Program RSVR2 User's Manual

Dam Safety Office

Report No. DSO-04-02
Department of the Interior
Bureau of Reclamation
February 2004

Contents

Introduction

RSVR2 is a modification of the RSVR program which originated from Kuo (J. S.-H. Kuo, Fluid-Structure Interactions: Added Mass Computations for Incompressible Fluid, Report No. UBC/EERC-82/09, Earthquake Engineering Research Center, University of California, Berkeley, California, August 1982). Brief descriptions are also included in the ADAP-88 user's manual (G. L. Fenves, S. Mojtahedi, and R. B. Reimer, ADAP-88 A Computer Program for Nonlinear Earthquake Analysis of Concrete Arch Dams, Report No. UBC/EERC-89/12, Earthquake Engineering Research Center, University of California, Berkeley, California, November 1989) and the EADAP user's manual (Y. Ghanaat and R. W. Clough, EADAP Enhanced Arch Dam Analysis Program User's Manual, Report No. UBC/EERC-89/07, Earthquake Engineering Research Center, University of California, Berkeley, California, November 1989). Kuo refers to the program as RSVOIR, and it is called RESVOR and INCRES in the ADAP-88 and EADAP user's manuals, respectively. The source code used for the present work is written in Fortran, contains a “program resvor” statement, and prints a RSVR title as well as a 1992 University of California copyright statement.

The versions of RSVR are currently used by many groups to calculate added mass matrices for including hydrodynamic effects in the earthquake response of dams. These versions, which are essentially the same, and RSVR2 compute an added mass matrix by employing a finite element mesh of the water. Among the improvements in the modified program are:

- Arbitrary location and orientation of the x, y, z axes
- Twenty-node water elements that can be mapped into triangular wedges, tetrahedra, etc., by arbitrarily superimposing nodes
- Automatic generation of 2-dimensional surface elements that define the portion of the reservoir boundary at the dam face
- Computation of the additional hydrodynamic forces on the dam that arise from accelerations of the floor and sides of the reservoir, which also includes automatic generation of the 2-dimensional surface elements on this boundary
- Output of a list of water mesh nodes which defines the locations and ordering of the added masses and forces

The added mass matrix is a square matrix [MA] of dimension ND*3; where ND is the number of nodes of the water mesh located on the upstream face of the dam that are at or below the water free surface. The factor of 3 accounts for the presence of x, y, and z translational degrees of freedom of the dam at each node. The hydrodynamic forces, denoted by {Fx}, {Fy}, and {Fz} in vector form, are also of dimension ND*3. {Fx} contains the forces acting on the dam arising from the pressures generated when the reservoir floor and sides accelerate uniformly with unit amplitude in the x direction. {Fy} and {Fz} are produced by accelerations in the y and z directions, respectively. Before being actually applied to the dam, these force vectors must be multiplied by the respective components of ground acceleration. The forces {Fx}, {Fy}, and {Fz} are applied to the right side of the equations of motion of the dam when the dam is analyzed.

To find [MA], the water is meshed with 3-dimensional finite elements and a "stiffness" matrix [K] is constructed by a finite element discretization of the Laplace equation for pressure in an incompressible fluid. There is one pressure degree of freedom per node, so the dimension of the square matrix [K] is the number of nodes in the 3-dimensional water mesh minus the number of nodes at the free surface (where the pressure is zero). The next step is to condense [K] down to the nodes on the dam face. The result is a square matrix [K'] of dimension NDBS, where NDBS equals ND minus the number of free-surface water mesh nodes on the dam face. Then, a rectangular NDBS by ND*3 matrix [B] is constructed which provides the physical connection between the mesh of the water and the mesh of the dam. When [B]T is multiplied by nodal pressures at the dam face, it gives a vector of corresponding x, y, z forces that act on the dam nodes. [B] is computed from the 2-dimensional mesh formed by the boundary of the 3-dimensional water element mesh at the dam face. Finally, [MA] is computed from

$$[M_A] = \frac{w}{g}[B]^T[K']^{-1}[B] \quad (1),$$

where w is the unit weight of water and g is the gravitational acceleration.

The existing versions of RSVR, as well as the modified program RSVR2, compute a diagonalized version of [MA] which is formed by scaling the diagonal terms of [MA]. Three scaling factors are used: cx, cy, and cz, which respectively, scale the terms corresponding to the x, y, and z translational degrees of freedom of the dam. The factor cx is found from

$$c_x = \frac{\{l_x\}^T[M_A]\{l_x\}}{\{l_x\}^T[\text{diagonal terms of } M_A]\{l_x\}} \quad (2),$$

where {lx} is a vector of length ND*3 containing 1s in locations corresponding to x translational degrees of freedom and 0s otherwise. Factors cy and cz are found with similar vectors, {ly} and {lz}, but contain ls in locations corresponding to y and z degrees of freedom. The formula for the cx scaling factor is based on having the x component of total hydrodynamic force acting on the dam because of a uniform acceleration of the dam in the x direction. The scaling factor is the same for both the original [MA] and its diagonalized version. Factors cy and cz have a similar basis, but they are associated with the y and z force components caused by y and z accelerations, respectively.

To compute the additional force vectors {Fx}, {Fy}, and {Fz}, uniform accelerations in the x, y, and z directions of the floor and sides of the 3-dimensional water mesh are converted into nodal accelerations {Qx}, {Qy}, and {Qz}, respectively. Then, in parallel to the condensation of [K] to [K'], these vectors are condensed down to the NDBS nodes on the dam face below the water surface; the condensed vectors are denoted by {Q'x}, {Q'y}, and {Q'z}. In terms of previously defined quantities, the force vectors are given by

$$[F_x, F_y, F_z] = -\frac{w}{g}[B]^T[K']^{-1}[Q'_x, Q'_y, Q'_z] \quad (3),$$

which is similar to the expression for [MA] except that the three-column [Q'] has replaced [B]. The negative sign is consistent with the forces {Fx}, {Fy}, and {Fz} being applied to the right side of the equation of motion of the dam.

With [B]T omitted from the above equation, it becomes

$$[p_x, p_y, p_z] = -\frac{w}{g}[K']^{-1}[Q'_x, Q'_y, Q'_z] \quad (4),$$

where {px}, {py}, and {pz} are the pressures at the NDBS nodes on the dam face below the water free surface caused by unit accelerations of the reservoir floor and sides in the x, y, and z directions, respectively. These pressures are computed by RSVR2 using a compression positive sign convention.

After computation of the added mass matrix [MA], or its diagonalized version, and the added loads {Fx}, {Fy}, and {Fz}, these quantities need to be assembled into the equations of motion of the dam, (i.e., input to a separate program that performs the dam analysis). For RSVR2 to be compatible with the other program, the water mesh must match the dam mesh at the interface between the dam and the water in terms of node location and element layout and the x, y, and z directions for the two meshes must coincide. The water mesh can cover all of the dam face or only a partial height. Assembly is to the x, y, and z translational degrees of freedom of the dam at the ND covered nodes on the upstream face. If the dam mesh does not employ nodes at its upstream face, such as with shell elements where the nodes are located at mid-thickness, there will be some error present. However, if the dam thickness is not too large, this error should be small.

The 3-dimensional water mesh does not employ a transmitting boundary or other boundary condition to represent an infinitely long reservoir. However, an infinite reservoir can be approximately modeled with a reservoir of finite length as long as the distance between the dam and the upstream end of the reservoir is at least one to two times the water depth.

Description of Input

This section defines the input for the water mesh. Entries are either text or real or integer numbers, as indicated. The data should be contained in a file named for005. Additional information is given in section 4.

I1: 80 character heading (text)

I2: Values of nnp, nel, w, g, watl, and iup; where

nnp = number of nodes in the water mesh (integer)

nel = number of 3-dimensional elements in the water mesh (integer)

w = weight density of water (real)

g = gravitational acceleration (real)

watl = elevation of the water free surface referenced to the x, y, z coordinate system (real)

iup = 1, 2, or 3 if the x, y, or z axis, respectively, is upward (integer)

I3: For each node—n, xcoor, ycoor, and zcoor; where

n = node number (integer)

xcoor = x coordinate (real) of node n

ycoor = y coordinate (real) of node n

zcoor = z coordinate (real) of node n

I4: For each sequence of boundary nodes to be assigned a type— n1, n2, inc, and itype; where

n1 = starting node number of a sequence (integer)

n2 = ending node number of a sequence (integer)

inc = node number increment for generated nodes (integer)

itype = –1 for nodes located on the dam face

= +1 for nodes located on the reservoir floor and sides.

If the sequence consists only of node n1, then n2 and inc should be input as zero. The final line of this section should contain four integer zeroes.

I5: For each 3-dimensional element -- n and (nce(i), i=1, 20); where

n = element number (integer)

nce(i) = node number (integer) corresponding to local node i of element n as shown in figure 1. Use zeroes for omitted mid-edge nodes. Repeat node numbers when nodes are superimposed. The connectivity array for element n is nce.

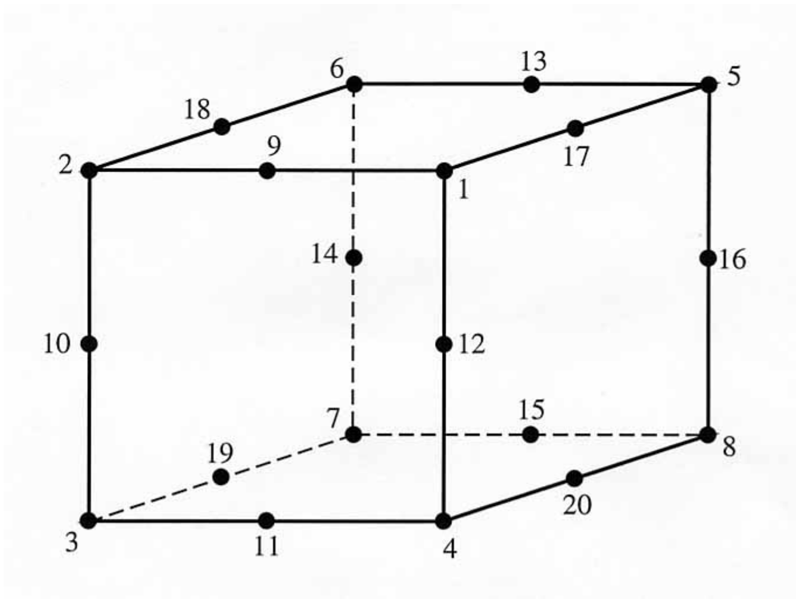

Figure 1.—Three-dimensional water element showing 20 nodes and ordering for connectivity array.

Description of Output

Output is written to three files that are named for001, for002, and for006. The first two are unformatted; file for006 is formatted. Unformatted records are either integer or double-precision real, as noted. Additional information is given in section 4.

File for001

first record:	Value of ND (integer).
second record:	List of the ND node numbers (integer) of the water mesh that are located on the dam face.
third record:	The ND*3 diagonalized added masses (d-p real). See equation 2 and accompanying text.
next three records:	The terms (d-p real) of the force vectors {Fx}, {Fy}, and {Fz}, one force vector of ND*3 terms per record. See equation 3.

File for002

first record:	Value of ND (integer).
second record:	List of the ND node numbers (integer) of the water mesh that are located on the dam face.
next ND*3 records:	The terms (d-p real) of each column of the added mass matrix [MA], one column of ND*3 terms per record. See equation 1.

next three records: The terms (d-p real) of the force vectors {Fx}, {Fy}, and {Fz}, one force vector of ND*3 terms per record. See equation 3.

File for006

O1: Heading line, as input

O2: Values of nnp, nel, w, g, watl, and iup; as input

O3: Boundary node type data n1, n2, inc, and itype for each sequence, as input

O4: x, y, and z coordinates (as input) and type (as input or set/reset by RSVR2) for each node of the water mesh

O5: 20-node connectivity array (as input) and volume (as computed by RSVR2) for each
3-dimensional water element

O6: 8-node connectivity array for each 2-dimensional surface element identified by RSVR2. The element is listed as "dam" when located at the dam face or as "rfs" when located along the reservoir floor and sides.

O7: Pressures at the NDBS nodes on the dam face below the elevation watl generated by 1g x, y, and z accelerations of the dam alone and of the dam and the reservoir floor and sides moving together. These cases are denoted by "dam" and "d+r", respectively. Pressure is positive in compression. See equation 4 (multiplied by 1g).

O8: The diagonalized added masses corresponding to the x, y, and z degrees of freedom of the dam at each of the ND nodes on the dam face (at and below the elevation watl). See equation 2 and the accompanying text.

O9: Terms of the force vector {Fx} corresponding to the x, y, and z degrees of freedom of the dam at each of the ND nodes on the dam face (at and below the elevation watl). See equation 3.

O10: Same as output O9 except {Fy} instead of {Fx}

O11: Same as output O9 except {Fz} instead of {Fx}

Additional Information

The x, y, z coordinate system is right-handed and can be placed in any orientation, except that one of the axes must be vertical with the positive direction upward. This axis is x, y, or z as indicated by iup = 1, 2, or 3, respectively. The location of the origin of the x, y, z system is arbitrary. Directions of the x, y, and z axes for the dam and water meshes must be the same. However, the respective x, y, z systems do not have to share the same origin.

Nodes of the water mesh can be numbered in any order. They can be input in any order.

The 3-dimensional water element is a 20-node brick as depicted in figure 1. Nodes of the connectivity array are input in the order shown. Any of the 12 mid-edge nodes can be omitted by inserting a zero in the connectivity array. The 3-dimensional water element can take distorted shapes, and nodes can be superimposed to make triangular wedges, tetrahedra, etc. Superimposed nodes have a single node number assigned, and this node number is repeated in the connectivity array. It is recommended that the mid-edge node(s) between corner nodes to be superimposed be omitted. For the 3-dimensional water elements, 2 by 2 by 2 Gauss quadrature integration is used. The elements can be numbered and input in any order.

There is no input generation capability in RSVR2 for the nodal coordinates nor for the element connectivity arrays. These data can be generated by a separate program and written out in a file compatible with RSVR2's input specifications. RSVR2's only input generation capability is for the node types (input I4).

The water free surface elevation, watl, does not have to be the elevation of the top of the water mesh. It can be below, but it still must coincide with the elevation of a horizontal plane between water element layers. This means that the same water mesh can be used to represent different water depths without altering the mesh. All that is required in the input data file, for005, is to change the value of watl.

Input I4 assigns type –1 or +1 to nodes on the reservoir boundary. All nodes on the dam face should be designated type –1. Nodes on the reservoir floor and sides should be type +1. Those nodes on the line of intersection between the reservoir boundary at the dam face and the reservoir floor and sides are type –1. These types are input whether the nodes are located above, at, or below the elevation, watl. Other node types are set or reset by RSVR2, as follows. Nodes input as type –1 or +1 are reset to –2 or +2, respectively, if they are located at elevation watl, and to type 0, if they are above elevation watl. All remaining nodes, including interior nodes, are also set by RSVR2 to type 0. These node types are listed in output O4.

The portion of the reservoir boundary at the dam needs to be defined for the computation of the matrix [B]. RSVR2 does this automatically by identifying the set of 2-dimensional surface elements (the 4 to 8-node faces of the 3-dimensional water elements) for which the node types are all –1 and –2. The 2-dimensional surface elements so identified are listed in output O6 and designated by "dam."

The reservoir floor and sides need to be defined for the computation of the load vectors {Fx}, {Fy}, and {Fz} and the computation of the "d+r" pressures output in O7. RSVR2 does this automatically by identifying the set of 2-dimensional surface elements (the 4 to 8-node faces of the 3-dimensional water elements) for which there is at least one type +1 or type +2 node but no type 0 node. The 2-dimensional surface elements so identified are listed in output O6 and designated by "rfs." If any node on the reservoir floor and sides is not input as type +1, it is set to type 0 as stated above, and no 2-dimensional surface elements connected to this node will be included in the calculation of the force vectors {Fx}, {Fy}, and {Fz} and the "d+r" pressures. For example, it may be desirable to omit accelerations of the upstream end of the reservoir boundary.

Figure 2 shows the boundaries of an example mesh. For clarity, the interior of the mesh is not shown, and the boundaries have been separated into two pieces. The curved portion is at the dam face, and the other portions include the reservoir floor, two lateral sides, and a side at the end that forms the upstream boundary. If node types –1 and +1 are input as indicated in the figure, with the elevation watl at the base of the top element layer, RSVR2 would maintain or set/reset node types as shown in figure 3. ND would equal 26 (the number of remaining type –1 nodes plus the number of nodes reset to type –2), and NDBS would equal 19 (the number of remaining type –1 nodes). Six "dam" 2-dimensional surface elements and 21 "rfs" 2-dimensional surface elements would be listed in output O6. These elements are shown hashed and shaded, respectively, in figure 3. Note that in this example no accelerations would be included over the upstream end of the reservoir in the computation of the force vectors {Fx}, {Fy}, and {Fz} and the "d+r" pressures. If such accelerations are desired, the nodes at the upstream end of the reservoir should be input as type +1.

The node list appearing in outputs O8 through O11 is the same one written to the second record of files for001 and for002. In these files, the terms of each column of [MA], the diagonalized version of [MA], and the force vectors {Fx}, {Fy}, and {Fz} are written in the following order: x, y, and z translational degrees of freedom for the first node in the list; x, y, and z degrees of freedom for the second node in the list; .....; x, y, and z degrees of freedom for the NDth node in the list.

There is a simple check on the input data defining the water mesh, which is worthwhile to run. If node types –1 and +1 are used over the entire reservoir boundaries, the "d+r" pressures in output O7 generated by the vertical component of acceleration should equal the static water pressure. This check still works if any vertical reservoir boundary is left out, such as in the case of figures 2 and 3, assuming the upstream end of the reservoir is vertical. This check will not pick up incorrect nodal locations; the mesh should be plotted for this purpose.

Example Problem

This example problem uses the water mesh shown in figure 4. There are 12 three-dimensional water elements and 80 nodes, and the y axis is positive upward. The dam face is the front boundary of the mesh. No accelerations are included on the upstream end of the reservoir. The water free surface is at the top of the mesh. Input and output files (for005 and for006, respectively) are included in the following pages. Units are pounds and feet. Note that in output O6, 15 two-dimensional surface elements are identified: 6 on the dam face and 9 on the reservoir floor and sides. Note that in output O7, the "d+r in y dir" pressures are equal to the static water pressures. Even though no accelerations were included on the upstream end of the reservoir, this check (mentioned in section 4) works because the upstream end is vertical.

Section 5.1 contains the for005 input file for the example problem. Section 5.2 contains the for006 output file for the example problem.

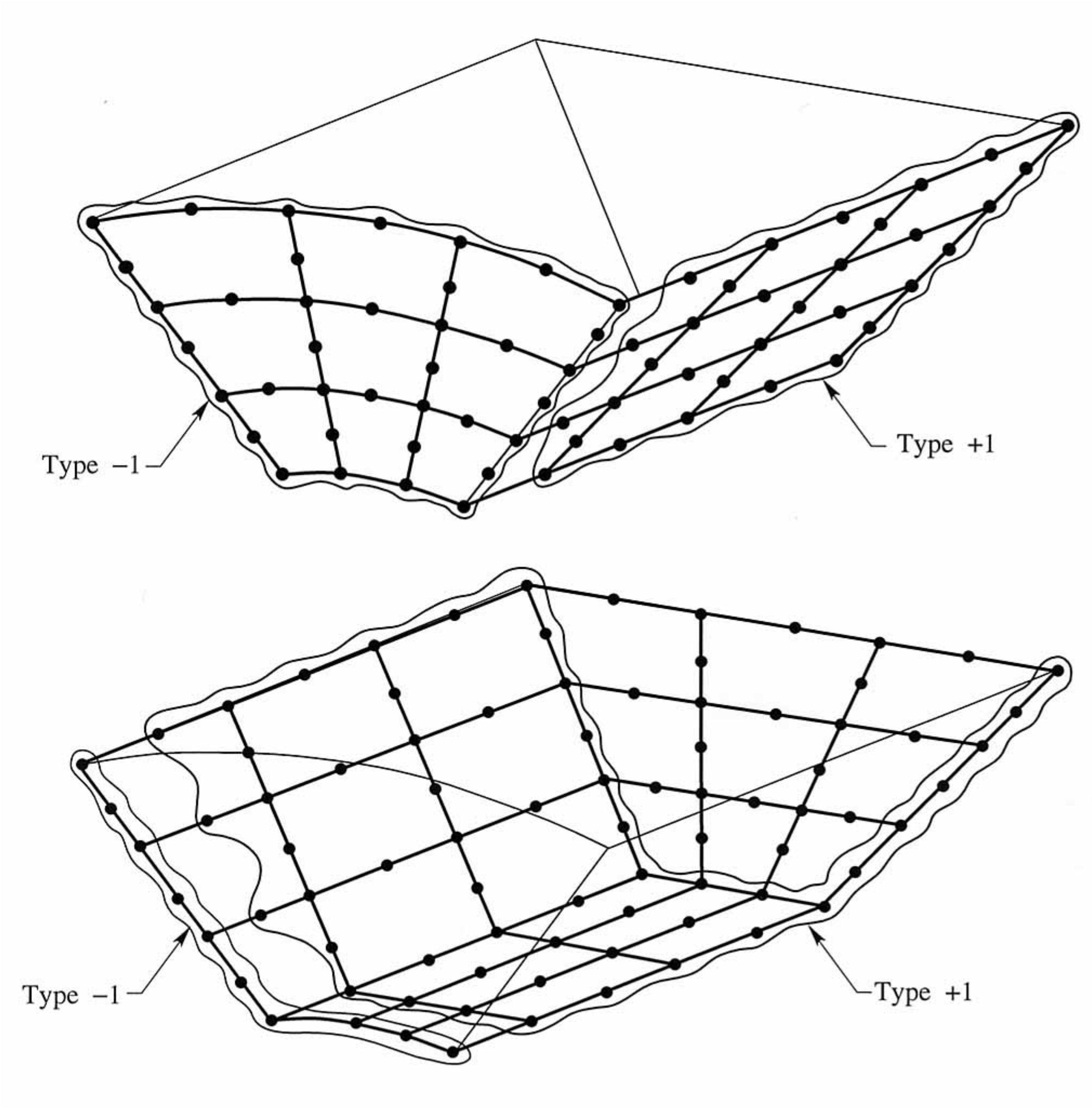

Figure 2.—Water mesh showing 2-dimensional surface elements and node types as input to RSVR2.

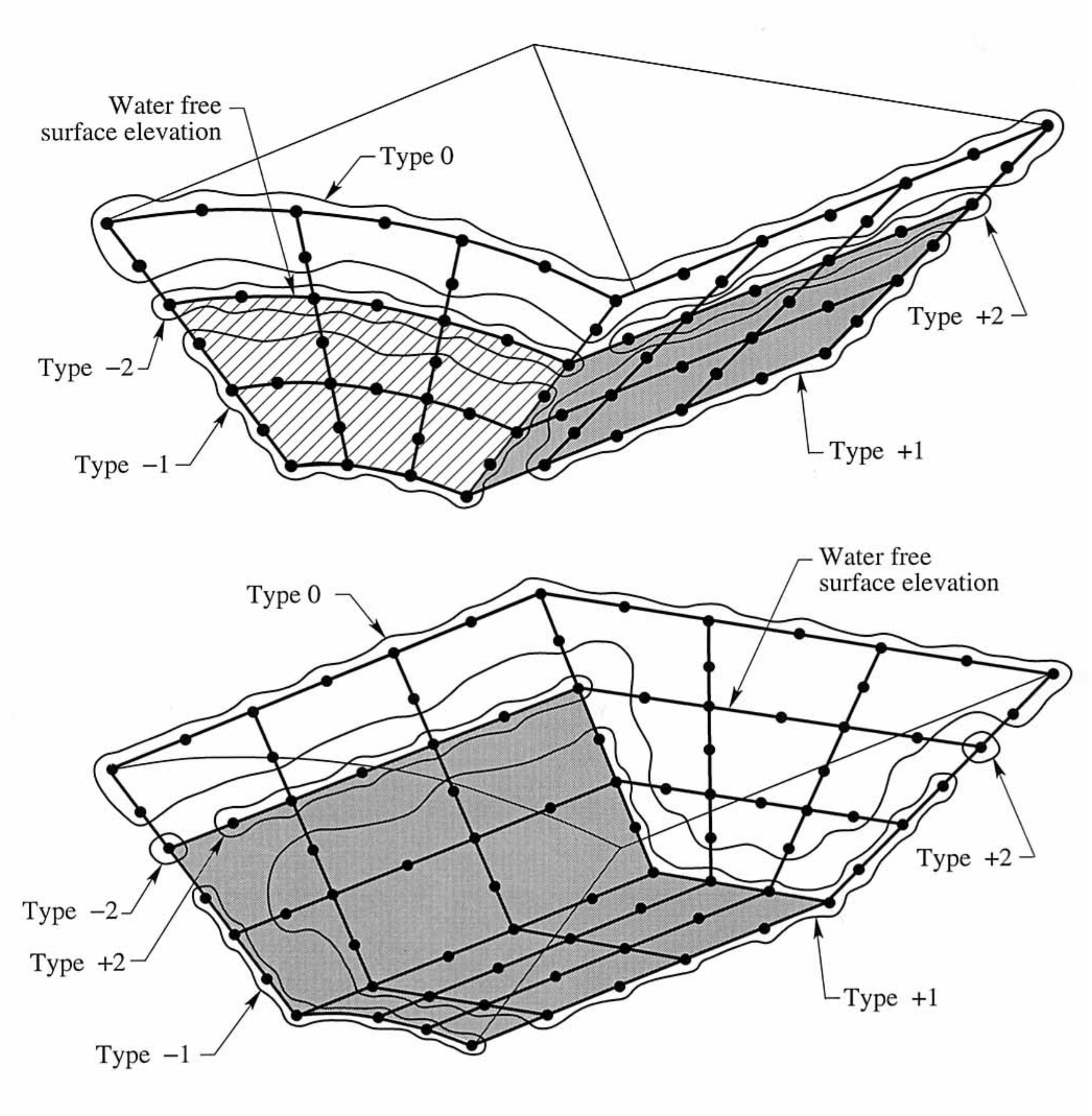

Figure 3.—Water mesh showing 2-dimensional surface elements and node types as input or set/reset by RSVR2. The water free surface coincides with the horizontal plane at the bottom of the top element layer. Hashed surface elements are at the dam face, and shaded ones are on the reservoir floor and sides where accelerations are included.

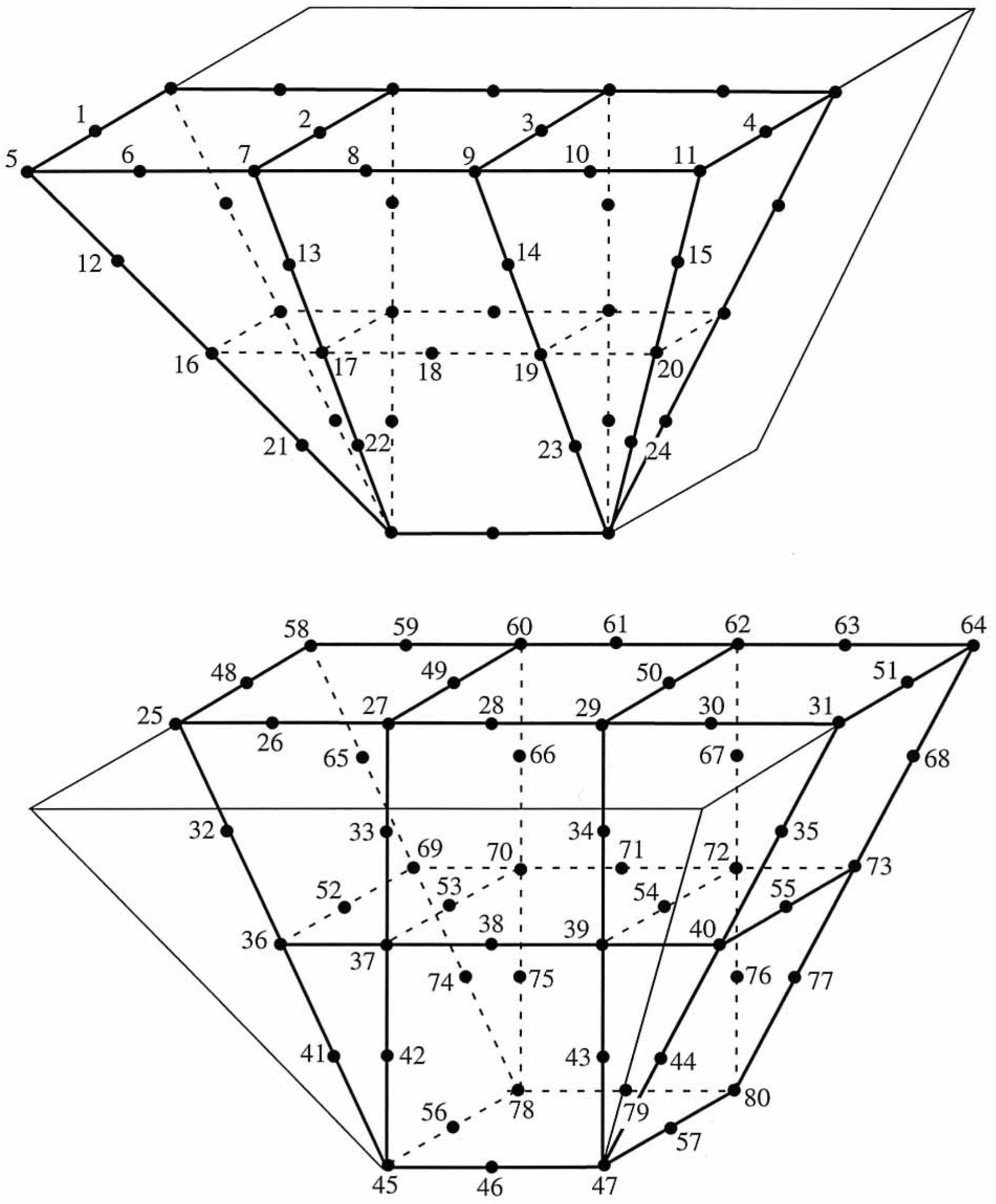

Figure 4.—Example water mesh showing node numbering and layout of 3-dimensional elements.

Example Problem Input File

test problem

```
80    12    62.4  32.2  40.   2     input I2

1     -10.  40.   -30.              input I3
2     -10.  40.   -10.              input I3
3     -10.  40.   10.               input I3
4     -10.  40.   30.               input I3
5     -20.  40.   -30.              input I3
6     -20.  40.   -20.              input I3
7     -20.  40.   -10.              input I3
8     -20.  40.   0.                input I3
9     -20.  40.   10.               input I3
10    -20.  40.   20.               input I3
11    -20.  40.   30.               input I3
12    -15.  30.   -25.              input I3
13    -15.  30.   -10.              input I3
14    -15.  30.   10.               input I3
15    -15.  30.   25.               input I3
16    -10.  20.   -20.              input I3
17    -10.  20.   -10.              input I3
18    -10.  20.   0.                input I3
19    -10.  20.   10.               input I3
20    -10.  20.   20.               input I3
21    -5.   10.   -15.              input I3
22    -5.   10.   -10.              input I3
23    -5.   10.   10.               input I3
24    -5.   10.   15.               input I3
25    0.    40.   -30.              input I3
26    0.    40.   -20.              input I3
27    0.    40.   -10.              input I3
28    0.    40.   0.                input I3
29    0.    40.   10.               input I3
30    0.    40.   20.               input I3
31    0.    40.   30.               input I3
32    0.    30.   -25.              input I3
33    0.    30.   -10.              input I3
34    0.    30.   10.               input I3
35    0.    30.   25.               input I3
36    0.    20.   -20.              input I3
37    0.    20.   -10.              input I3
38    0.    20.   0.                input I3
39    0.    20.   10.               input I3
40    0.    20.   20.               input I3
41    0.    10.   -15.              input I3
42    0.    10.   -10.              input I3
43    0.    10.   10.               input I3
44    0.    10.   15.               input I3
45    0.    0.    -10.              input I3
46    0.    0.    0.                input I3
47    0.    0.    10.               input I3
48    10.   40.   -30.              input I3
```

```
49    10.   40.   -10.              input I3
50    10.   40.   10.               input I3
51    10.   40.   30.               input I3
52    10.   20.   -20.              input I3
53    10.   20.   -10.              input I3
54    10.   20.   10.               input I3
55    10.   20.   20.               input I3
56    10.   0.    -10.              input I3
57    10.   0.    10.               input I3
58    20.   40.   -30.              input I3
59    20.   40.   -20.              input I3
60    20.   40.   -10.              input I3
61    20.   40.   0.                input I3
62    20.   40.   10.               input I3
63    20.   40.   20.               input I3
64    20.   40.   30.               input I3
65    20.   30.   -25.              input I3
66    20.   30.   -10.              input I3
67    20.   30.   10.               input I3
68    20.   30.   25.               input I3
69    20.   20.   -20.              input I3
70    20.   20.   -10.              input I3
71    20.   20.   0.                input I3
72    20.   20.   10.               input I3
73    20.   20.   20.               input I3
74    20.   10.   -15.              input I3
75    20.   10.   -10.              input I3
76    20.   10.   10.               input I3
77    20.   10.   15.               input I3
78    20.   0.    -10.              input I3
79    20.   0.    0.                input I3
80    20.   0.    10.               input I3

  5 24  1 -1                        input I4
 45 47  1 -1                        input I4
  4  0  0  1                        input I4
 31  0  0  1                        input I4
 51  0  0  1                        input I4
 64  0  0  1                        input I4
 35  0  0  1                        input I4
 68  0  0  1                        input I4
 40  0  0  1                        input I4
 55  0  0  1                        input I4
 73  0  0  1                        input I4
 44  0  0  1                        input I4
 77  0  0  1                        input I4
 57  0  0  1                        input I4
 80  0  0  1                        input I4
  1  0  0  1                        input I4
 25  0  0  1                        input I4
 48  0  0  1                        input I4
 58  0  0  1                        input I4
 32  0  0  1                        input I4
 65  0  0  1                        input I4
 36  0  0  1                        input I4
 52  0  0  1                        input I4
```

```
69  0  0  1                        input I4
41  0  0  1                        input I4
74  0  0  1                        input I4
56  0  0  1                        input I4
78  0  0  1                        input I4
 0  0  0  0                        input I4

 1   7  5 16 17 27 25 36 37  6 12  0 13 26 32  0 33  2  1  0  0   input I5
 2   9  7 17 19 29 27 37 39  8 13 18 14 28 33 38 34  3  2  0  0   input I5
 3  11  9 19 20 31 29 39 40 10 14  0 15 30 34  0 35  4  3  0  0   input I5
 4  17 16 45 45 37 36 45 45  0 21  0 22  0 41  0 42  0  0  0  0   input I5
 5  19 17 45 47 39 37 45 47 18 22 46 23 38 42 46 43  0  0  0  0   input I5
 6  20 19 47 47 40 39 47 47  0 23  0 24  0 43  0 44  0  0  0  0   input I5
 7  27 25 36 37 60 58 69 70 26 32  0 33 59 65  0 66 49 48 52 53   input I5
 8  29 27 37 39 62 60 70 72 28 33 38 34 61 66 71 67 50 49 53 54   input I5
 9  31 29 39 40 64 62 72 73 30 34  0 35 63 67  0 68 51 50 54 55   input I5
10  37 36 45 45 70 69 78 78  0 41  0 42  0 74  0 75 53 52 56 56   input I5
11  39 37 45 47 72 70 78 80 38 42 46 43 71 75 79 76 54 53 56 57   input I5
12  40 39 47 47 73 72 80 80  0 43  0 44  0 76  0 77 55 54 57 57   input I5
```

Example Problem Output File

```
O1:   test problem


O2:  nnp =    80
     nel =    12
     w = 0.10000E+01
     g = 0.10000E+01
     watl = 0.40000E+02
     iup =  2


O3:  Boundary node type data

       n1      n2     inc     itype
        5      24       1        -1
       45      47       1        -1
        4       0       0         1
       31       0       0         1
       51       0       0         1
       64       0       0         1
       35       0       0         1
       68       0       0         1
       40       0       0         1
       55       0       0         1
       73       0       0         1
       44       0       0         1
       77       0       0         1
       57       0       0         1
       80       0       0         1
        1       0       0         1
       25       0       0         1
       48       0       0         1
       58       0       0         1
       32       0       0         1
```

```
        65       0       0        1
        36       0       0        1
        52       0       0        1
        69       0       0        1
        41       0       0        1
        74       0       0        1
        56       0       0        1
        78       0       0        1


O4:  Nodal coordinates and types

     node         xcoor         ycoor         zcoor  type
        1  -0.10000E+02   0.40000E+02  -0.30000E+02     2
        2  -0.10000E+02   0.40000E+02  -0.10000E+02     0
        3  -0.10000E+02   0.40000E+02   0.10000E+02     0
        4  -0.10000E+02   0.40000E+02   0.30000E+02     2
        5  -0.20000E+02   0.40000E+02  -0.30000E+02    -2
        6  -0.20000E+02   0.40000E+02  -0.20000E+02    -2
        7  -0.20000E+02   0.40000E+02  -0.10000E+02    -2
        8  -0.20000E+02   0.40000E+02   0.00000E+00    -2
        9  -0.20000E+02   0.40000E+02   0.10000E+02    -2
       10  -0.20000E+02   0.40000E+02   0.20000E+02    -2
       11  -0.20000E+02   0.40000E+02   0.30000E+02    -2
       12  -0.15000E+02   0.30000E+02  -0.25000E+02    -1
       13  -0.15000E+02   0.30000E+02  -0.10000E+02    -1
       14  -0.15000E+02   0.30000E+02   0.10000E+02    -1
       15  -0.15000E+02   0.30000E+02   0.25000E+02    -1
       16  -0.10000E+02   0.20000E+02  -0.20000E+02    -1
       17  -0.10000E+02   0.20000E+02  -0.10000E+02    -1
       18  -0.10000E+02   0.20000E+02   0.00000E+00    -1
       19  -0.10000E+02   0.20000E+02   0.10000E+02    -1
       20  -0.10000E+02   0.20000E+02   0.20000E+02    -1
       21  -0.50000E+01   0.10000E+02  -0.15000E+02    -1
       22  -0.50000E+01   0.10000E+02  -0.10000E+02    -1
       23  -0.50000E+01   0.10000E+02   0.10000E+02    -1
       24  -0.50000E+01   0.10000E+02   0.15000E+02    -1
       25   0.00000E+00   0.40000E+02  -0.30000E+02     2
       26   0.00000E+00   0.40000E+02  -0.20000E+02     0
       27   0.00000E+00   0.40000E+02  -0.10000E+02     0
       28   0.00000E+00   0.40000E+02   0.00000E+00     0
       29   0.00000E+00   0.40000E+02   0.10000E+02     0
       30   0.00000E+00   0.40000E+02   0.20000E+02     0
       31   0.00000E+00   0.40000E+02   0.30000E+02     2
       32   0.00000E+00   0.30000E+02  -0.25000E+02     1
       33   0.00000E+00   0.30000E+02  -0.10000E+02     0
       34   0.00000E+00   0.30000E+02   0.10000E+02     0
       35   0.00000E+00   0.30000E+02   0.25000E+02     1
       36   0.00000E+00   0.20000E+02  -0.20000E+02     1
       37   0.00000E+00   0.20000E+02  -0.10000E+02     0
       38   0.00000E+00   0.20000E+02   0.00000E+00     0
       39   0.00000E+00   0.20000E+02   0.10000E+02     0
       40   0.00000E+00   0.20000E+02   0.20000E+02     1
       41   0.00000E+00   0.10000E+02  -0.15000E+02     1
       42   0.00000E+00   0.10000E+02  -0.10000E+02     0
       43   0.00000E+00   0.10000E+02   0.10000E+02     0
       44   0.00000E+00   0.10000E+02   0.15000E+02     1
       45   0.00000E+00   0.00000E+00  -0.10000E+02    -1
       46   0.00000E+00   0.00000E+00   0.00000E+00    -1
```

```
 47   0.00000E+00   0.00000E+00   0.10000E+02    -1
 48   0.10000E+02   0.40000E+02  -0.30000E+02     2
 49   0.10000E+02   0.40000E+02  -0.10000E+02     0
 50   0.10000E+02   0.40000E+02   0.10000E+02     0
 51   0.10000E+02   0.40000E+02   0.30000E+02     2
 52   0.10000E+02   0.20000E+02  -0.20000E+02     1
 53   0.10000E+02   0.20000E+02  -0.10000E+02     0
 54   0.10000E+02   0.20000E+02   0.10000E+02     0
 55   0.10000E+02   0.20000E+02   0.20000E+02     1
 56   0.10000E+02   0.00000E+00  -0.10000E+02     1
 57   0.10000E+02   0.00000E+00   0.10000E+02     1
 58   0.20000E+02   0.40000E+02  -0.30000E+02     2
 59   0.20000E+02   0.40000E+02  -0.20000E+02     0
 60   0.20000E+02   0.40000E+02  -0.10000E+02     0
 61   0.20000E+02   0.40000E+02   0.00000E+00     0
 62   0.20000E+02   0.40000E+02   0.10000E+02     0
 63   0.20000E+02   0.40000E+02   0.20000E+02     0
 64   0.20000E+02   0.40000E+02   0.30000E+02     2
 65   0.20000E+02   0.30000E+02  -0.25000E+02     1
 66   0.20000E+02   0.30000E+02  -0.10000E+02     0
 67   0.20000E+02   0.30000E+02   0.10000E+02     0
 68   0.20000E+02   0.30000E+02   0.25000E+02     1
 69   0.20000E+02   0.20000E+02  -0.20000E+02     1
 70   0.20000E+02   0.20000E+02  -0.10000E+02     0
 71   0.20000E+02   0.20000E+02   0.00000E+00     0
 72   0.20000E+02   0.20000E+02   0.10000E+02     0
 73   0.20000E+02   0.20000E+02   0.20000E+02     1
 74   0.20000E+02   0.10000E+02  -0.15000E+02     1
 75   0.20000E+02   0.10000E+02  -0.10000E+02     0
 76   0.20000E+02   0.10000E+02   0.10000E+02     0
 77   0.20000E+02   0.10000E+02   0.15000E+02     1
 78   0.20000E+02   0.00000E+00  -0.10000E+02     1
 79   0.20000E+02   0.00000E+00   0.00000E+00     0
 80   0.20000E+02   0.00000E+00   0.10000E+02     1
  1  -0.10000E+02   0.40000E+02  -0.30000E+02     0
  2  -0.10000E+02   0.40000E+02  -0.10000E+02     0
  3  -0.10000E+02   0.40000E+02   0.10000E+02     0
  4  -0.10000E+02   0.40000E+02   0.30000E+02     0
  5  -0.20000E+02   0.40000E+02  -0.30000E+02    -1
  6  -0.20000E+02   0.40000E+02  -0.20000E+02    -2
  7  -0.20000E+02   0.40000E+02  -0.10000E+02    -3
  8  -0.20000E+02   0.40000E+02   0.00000E+00    -4
  9  -0.20000E+02   0.40000E+02   0.10000E+02    -5
 10  -0.20000E+02   0.40000E+02   0.20000E+02    -6
 11  -0.20000E+02   0.40000E+02   0.30000E+02    -7
 12  -0.15000E+02   0.30000E+02  -0.25000E+02    36
 13  -0.15000E+02   0.30000E+02  -0.10000E+02    37
 14  -0.15000E+02   0.30000E+02   0.10000E+02    38
 15  -0.15000E+02   0.30000E+02   0.25000E+02    39
 16  -0.10000E+02   0.20000E+02  -0.20000E+02    40
 17  -0.10000E+02   0.20000E+02  -0.10000E+02    41
 18  -0.10000E+02   0.20000E+02   0.00000E+00    42
 19  -0.10000E+02   0.20000E+02   0.10000E+02    43
 20  -0.10000E+02   0.20000E+02   0.20000E+02    44
 21  -0.50000E+01   0.10000E+02  -0.15000E+02    45
 22  -0.50000E+01   0.10000E+02  -0.10000E+02    46
 23  -0.50000E+01   0.10000E+02   0.10000E+02    47
```

```
24  -0.50000E+01   0.10000E+02   0.15000E+02    48
25   0.00000E+00   0.40000E+02  -0.30000E+02     0
26   0.00000E+00   0.40000E+02  -0.20000E+02     0
27   0.00000E+00   0.40000E+02  -0.10000E+02     0
28   0.00000E+00   0.40000E+02   0.00000E+00     0
29   0.00000E+00   0.40000E+02   0.10000E+02     0
30   0.00000E+00   0.40000E+02   0.20000E+02     0
31   0.00000E+00   0.40000E+02   0.30000E+02     0
32   0.00000E+00   0.30000E+02  -0.25000E+02     1
33   0.00000E+00   0.30000E+02  -0.10000E+02     2
34   0.00000E+00   0.30000E+02   0.10000E+02     3
35   0.00000E+00   0.30000E+02   0.25000E+02     4
36   0.00000E+00   0.20000E+02  -0.20000E+02     5
37   0.00000E+00   0.20000E+02  -0.10000E+02     6
38   0.00000E+00   0.20000E+02   0.00000E+00     7
39   0.00000E+00   0.20000E+02   0.10000E+02     8
40   0.00000E+00   0.20000E+02   0.20000E+02     9
41   0.00000E+00   0.10000E+02  -0.15000E+02    10
42   0.00000E+00   0.10000E+02  -0.10000E+02    11
43   0.00000E+00   0.10000E+02   0.10000E+02    12
44   0.00000E+00   0.10000E+02   0.15000E+02    13
45   0.00000E+00   0.00000E+00  -0.10000E+02    49
46   0.00000E+00   0.00000E+00   0.00000E+00    50
47   0.00000E+00   0.00000E+00   0.10000E+02    51
48   0.10000E+02   0.40000E+02  -0.30000E+02     0
49   0.10000E+02   0.40000E+02  -0.10000E+02     0
50   0.10000E+02   0.40000E+02   0.10000E+02     0
51   0.10000E+02   0.40000E+02   0.30000E+02     0
52   0.10000E+02   0.20000E+02  -0.20000E+02    14
53   0.10000E+02   0.20000E+02  -0.10000E+02    15
54   0.10000E+02   0.20000E+02   0.10000E+02    16
55   0.10000E+02   0.20000E+02   0.20000E+02    17
56   0.10000E+02   0.00000E+00  -0.10000E+02    18
57   0.10000E+02   0.00000E+00   0.10000E+02    19
58   0.20000E+02   0.40000E+02  -0.30000E+02     0
59   0.20000E+02   0.40000E+02  -0.20000E+02     0
60   0.20000E+02   0.40000E+02  -0.10000E+02     0
61   0.20000E+02   0.40000E+02   0.00000E+00     0
62   0.20000E+02   0.40000E+02   0.10000E+02     0
63   0.20000E+02   0.40000E+02   0.20000E+02     0
64   0.20000E+02   0.40000E+02   0.30000E+02     0
65   0.20000E+02   0.30000E+02  -0.25000E+02    20
66   0.20000E+02   0.30000E+02  -0.10000E+02    21
67   0.20000E+02   0.30000E+02   0.10000E+02    22
68   0.20000E+02   0.30000E+02   0.25000E+02    23
69   0.20000E+02   0.20000E+02  -0.20000E+02    24
70   0.20000E+02   0.20000E+02  -0.10000E+02    25
71   0.20000E+02   0.20000E+02   0.00000E+00    26
72   0.20000E+02   0.20000E+02   0.10000E+02    27
73   0.20000E+02   0.20000E+02   0.20000E+02    28
74   0.20000E+02   0.10000E+02  -0.15000E+02    29
75   0.20000E+02   0.10000E+02  -0.10000E+02    30
76   0.20000E+02   0.10000E+02   0.10000E+02    31
77   0.20000E+02   0.10000E+02   0.15000E+02    32
78   0.20000E+02   0.00000E+00  -0.10000E+02    33
79   0.20000E+02   0.00000E+00   0.00000E+00    34
80   0.20000E+02   0.00000E+00   0.10000E+02    35
```

```
O5:  3-d element connectivity arrays

  element                    node list
       1      7    5   16   17   27   25   36   37
              6   12    0   13   26   32    0   33    2    1    0    0
         vol = 0.46667E+04
       2      9    7   17   19   29   27   37   39
              8   13   18   14   28   33   38   34    3    2    0    0
         vol = 0.60000E+04
       3     11    9   19   20   31   29   39   40
             10   14    0   15   30   34    0   35    4    3    0    0
         vol = 0.46667E+04
       4     17   16   45   45   37   36   45   45
              0   21    0   22    0   41    0   42    0    0    0    0
         vol = 0.66667E+03
       5     19   17   45   47   39   37   45   47
             18   22   46   23   38   42   46   43    0    0    0    0
         vol = 0.20000E+04
       6     20   19   47   47   40   39   47   47
              0   23    0   24    0   43    0   44    0    0    0    0
         vol = 0.66667E+03
       7     27   25   36   37   60   58   69   70
             26   32    0   33   59   65    0   66   49   48   52   53
         vol = 0.60000E+04
       8     29   27   37   39   62   60   70   72
             28   33   38   34   61   66   71   67   50   49   53   54
         vol = 0.80000E+04
       9     31   29   39   40   64   62   72   73
             30   34    0   35   63   67    0   68   51   50   54   55
         vol = 0.60000E+04
      10     37   36   45   45   70   69   78   78
              0   41    0   42    0   74    0   75   53   52   56   56
         vol = 0.20000E+04
      11     39   37   45   47   72   70   78   80
             38   42   46   43   71   75   79   76   54   53   56   57
         vol = 0.80000E+04
      12     40   39   47   47   73   72   80   80
              0   43    0   44    0   76    0   77   55   54   57   57
         vol = 0.20000E+04


O6:  2-d surface element connectivity arrays

  element  type                    node list
       1    dam      7    5   16   17    6   12    0   13
               area = 0.33541E+03
       2    rfs      5   25   36   16    1   32    0   12
               area = 0.33541E+03
       3    dam      9    7   17   19    8   13   18   14
               area = 0.44721E+03
       4    dam     11    9   19   20   10   14    0   15
               area = 0.33541E+03
       5    rfs     31   11   20   40    4   15    0   35
               area = 0.33541E+03
       6    dam     17   16   45   45    0   21    0   22
               area = 0.11180E+03
       7    rfs     16   36   45   45    0   41    0   21
               area = 0.11180E+03
```

```
          8   dam     19   17   45   47   18   22   46   23
                   area = 0.44721E+03
          9   dam     20   19   47   47    0   23    0   24
                   area = 0.11180E+03
         10   rfs     40   20   47   47    0   24    0   44
                   area = 0.11180E+03
         11   rfs     25   58   69   36   48   65   52   32
                   area = 0.44721E+03
         12   rfs     64   31   40   73   51   35   55   68
                   area = 0.44721E+03
         13   rfs     36   69   78   45   52   74   56   41
                   area = 0.44721E+03
         14   rfs     78   80   47   45   79   57   46   56
                   area = 0.40000E+03
         15   rfs     73   40   47   80   55   44   57   77
                   area = 0.44721E+03


 O7:  Pressures resulting from 1g accel of dam and dam plus reservoir floor
and sides in x, y, and z directions

  node  dam in x dir  dam in y dir  dam in z dir  d+r in x dir  d+r in y dir
d+r in z dir
    12   0.87178E+01   0.43589E+01   0.00000E+00   0.87178E+01   0.10000E+02
0.75832E+01
    13   0.11483E+02   0.57415E+01   0.00000E+00   0.11483E+02   0.10000E+02
0.17832E+01
    14   0.11483E+02   0.57415E+01   0.00000E+00   0.11483E+02   0.10000E+02
-0.17832E+01
    15   0.87178E+01   0.43589E+01   0.00000E+00   0.87178E+01   0.10000E+02
-0.75832E+01
    16   0.15190E+02   0.75951E+01   0.00000E+00   0.15190E+02   0.20000E+02
0.11086E+02
    17   0.17218E+02   0.86088E+01   0.00000E+00   0.17218E+02   0.20000E+02
0.49880E+01
    18   0.17770E+02   0.88851E+01   0.00000E+00   0.17770E+02   0.20000E+02
-0.53291E-14
    19   0.17218E+02   0.86088E+01   0.00000E+00   0.17218E+02   0.20000E+02
-0.49880E+01
    20   0.15190E+02   0.75951E+01   0.00000E+00   0.15190E+02   0.20000E+02
-0.11086E+02
    21   0.19717E+02   0.98587E+01   0.00000E+00   0.19717E+02   0.30000E+02
0.11546E+02
    22   0.20679E+02   0.10339E+02   0.00000E+00   0.20679E+02   0.30000E+02
0.73284E+01
    23   0.20679E+02   0.10339E+02   0.00000E+00   0.20679E+02   0.30000E+02
-0.73284E+01
    24   0.19717E+02   0.98587E+01   0.00000E+00   0.19717E+02   0.30000E+02
-0.11546E+02
    45   0.20157E+02   0.10079E+02   0.00000E+00   0.20157E+02   0.40000E+02
0.84466E+01
    46   0.20719E+02   0.10359E+02   0.00000E+00   0.20719E+02   0.40000E+02
-0.18208E-13
    47   0.20157E+02   0.10079E+02   0.00000E+00   0.20157E+02   0.40000E+02
-0.84466E+01
```

```
 O8:  Diagonalized added masses

      node        x mass        y mass        z mass
         5   0.42740E+02   0.10685E+02   0.00000E+00
         6   0.48069E+03   0.12017E+03   0.00000E+00
         7   0.14350E+03   0.35876E+02   0.00000E+00
         8   0.50196E+03   0.12549E+03   0.00000E+00
         9   0.14350E+03   0.35876E+02   0.00000E+00
        10   0.48069E+03   0.12017E+03   0.00000E+00
        11   0.42740E+02   0.10685E+02   0.00000E+00
        12   0.61556E+03   0.15389E+03   0.00000E+00
        13   0.19623E+04   0.49058E+03   0.00000E+00
        14   0.19623E+04   0.49058E+03   0.00000E+00
        15   0.61556E+03   0.15389E+03   0.00000E+00
        16   0.23475E+03   0.58689E+02   0.00000E+00
        17   0.40712E+03   0.10178E+03   0.00000E+00
        18   0.41248E+04   0.10312E+04   0.00000E+00
        19   0.40712E+03   0.10178E+03   0.00000E+00
        20   0.23475E+03   0.58689E+02   0.00000E+00
        21   0.21138E+03   0.52846E+02   0.00000E+00
        22   0.29447E+04   0.73617E+03   0.00000E+00
        23   0.29447E+04   0.73617E+03   0.00000E+00
        24   0.21138E+03   0.52846E+02   0.00000E+00
        45   0.17669E+03   0.44172E+02   0.00000E+00
        46   0.22663E+04   0.56657E+03   0.00000E+00
        47   0.17669E+03   0.44172E+02   0.00000E+00


 O9:  Additional forces to dam from unit accel of the reservoir floor and
sides in the x direction

      node        x load        y load        z load
         5   0.41129E-14   0.20564E-14   0.00000E+00
         6  -0.15888E-13  -0.79439E-14   0.00000E+00
         7   0.11081E-14   0.55407E-15   0.00000E+00
         8   0.84219E-14   0.42109E-14   0.00000E+00
         9  -0.18332E-13  -0.91662E-14   0.00000E+00
        10   0.34192E-13   0.17096E-13   0.00000E+00
        11  -0.12886E-13  -0.64430E-14   0.00000E+00
        12  -0.25403E-13  -0.12702E-13   0.00000E+00
        13  -0.17880E-15  -0.89399E-16   0.00000E+00
        14   0.51093E-13   0.25547E-13   0.00000E+00
        15   0.53210E-13   0.26605E-13   0.00000E+00
        16  -0.84345E-14  -0.42172E-14   0.00000E+00
        17  -0.11797E-13  -0.58984E-14   0.00000E+00
        18   0.48917E-13   0.24459E-13   0.00000E+00
        19   0.16706E-13   0.83529E-14   0.00000E+00
        20   0.35706E-13   0.17853E-13   0.00000E+00
        21   0.38888E-14   0.19444E-14   0.00000E+00
        22   0.27941E-13   0.13971E-13   0.00000E+00
        23   0.62832E-13   0.31416E-13   0.00000E+00
        24   0.26287E-13   0.13143E-13   0.00000E+00
        45  -0.99906E-14  -0.49953E-14   0.00000E+00
        46   0.30686E-13   0.15343E-13   0.00000E+00
        47  -0.26868E-14  -0.13434E-14   0.00000E+00
 O10:  Additional forces to dam from unit accel of the res floor and sides in
the y dir

      node        x load        y load        z load
         5   0.14673E+03   0.73367E+02   0.00000E+00
         6  -0.32258E+03  -0.16129E+03   0.00000E+00
         7   0.37127E+03   0.18563E+03   0.00000E+00
         8  -0.37034E+03  -0.18517E+03   0.00000E+00
         9   0.37127E+03   0.18563E+03   0.00000E+00
        10  -0.32258E+03  -0.16129E+03   0.00000E+00
        11   0.14673E+03   0.73367E+02   0.00000E+00
        12  -0.47943E+03  -0.23971E+03   0.00000E+00
        13  -0.10643E+04  -0.53217E+03   0.00000E+00
        14  -0.10643E+04  -0.53217E+03   0.00000E+00
        15  -0.47943E+03  -0.23971E+03   0.00000E+00
        16  -0.46461E+03  -0.23231E+03   0.00000E+00
        17   0.40554E+03   0.20277E+03   0.00000E+00
        18  -0.30976E+04  -0.15488E+04   0.00000E+00
        19   0.40554E+03   0.20277E+03   0.00000E+00
        20  -0.46461E+03  -0.23231E+03   0.00000E+00
        21  -0.57810E+03  -0.28905E+03   0.00000E+00
        22  -0.32117E+04  -0.16059E+04   0.00000E+00
        23  -0.32117E+04  -0.16059E+04   0.00000E+00
        24  -0.57810E+03  -0.28905E+03   0.00000E+00
        45   0.45717E+03   0.22858E+03   0.00000E+00
        46  -0.30526E+04  -0.15263E+04   0.00000E+00
        47   0.45717E+03   0.22858E+03   0.00000E+00


 O11:  Additional forces to dam from unit accel of the reservoir floor and
sides in the z direction

      node        x load        y load        z load
         5   0.10846E+03   0.54231E+02   0.00000E+00
         6  -0.31714E+03  -0.15857E+03   0.00000E+00
         7   0.13975E+03   0.69875E+02   0.00000E+00
         8   0.22737E-12   0.11369E-12   0.00000E+00
         9  -0.13975E+03  -0.69875E+02   0.00000E+00
        10   0.31714E+03   0.15857E+03   0.00000E+00
        11  -0.10846E+03  -0.54231E+02   0.00000E+00
        12  -0.47725E+03  -0.23863E+03   0.00000E+00
        13  -0.41556E+03  -0.20778E+03   0.00000E+00
        14   0.41556E+03   0.20778E+03   0.00000E+00
        15   0.47725E+03   0.23863E+03   0.00000E+00
        16  -0.35317E+03  -0.17659E+03   0.00000E+00
        17  -0.38100E+03  -0.19050E+03   0.00000E+00
        18   0.20748E-11   0.10374E-11   0.00000E+00
        19   0.38100E+03   0.19050E+03   0.00000E+00
        20   0.35317E+03   0.17659E+03   0.00000E+00
        21  -0.32593E+03  -0.16297E+03   0.00000E+00
        22  -0.58876E+03  -0.29438E+03   0.00000E+00
        23   0.58876E+03   0.29438E+03   0.00000E+00
        24   0.32593E+03   0.16297E+03   0.00000E+00
        45  -0.10065E+03  -0.50326E+02   0.00000E+00
        46   0.21600E-11   0.10800E-11   0.00000E+00
        47   0.10065E+03   0.50326E+02   0.00000E+00
```

Listing for RSVR2

```
      program rsvr2
      implicit real*8 (a-h,o-z)
c
      common a(2000000)
      common /shape/ head(20),dum(202)
      open(1,file='for001',form='unformatted')
      open(2,file='for002',form='unformatted')
      open(3,file='for003',form='unformatted')
      open(5,file='for005',form='formatted')
      open(6,file='for006',form='formatted')
c
      mtot=2000000
      idiag=1
      ielmt=2
      ielmt2d=3
c
      read(5,1000) head
 1000 format(20a4)
      write(6,1001) head
 1001 format(///,1x,'O1:  ',20a4,//)
      read(5,*) nnp,nel,w,g,watl,iup
      if(nnp.eq.0) stop
      write(6,1002) nnp,nel,w,g,watl,iup
 1002 format(1x,'O2:  nnp =',i6,/,6x,'nel =',i6,/,6x,'w =',e12.5,/,
     1 6x,'g =',e12.5,/,6x,'watl =',e12.5,/,6x,'iup =',i3)
c
      n1=1
      n2=n1+3*nnp
      n3=n2+nnp
      n4=n3+nnp
      n5=n4+nnp
      if(n5.gt.mtot) call error(n5)
      call nodes(a(n1),a(n2),a(n3),a(n4),nnp,watl,iup,neq,nes,ner,net)
c
      n6=n5+neq
      if(n6.gt.mtot) call error(n6)
      call fluide(a(n1),a(n3),a(n5),nnp,nel,ielmt,neq)
      call diagad(a(n5),a(n5),inode,neq,nwk)
      call intact(a(n1),a(n2),a(n3),nnp,nel,ielmt,ielmt2d,n2del)
c
      net3=net*3
      n7=n6+nes*net3
      n8=n7+neq*3
      if(n8.gt.mtot) call error(n8)
      call assembf(a(n6),a(n7),nnp,n2del,ielmt2d,neq,ner,nes,net,
     1 net3)
c
      n9=n8+nwk
      if(n9.gt.mtot) call error(n9)
      call assemxk(a(n5),a(n8),nel,ielmt,neq,nwk)
      call scondn(a(n5),a(n7),a(n8),neq,ner,nwk,nea)
      call subff(a(n5),a(n5),a(n7),a(n7),a(n8),a(n8),neq,nes,nwk,nea)
      call solutn(a(n4),a(n5),a(n6),a(n7),a(n8),w,nes,net,net3,nea)
c
      net3p3=net3+3
      n10=n8+nes*net3
      n11=n10+net3
      n12=n11+net3
      if(n12.gt.mtot) call error(n12)
      call addms(a(n4),a(n6),a(n8),a(n10),a(n11),w,g,ielmt,
     1 ielmt2d,idiag,nes,net,net3,net3p3)
      stop
      end

      subroutine addms(nlist,xkibf,b,col,diag,w,g,ielmt,ielmt2d,
     1 idiag,nes,net,net3,net3p3)
      implicit real*8 (a-h,o-z)
      dimension nlist(net),xkibf(nes,net3p3),b(nes,net3),col(net3),
     1 diag(net3),umu(3),udu(3)
c
      rewind ielmt
      rewind ielmt2d
      write(ielmt) net
      write(ielmt) nlist
      read(ielmt2d) b
      call zero(umu,3,1)
      call zero(udu,3,1)
      ii=1
      do 100 k=1,net3p3
      do 120 i=1,net3
      c=0.0d0
      do 140 j=1,nes
  140 c=c+b(j,i)*xkibf(j,k)
      col(i)=c*w/g
      if(k.gt.net3) col(i)=-col(i)
  120 continue
      write(ielmt) col
      if(k.gt.net3) go to 100
      do 160 i=ii,net3,3
  160 umu(ii)=umu(ii)+col(i)
      udu(ii)=udu(ii)+col(k)
      ii=ii+1
      if(ii.gt.3) ii=1
  100 continue
c
      do 200 i=1,3
      if(udu(i).eq.0.0d0) udu(i)=1.d0
  200 umu(i)=umu(i)/udu(i)
      ii=1
      rewind ielmt
      read(ielmt)
      read(ielmt)
      do 300 k=1,net3
      read (ielmt) col
      diag(k)=col(k)*umu(ii)
      ii=ii+1
  300 if(ii.gt.3) ii=1
      rewind idiag
      write(idiag) net
      write(idiag) nlist
      write(idiag) diag
      write(6,1000)
 1000 format(//,1x,'O8:  Diagonalized added masses',//,5x,' node',
     1 8x,'x mass',8x,'y mass',8x,'z mass')
      do 400 i=1,net
      ia=(i-1)*3
      write(6,1001) nlist(i),diag(ia+1),diag(ia+2),diag(ia+3)
 1001 format(5x,i5,2x,e12.5,2x,e12.5,2x,e12.5)
  400 continue
c
      read(ielmt) col
      write(idiag) col
      write(6,1002)
 1002 format(//,1x,'O9:  Additional forces to dam from unit accel',
     1 ' of the res floor and sides in the x dir',//,5x,' node',8x,
     1 'x load',8x,'y load',8x,'z load')
      do 500 i=1,net
      ia=(i-1)*3
      write(6,1003) nlist(i),col(ia+1),col(ia+2),col(ia+3)
 1003 format(5x,i5,2x,e12.5,2x,e12.5,2x,e12.5)
  500 continue
      read(ielmt) col
      write(idiag) col
      write(6,1004)
 1004 format(//,1x,'O10:  Additional forces to dam from unit accel',
     1 ' of the res floor and sides in the y dir',//,5x,' node',8x,
     1 'x load',8x,'y load',8x,'z load')
      do 600 i=1,net
      ia=(i-1)*3
      write(6,1003) nlist(i),col(ia+1),col(ia+2),col(ia+3)
  600 continue
      read(ielmt) col
      write(idiag) col
      write(6,1005)
 1005 format(//,1x,'O11:  Additional forces to dam from unit accel',
     1 ' of the res floor and sides in the z dir',//,5x,' node',8x,
     1 'x load',8x,'y load',8x,'z load')
      do 700 i=1,net
      ia=(i-1)*3
      write(6,1003) nlist(i),col(ia+1),col(ia+2),col(ia+3)
  700 continue
c
      return
      end

      subroutine assembf(b,f,nnp,n2del,ielmt2d,neq,ner,nes,net,net3)
      implicit real*8 (a-h,o-z)
      dimension b(nes,net3),f(neq,3),be(8,24),ic2de(8),id2de(8)
c
      rewind ielmt2d
      call zero(b,nes,net3)
      call zero(f,neq,3)
      if(n2del.eq.0) go to 199
      do 100 n=1,n2del
      read(ielmt2d) ncol,ic2de,id2de,((be(k,l),k=1,8),l=1,ncol)
      nrr=neq-net
      do 120 i=1,8
      k=id2de(i)
      if(k.le.0) go to 120
      if(ncol.eq.24) then
        k=k-ner
        do 140 j=1,8
        l=id2de(j)
        if(l.eq.0) go to 140
        if(l.lt.0) then
        l=-id2de(j)
        else
        l=l-nrr
        endif
        lj=3*(l-1)
        kj=3*(j-1)
        do 160 m=1,3
        lj=lj+1
        kj=kj+1
  160   b(k,lj)=b(k,lj)+be(i,kj)
  140   continue
      else if(ncol.eq.3) then
        do 180 m=1,3
  180   f(k,m)=f(k,m)+be(i,m)
      endif
  120 continue
  100 continue
  199 rewind ielmt2d
      write(ielmt2d) b
      return
      end

      subroutine assemxk(lc,xk,nel,ielmt,neq,nwk)
      implicit real*8 (a-h,o-z)
      dimension lc(neq),xk(nwk),xke(20,20),ice(20),ide(20)
c
      rewind ielmt
      call zero(xk,nwk,1)
      do 100 n=1,nel
      read(ielmt) ice,ide,xke
      do 110 j=1,20
      mm=ide(j)
      if(mm.le.0) go to 110
      do 120 i=1,20
      kk=ide(i)
      if(kk.le.0) go to 120
      me=mm-kk
      if(me) 120,140,140
  140 ln=lc(mm)-me
      mk=mm-1
      if(mk-1) 150,170,170
  150 if(ln.ne.1) go to 199
      go to 180
  170 if(ln.le.lc(mk)) go to 199
  180 xk(ln)=xk(ln)+xke(i,j)
  120 continue
  110 continue
  100 continue
      return
c
  199 write(6,600) n
  600 format(/,1x,'array storage error in subroutine assemxk for'
     1 ' element',i6)
      stop
      end

      subroutine diagad(mht,lc,inode,neq,nwk)
      implicit real*8 (a-h,o-z)
      dimension mht(neq),lc(neq)
      lc(1)=1
      do 100 i=2,neq
      j=i-1
      if(mht(i).gt.j) go to 200
      lc(i)=lc(j)+mht(i)+1
  100 continue
      nwk=lc(neq)
      return
  200 write(6,300)
  300 format(//,1x,'error in subroutine diagad')
      stop
      end

      subroutine fluide(xyz,id,mht,nnp,nel,ielmt,neq)
      implicit real*8 (a-h,o-z)
      common /shape/ e1,e2,e3,det,h(20),g(3,20),xx(3,20),aj(3,3),
     1 bj(3,3),d(3,20)
      dimension xyz(3,nnp),id(nnp),mht(neq),xke(20,20),ice(20),
     1 ide(20),gloc(2)
      data gloc /0.5773502691896d0,-0.5773502691896d0/
c
      rewind ielmt
      write(6,1000)
 1000 format(//,1x,'O5:  3-d element connectivity arrays',//,3x,
     1 'element',20x,'node list')
      call izero(mht,neq,1)
      do 100 n=1,nel
      read(5,*) nel,ice
      write(6,1001) nel,ice
 1001 format(5x,i5,2x,8i5,/,12x,12i5)
c
      vol=0.0d0
      call zero(xx,3,20)
      call zero(xke,20,20)
      do 200 k=1,20
      node=ice(k)
      if(node.eq.0) go to 200
      do 205 l=1,3
  205 xx(l,k)=xyz(l,node)
  200 continue
      do 300 lr=1,2
      e1=gloc(lr)
      do 300 ls=1,2
      e2=gloc(ls)
      do 300 lt=1,2
      e3=gloc(lt)
      call shap3d(ice)
      vol=vol+det
      do 310 i=1,20
      do 310 j=1,20
      do 310 k=1,3
  310 xke(i,j)=xke(i,j)+d(k,i)*d(k,j)*det
  300 continue
      write(6,1002) vol
 1002 format(12x,'vol =',e12.5)
c
      do 750 i=1,20
      node=ice(i)
      if(node.eq.0) ide(i)=0
      if(node.ne.0) ide(i)=id(node)
  750 continue
      write(ielmt) ice,ide,xke
c
      min=100000
      do 820 i=1,20
      if(ide(i).le.0) go to 820
      if(ide(i).lt.min) min=ide(i)
  820 continue
      do 850 i=1,20
      ii=ide(i)
      if(ii.le.0) go to 850
      me=ii-min
      if(me.gt.mht(ii)) mht(ii)=me
  850 continue
c
  100 continue
      return
      end

      subroutine intact(xyz,itype,id,nnp,nel,ielmt,ielmt2d,n2del)
      implicit real*8 (a-h,o-z)
      common /shape/ e1,e2,det,h(8),g(2,8),xx(3,8),aj(2,3),bn(3),
     1 dum(162)
      dimension xyz(3,nnp),itype(nnp),id(nnp),be(8,24),ice(20),
     1 ic2de(8),id2de(8),ln6f(8,6),gloc(2)
      data ln6f /1,2,3,4,9,10,11,12,6,5,8,7,13,16,15,14,
     1     5,1,4,8,17,12,20,16,5,6,2,1,13,18,9,17,
     1     2,6,7,3,18,14,19,10,7,8,4,3,15,20,11,19/
      data gloc /0.5773502691896d0,-0.5773502691896d0/
c
      rewind ielmt
      rewind ielmt2d
      n2del=0
      write(6,1000)
 1000 format(//,1x,'O6:  2-d surface element connectivity arrays',//,3x,
     1 'element',2x,'type',20x,'node list')
      do 100 n=1,nel
      read(ielmt) ice
      do 102 j=1,6
      do 103 k=1,8
      lnode=ln6f(k,j)
  103 ic2de(k)=ice(lnode)
      ict=0
      do 104 k=1,3
      do 104 l=k+1,4
 104  if(ic2de(k).ne.ic2de(l)) ict=ict+1
      if(ict.le.4) go to 102
      do 105 k=1,8
      node=ic2de(k)
      if(node.eq.0) go to 105
      if(itype(node).ge.0) go to 109
  105 continue
      ncol=24
      go to 112
  109 ict=0
      do 110 k=1,4
      node=ic2de(k)
      if(itype(node).eq.0) go to 102
  110 if(itype(node).gt.0) ict=ict+1
      if(ict.eq.0) go to 102
      ncol=3
  112 n2del=n2del+1
      do 114 k=1,8
      node=ic2de(k)
      if(node.eq.0) id2de(k)=0
      if(node.ne.0) id2de(k)=id(node)
  114 continue
      if(ncol.eq.24) write(6,1001) n2del,ic2de
 1001 format(5x,i5,3x,'dam',2x,8i5)
      if(ncol.eq.3) write(6,1002) n2del,ic2de
 1002 format(5x,i5,3x,'rfs',2x,8i5)
      area=0.d0
      call zero(xx,3,8)
      call zero(be,8,24)
      do 200 k=1,8
      node=ic2de(k)
      if(node.eq.0) go to 200
      do 205 l=1,3
  205 xx(l,k)=xyz(l,node)
  200 continue
      do 300 lr=1,2
      e1=gloc(lr)
      do 300 ls=1,2
      e2=gloc(ls)
      call shap2d(ic2de)
      area=area+det
      do 310 k=1,8
      do 310 l=1,8
      mm=3*(l-1)
      do 310 m=1,3
      mm=mm+1
      hhb=h(k)*h(l)*bn(m)
      if(ncol.eq.24) be(k,mm)=be(k,mm)+hhb*det
      if(ncol.eq.3) be(k,m)=be(k,m)+hhb*det
  310 continue
  300 continue
      write(ielmt2d) ncol,ic2de,id2de,((be(k,l),k=1,8),l=1,ncol)
      write(6,1003) area
 1003 format(18x,'area =',e12.5)
  102 continue
  100 continue
      return
      end
      subroutine nodes(xyz,itype,id,nlist,nnp,watl,iup,neq,nes,ner,net)
      implicit real*8 (a-h,o-z)
      dimension xyz(3,nnp),itype(nnp),id(nnp),nlist(net)
c
      do 100 i=1,nnp
  100 read(5,*) n,(xyz(j,n),j=1,3)
      vmax=-9.d20
      vmin=9.d20
      do 200 i=1,nnp
      coor=xyz(iup,i)
      if(coor.gt.vmax) vmax=coor
  200 if(coor.lt.vmin) vmin=coor
      diff=0.001d0*(vmax-vmin)
      watlmax=watl+diff
      watlmin=watl-diff
      call izero(itype,nnp,1)
      write(6,1000)
 1000 format(//,1x,'O3:  Boundary node type data',//,8x,'n1',6x,
     1 'n2',5x,'inc',5x,'itype')
      do 400 i=1,1000
      read(5,*) is,ie,inc,ival
      if(is.eq.0) go to 499
      write(6,1001) is,ie,inc,ival
 1001 format(5x,i5,3x,i5,3x,i5,5x,i5)
      if(((ie.ne.0).and.(inc.eq.0)).or.(iabs(ival).ne.1)) go to 498
      if(ie.eq.0) ie=is
      if(inc.eq.0) inc=1
      do 400 k=is,ie,inc
      if(itype(k).ne.0) go to 498
      itype(k)=ival
      coor=xyz(iup,k)
      if(coor.gt.watlmax) itype(k)=0
      if((coor.le.watlmax).and.(coor.ge.watlmin)) itype(k)=ival*2
  400 continue
  498 write(6,1002)
 1002 format(//,1x,'boundary condition input error')
      stop
c
  499 write(6,1003)
 1003 format(//,1x,'O4:  Nodal coordinates and types',//,5x,' node',
     1 9x,'xcoor',9x,'ycoor',9x,'zcoor',2x,'type')
      write(6,1004) (n,(xyz(i,n),i=1,3),itype(n),n=1,nnp)
 1004 format (5x,i5,2x,e12.5,2x,e12.5,2x,e12.5,2x,i4)
      neq=0
      call izero(id,nnp,1)
      do 700 i=1,nnp
      coor=xyz(iup,i)
      if(coor.ge.watlmin) go to 700
      if(itype(i).ge.0) neq=neq+1
      if(itype(i).ge.0) id(i)=neq
  700 continue
      ner=neq
      do 710 i=1,nnp
      if(itype(i).eq.-1) neq=neq+1
      if(itype(i).eq.-1) id(i)=neq
  710 continue
      nes=neq-ner
      net=0
      do 720 i=1,nnp
      if(itype(i).eq.-2) net=net+1
      if(itype(i).eq.-2) id(i)=-net
  720 continue
      net=net+nes
      write(6,1004) (n,(xyz(i,n),i=1,3),id(n),n=1,nnp)
c
      n=0
      do 900 i=1,nnp
      if(itype(i).eq.-2) n=n+1
  900 if(itype(i).eq.-2) nlist(n)=i
      do 910 i=1,nnp
      if(itype(i).eq.-1) n=n+1
  910 if(itype(i).eq.-1) nlist(n)=i
      return
      end

      subroutine scondn(lc,f,xk,neq,ner,nwk,nea)
      implicit real*8 (a-h,o-z)
      dimension lc(neq),f(neq,3),xk(nwk)
      call subsol(xk,f,lc,neq,ner,3,nwk,1)
      call subsol(xk,f,lc,neq,ner,3,nwk,2)
      ner1=ner+1
      nea=0
      do 100 j=ner1,neq
      me=lc(j)-lc(j-1)
      ik=j-me
      nea=nea+j-max0(ik,ner)
  100 continue
      return
      end

      subroutine shap2d(ice)
      implicit real*8 (a-h,o-z)
      common /shape/ r,s,det,h(8),g(2,8),xx(3,8),aj(2,3),bn(3),
     1 dum(162)
      dimension ice(8),mn(8)
c
      do 10 i=1,8
      if(ice(i).eq.0) mn(i)=0
   10 if(ice(i).ne.0) mn(i)=1
      rp=0.5d0*(1.0d0+r)
      rn=0.5d0*(1.0d0-r)
      sp=0.5d0*(1.0d0+s)
      sn=0.5d0*(1.0d0-s)
      rr=1.0d0-r*r
      ss=1.0d0-s*s
      h(5)=rr*sp*mn(5)
      h(6)=rn*ss*mn(6)
      h(7)=rr*sn*mn(7)
      h(8)=rp*ss*mn(8)
      h(1)=rp*sp-0.5d0*(h(5)+h(8))
      h(2)=rn*sp-0.5d0*(h(5)+h(6))
      h(3)=rn*sn-0.5d0*(h(6)+h(7))
      h(4)=rp*sn-0.5d0*(h(7)+h(8))
c
      drp=0.5d0
      drn=-0.5d0
      dsp=0.5d0
      dsn=-0.5d0
      drr=-2.0d0*r
      dss=-2.0d0*s
      g(1,5)=drr*sp*mn(5)
      g(2,5)=rr*dsp*mn(5)
      g(1,6)=drn*ss*mn(6)
      g(2,6)=rn*dss*mn(6)
      g(1,7)=drr*sn*mn(7)
      g(2,7)=rr*dsn*mn(7)
      g(1,8)=drp*ss*mn(8)
      g(2,8)=rp*dss*mn(8)
      g(1,1)=drp*sp-0.5d0*(g(1,5)+g(1,8))
      g(2,1)=rp*dsp-0.5d0*(g(2,5)+g(2,8))
      g(1,2)=drn*sp-0.5d0*(g(1,5)+g(1,6))
      g(2,2)=rn*dsp-0.5d0*(g(2,5)+g(2,6))
      g(1,3)=drn*sn-0.5d0*(g(1,6)+g(1,7))
      g(2,3)=rn*dsn-0.5d0*(g(2,6)+g(2,7))
      g(1,4)=drp*sn-0.5d0*(g(1,7)+g(1,8))
      g(2,4)=rp*dsn-0.5d0*(g(2,7)+g(2,8))
c
      do 80 i=1,2
      do 80 j=1,3
      c=0.0d0
      do 90 k=1,8
      c=c+g(i,k)*xx(j,k)
   90 continue
   80 aj(i,j)=c
      bn(1)=aj(1,2)*aj(2,3)-aj(2,2)*aj(1,3)
      bn(2)=aj(1,3)*aj(2,1)-aj(2,3)*aj(1,1)
      bn(3)=aj(1,1)*aj(2,2)-aj(2,1)*aj(1,2)
      det=0.0d0
      do 200 i=1,3
  200 det=det+bn(i)*bn(i)
      det=sqrt(det)
      if(det.le.1.0d-10) go to 300
      do 400 i=1,3
  400 bn(i)=bn(i)/det
c
      return
  300 write(6,1000) det
 1000 format(/,1x,'bad jacobian; det =',e12.5)
      stop
      end

      subroutine shap3d(ice)
      implicit real*8 (a-h,o-z)
      common /shape/ r,s,t,det,h(20),g(3,20),xx(3,20),aj(3,3),bj(3,3),
     1 d(3,20)
      dimension ice(20),mn(20),iperm(3)
      data iperm /2,3,1/
c
      do 10 i=1,20
      if(ice(i).eq.0) mn(i)=0
   10 if(ice(i).ne.0) mn(i)=1
      rp=0.5d0*(1.0d0+r)
      rn=0.5d0*(1.0d0-r)
      sp=0.5d0*(1.0d0+s)
      sn=0.5d0*(1.0d0-s)
      tp=0.5d0*(1.0d0+t)
      tn=0.5d0*(1.0d0-t)
      rr=1.0d0-r*r
      ss=1.0d0-s*s
      tt=1.0d0-t*t
      h(9)=rr*sp*tp*mn(9)
      h(10)=rn*ss*tp*mn(10)
      h(11)=rr*sn*tp*mn(11)
      h(12)=rp*ss*tp*mn(12)
      h(13)=rr*sp*tn*mn(13)
      h(14)=rn*ss*tn*mn(14)
      h(15)=rr*sn*tn*mn(15)
      h(16)=rp*ss*tn*mn(16)
      h(17)=rp*sp*tt*mn(17)
      h(18)=rn*sp*tt*mn(18)
      h(19)=rn*sn*tt*mn(19)
      h(20)=rp*sn*tt*mn(20)
      h(1)=rp*sp*tp-0.5d0*(h(9)+h(12)+h(17))
      h(2)=rn*sp*tp-0.5d0*(h(9)+h(10)+h(18))
      h(3)=rn*sn*tp-0.5d0*(h(10)+h(11)+h(19))
      h(4)=rp*sn*tp-0.5d0*(h(11)+h(12)+h(20))
      h(5)=rp*sp*tn-0.5d0*(h(13)+h(16)+h(17))
      h(6)=rn*sp*tn-0.5d0*(h(13)+h(14)+h(18))
      h(7)=rn*sn*tn-0.5d0*(h(14)+h(15)+h(19))
      h(8)=rp*sn*tn-0.5d0*(h(15)+h(16)+h(20))
c
      drp=0.5d0
      drn=-0.5d0
      dsp=0.5d0
      dsn=-0.5d0
      dtp=0.5d0
      dtn=-0.5d0
      drr=-2.0d0*r
      dss=-2.0d0*s
      dtt=-2.0d0*t
      g(1,9)=drr*sp*tp*mn(9)
      g(2,9)=rr*dsp*tp*mn(9)
      g(3,9)=rr*sp*dtp*mn(9)
      g(1,10)=drn*ss*tp*mn(10)
      g(2,10)=rn*dss*tp*mn(10)
      g(3,10)=rn*ss*dtp*mn(10)
      g(1,11)=drr*sn*tp*mn(11)
      g(2,11)=rr*dsn*tp*mn(11)
      g(3,11)=rr*sn*dtp*mn(11)
      g(1,12)=drp*ss*tp*mn(12)
      g(2,12)=rp*dss*tp*mn(12)
      g(3,12)=rp*ss*dtp*mn(12)
      g(1,13)=drr*sp*tn*mn(13)
      g(2,13)=rr*dsp*tn*mn(13)
      g(3,13)=rr*sp*dtn*mn(13)
      g(1,14)=drn*ss*tn*mn(14)
      g(2,14)=rn*dss*tn*mn(14)
      g(3,14)=rn*ss*dtn*mn(14)
      g(1,15)=drr*sn*tn*mn(15)
      g(2,15)=rr*dsn*tn*mn(15)
      g(3,15)=rr*sn*dtn*mn(15)
      g(1,16)=drp*ss*tn*mn(16)
      g(2,16)=rp*dss*tn*mn(16)
      g(3,16)=rp*ss*dtn*mn(16)
      g(1,17)=drp*sp*tt*mn(17)
      g(2,17)=rp*dsp*tt*mn(17)
      g(3,17)=rp*sp*dtt*mn(17)
      g(1,18)=drn*sp*tt*mn(18)
      g(2,18)=rn*dsp*tt*mn(18)
      g(3,18)=rn*sp*dtt*mn(18)
      g(1,19)=drn*sn*tt*mn(19)
      g(2,19)=rn*dsn*tt*mn(19)
      g(3,19)=rn*sn*dtt*mn(19)
      g(1,20)=drp*sn*tt*mn(20)
      g(2,20)=rp*dsn*tt*mn(20)
      g(3,20)=rp*sn*dtt*mn(20)
      g(1,1)=drp*sp*tp-0.5d0*(g(1,9)+g(1,12)+g(1,17))
      g(2,1)=rp*dsp*tp-0.5d0*(g(2,9)+g(2,12)+g(2,17))
      g(3,1)=rp*sp*dtp-0.5d0*(g(3,9)+g(3,12)+g(3,17))
      g(1,2)=drn*sp*tp-0.5d0*(g(1,9)+g(1,10)+g(1,18))
      g(2,2)=rn*dsp*tp-0.5d0*(g(2,9)+g(2,10)+g(2,18))
      g(3,2)=rn*sp*dtp-0.5d0*(g(3,9)+g(3,10)+g(3,18))
      g(1,3)=drn*sn*tp-0.5d0*(g(1,10)+g(1,11)+g(1,19))
      g(2,3)=rn*dsn*tp-0.5d0*(g(2,10)+g(2,11)+g(2,19))
      g(3,3)=rn*sn*dtp-0.5d0*(g(3,10)+g(3,11)+g(3,19))
      g(1,4)=drp*sn*tp-0.5d0*(g(1,11)+g(1,12)+g(1,20))
      g(2,4)=rp*dsn*tp-0.5d0*(g(2,11)+g(2,12)+g(2,20))
      g(3,4)=rp*sn*dtp-0.5d0*(g(3,11)+g(3,12)+g(3,20))
      g(1,5)=drp*sp*tn-0.5d0*(g(1,13)+g(1,16)+g(1,17))
      g(2,5)=rp*dsp*tn-0.5d0*(g(2,13)+g(2,16)+g(2,17))
      g(3,5)=rp*sp*dtn-0.5d0*(g(3,13)+g(3,16)+g(3,17))
      g(1,6)=drn*sp*tn-0.5d0*(g(1,13)+g(1,14)+g(1,18))
      g(2,6)=rn*dsp*tn-0.5d0*(g(2,13)+g(2,14)+g(2,18))
      g(3,6)=rn*sp*dtn-0.5d0*(g(3,13)+g(3,14)+g(3,18))
      g(1,7)=drn*sn*tn-0.5d0*(g(1,14)+g(1,15)+g(1,19))
      g(2,7)=rn*dsn*tn-0.5d0*(g(2,14)+g(2,15)+g(2,19))
      g(3,7)=rn*sn*dtn-0.5d0*(g(3,14)+g(3,15)+g(3,19))
      g(1,8)=drp*sn*tn-0.5d0*(g(1,15)+g(1,16)+g(1,20))
      g(2,8)=rp*dsn*tn-0.5d0*(g(2,15)+g(2,16)+g(2,20))
      g(3,8)=rp*sn*dtn-0.5d0*(g(3,15)+g(3,16)+g(3,20))
c
      do 300 i=1,3
      do 300 j=1,3
      c=0.0d0
      do 350 k=1,20
      c=c+g(i,k)*xx(j,k)
  350 continue
  300 aj(i,j)=c
      do 400 i=1,3
      j=iperm(i)
      k=iperm(j)
      bj(i,i)=aj(j,j)*aj(k,k)-aj(k,j)*aj(j,k)
      bj(i,j)=aj(i,k)*aj(k,j)-aj(i,j)*aj(k,k)
  400 bj(j,i)=aj(j,k)*aj(k,i)-aj(j,i)*aj(k,k)
      det=0.0d0
      do 410 i=1,3
  410 det=det+aj(1,i)*bj(i,1)
      if(det.le.1.0d-10) go to 700
      do 500 i=1,3
      do 550 j=1,20
      c=0.0d0
      do 600 k=1,3
  600 c=c+bj(i,k)*g(k,j)
      d(i,j)=c/det
  550 continue
  500 continue
c
      return
  700 write(6,1000) det
 1000 format(/,1x,'bad jacobian; det =',e12.5)
      stop
      end

      subroutine solutn(nlist,mc,b,fc,xkc,w,nes,net,net3,nea)
      implicit real*8 (a-h,o-z)
      dimension nlist(net),mc(nes),b(nes,net3),fc(nes,3),xkc(nea),p(6)
c
      call subsol(xkc,b,mc,nes,nes,net3,nea,1)
      call subsol(xkc,b,mc,nes,nes,net3,nea,2)
      call subsol(xkc,b,mc,nes,nes,net3,nea,3)
      call subsol(xkc,fc,mc,nes,nes,3,nea,2)
      call subsol(xkc,fc,mc,nes,nes,3,nea,3)
      write(6,1000)
 1000 format(//,1x,'O7:  Pressures due to 1g accel of dam and',
     1 ' dam plus res floor and sides in x, y and z dirs',//,1x,' node',
     1 2x,'dam in x dir',2x,'dam in y dir',2x,'dam in z dir',
     1 2x,'d+r in x dir',2x,'d+r in y dir',2x,'d+r in z dir')
      do 200 i=1,nes
      call zero(p,6,1)
      do 210 j=1,net3,3
      p(1)=p(1)-b(i,j)*w
      p(2)=p(2)-b(i,j+1)*w
  210 p(3)=p(3)-b(i,j+2)*w
      p(4)=p(1)-fc(i,1)*w
      p(5)=p(2)-fc(i,2)*w
      p(6)=p(3)-fc(i,3)*w
      ia=net-nes+i
      write(6,1001) nlist(ia),p
 1001 format(1x,i5,6(2x,e12.5))
  200 continue
      return
      end

      subroutine subff(lc,mc,f,fc,xk,xkc,neq,nes,nwk,nea)
      implicit real*8 (a-h,o-z)
      dimension lc(neq),mc(nes),f(neq,3),fc(nes,3),xk(nwk),xkc(nea)
c
      kj=0
      nn=0
      nep=neq-nes+1
      do 100 j=nep,neq
      me=lc(j)-lc(j-1)
      ik=j-me+1
      is=max0(ik,nep)
      kj=kj+1
      do 200 i=is,j
      ln=lc(j)-j+i
      nn=nn+1
      if(i.eq.j) mc(kj)=nn
      xkc(nn)=xk(ln)
  200 continue
  100 continue
      if(nn.eq.nea) go to 399
      write(6,1000)
 1000 format(//,1x,'error in subroutine subff')
      stop
  399 do 400 j=1,3
      do 400 i=1,nes
  400 fc(i,j)=f(i+neq-nes,j)
      return
      end

      subroutine subsol(a,b,lc,neq,leq,nl,nwk,kk)
      implicit real*8 (a-h,o-z)
      dimension a(nwk),b(neq,nl),lc(neq)
c
      go to (50,550,890),kk
c
   50 if(neq.eq.1) return
      do 500 j=2,neq
      jh=lc(j)-lc(j-1)
      if(jh.eq.1) go to 500
      k=j-jh+1
      i=k
  100 if(i-1) 120,120,140
  120 lcc=lc(i)
      go to 160
  140 lcc=lc(i)-lc(i-1)
  160 nt=min0(i-j+jh,lcc)-1
      ns=lc(i)-nt
      ne=lc(i)-1
      ij=lc(j)-j+i
      ic=ij-lc(i)
      s=0.0d0
      if(i.eq.j) go to 400
      if(i.gt.leq) nt=nt+leq-i+1
      if(nt) 300,300,150
  150 call dotp(a(ns),a(ns+ic),s,nt)
      a(ij)=a(ij)-s
  300 i=i+1
      go to 100
  400 if(i.gt.leq) ne=ne+leq-i+1
      if(ne.lt.ns) go to 460
      do 450 n=ns,ne
      nd=lc(k)
      k=k+1
      t=a(n)
      a(n)=a(n)/a(nd)
  450 s=s+a(n)*t
  460 a(ij)=a(ij)-s
c
      if(s.eq.0.0d0) go to 500
      if(a(ij).eq.0.0d0) go to 490
      ff=dabs(s/a(ij))
      nf=dlog10(ff)
      if(nf.lt.9) go to 500
  490 a(1)=0.0d0
      write(6,2000)
 2000 format(//,1x,'matrix may be singular')
      stop
  500 continue
      return
c
  550 do 860 l=1,nl
      do 700 j=2,neq
      jh=lc(j)-lc(j-1)-1
      ns=lc(j)-jh
      k=j-jh
      if(j.gt.leq) jh=jh+leq-j+1
      if(jh.le.0) go to 700
      call dotp(a(ns),b(k,l),s,jh)
      b(j,l)=b(j,l)-s
  700 continue
  800 do 850 i=1,leq
      k=lc(i)
  850 b(i,l)=b(i,l)/a(k)
  860 continue
      return
c
  890 do 960 l=1,nl
      j=neq
  900 if(j.eq.1) go to 960
      k=j-lc(j)+lc(j-1)+1
      ns=lc(j-1)+1
      jh=lc(j)-lc(j-1)-1
      if(j.gt.leq) jh=jh+leq-j+1
      if(jh.le.0) go to 950
      s=-b(j,l)
      call addv(b(k,l),a(ns),s,jh)
  950 j=j-1
      go to 900
  960 continue
      return
      end

      subroutine zero(s,n,m)
      implicit real*8 (a-h,o-z)
      dimension s(n,m)
      do 10 j=1,m
      do 10 i=1,n
   10 s(i,j)=0.0d0
      return
      end

      subroutine izero(is,n,m)
      implicit real*8 (a-h,o-z)
      dimension is(n,m)
      do 10 j=1,m
      do 10 i=1,n
   10 is(i,j)=0
      return
      end
      subroutine dotp(a,b,s,n)
      implicit real*8 (a-h,o-z)
      dimension a(n),b(n)
      s=0.0d0
      do 100 i=1,n
  100 s=s+a(i)*b(i)
      return
      end

      subroutine addv(a,b,s,n)
      implicit real*8 (a-h,o-z)
      dimension a(n),b(n)
      do 100 i=1,n
  100 a(i)=a(i)+b(i)*s
      return
      end

      subroutine error(n)
      implicit real*8 (a-h,o-z)
      write(6,1000) n
 1000 format(//,1x,'increase storage to',i8)
      stop
      end
```