI2,RL,RO,Z,KZ)
              CALL M1C2     (PI1,PI2,RL,RO,Z,KZ)

.03.02     BEAM
              CALL MAS1B    (CJ,EJ,A1,A2,PI1,PI2,RO,6HNAMEM ,Z,W,KCJ,KEJ,
                             KZ,KW)
                 USES BTABA,DCOS1B,EULER,M1A1,M1A2,M1B1,M1B2,M1C1,
                      M1C2,MULTB,ZZBOMB.
              CALL M1B1     (A1,A2,RL,RO,Z,KZ)
              CALL M1B2     (A1,A2,RL,RO,Z,KZ)

.03.03     TRIANGLE
              CALL MAS2     (CJ,EJ,TMAS,RO,6HNAMEM ,Z,W1,W2,KCJ,KEJ,KZ,KW1,
                             KW2)
                 USES BTABA,DCOS2,EULER,M2A1,M2A2,M2B1,M2B2,MULTB,ZZBOMB.
              CALL M2A1     (X2,Y3,TH,RO,Z,KZ)
              CALL M2A2     (X2,X3,Y3,TH,RO,Z,T,R,KZ,KT,KR)
                 USES BTABA.
              CALL M2B1     (X2,Y3,TH,RO,Z,KZ)
              CALL M2B2     (X2,X3,Y3,TH,RO,Z,T,R,KZ,KT,KR)
                 USES BTABA.

.03.04     QUADRILATERAL
              CALL MAS3     (CJ,EJ,TMAS,RO,6HNAMEM ,Z,W1,W2,KCJ,KEJ,KZ,KW1,
                             KW2)
                 USES BTABA,DCOS2,EULER,M2A1,M2A2,M2B1,M2B2,MAS2,
                      MULTB,ZZBOMB.

.03.05     RECTANGULAR SHEAR PANEL
              CALL MAS3A    (CJ,EJ,TMAS,RO,6HNAMEM ,Z,W1,W2,KCJ,KEJ,
                             KZ,KW1,KW2)
                 USES BTABA,DCOS3C,M3C1,ZZBOMB.
              CALL M3C1     (X3,Y3,TH,RO,Z,KZ)
```

```
.04      STIFFNESS

.04.01      ROD
               CALL STF1A   (CJ,EJ,A1,A2,E,6HNAMEK  ,6HNAMEST,S,TL,TS,NRST,
                             KCJ,KEJ,KS,KTL,KTS)
                  USES ATXBA1,DCOS1A,EULER,K1A1,MULTA,MULTB,ZZBOMB.
               CALL K1A1    (A1,A2,RL,E,Z,TS,KZ,KTS)
               CALL K1C1    (TJ1,TJ2,R1,R2,RL,G,Z,TS,KZ,KTS)

.04.02      BEAM
               CALL STF1B   (CJ,EJ,KODE,A1,A2,TJ1,TJ2,BIZ1,BIZ2,BIY1,
                             BIY2,R1,R2,CY1,CY2,CZ1,CZ2,SF,E,G,6HNAMEK  ,
                             6HNAMEST,S,TL,TS,NRST,KCJ,KEJ,KS,KTL,KTS)
                  USES ATXBA1,DCOS1B,K1A1,K1B1,K1C1,MULTA,MULTB,ZZBOMB.
               CALL K1B1    (BI1,BI2,C1,C2,A1,A2,SF,RL,E,G,Z,TS,KZ,KTS)

.04.03      TRIANGLE
               CALL STF2    (CJ,EJ,TMEM,TBEN,E,ANU,6HNAMEK  ,6HNAMEST,S,TL,TS,
                             NRST,KCJ,KEJ,KS,KTL,KTS)
                  USES ATXBA1,DCOS2,EULER,K2A1,K2B1,MULTA,MULTB,ZZBOMB.
               CALL K2A1    (X2,X3,Y3,TH,E,ANU,Z,T,R,KZ,KT,KR)
                  USES BTABA,MULTA,ZZBOMB.
               CALL K2B1    (X2,X3,Y3,TH,E,ANU,Z,TS,T,KZ,KTS,KT)
                  USES BTABA,MULTA,ZZBOMB.

.04.04      QUADRILATERAL
               CALL STF3    (CJ,EJ,TMEM,TBEN,E,ANU,6HNAMEK  ,6HNAMEST,S,TL,TS,
                             NRST,KCJ,KEJ,KS,KTL,KTS)
                  USES ATXBA1,DCOS2,EULER,K2A1,K2B1,MULTA,MULTB,REVADD,
                        STF2,ZZBOMB.

.04.05      RECTANGULAR SHEAR PANEL
               CALL STF3A   (CJ,EJ,TH,G,6HNAMEK  ,6HNAMEST,S,TL,TS,NRST,KCJ,
                             KEJ,KS,KTL,KTS)
                  USES ATXBA1,DCOS3C,K3C1,MULTA,ZZBOMB.
               CALL K3C1    (X3,Y3,TH,G,Z,T,KZ,KT)

.04.06      FLUID
               CALL KGRAV   (CJ,JM,FV,A,W,KW,KCJ)
                  USES MULTA,MULTB,VCROSS.
```

.05   BUCKLING

```
          CALL BUCIB   (CJ,EJ,KODEB,6HNAMEB  ,Z,W,KCJ,KEJ,KZ,KW)
             USES B1A1,B1A2,BTABA,DCOSIB,EULER,MULTB,ZZHOMB.
          CALL B1A1    (RL,Z,KZ)
          CALL B1A2    (RL,Z,KZ)
```

.06 FLUID PRESSURE

```
        CALL PRESS  (CJ,T,NJN,NCOL,KCJ,KW)
           USES REVADD,INVINP,MULTR.
```