CHAPTER IV

FINITE ELEMENT MESH GENERATION

The first step in finding an approximate solution using finite element method is dividing the physical region or domain Ω into subdomain or finite elements i.e. to construct a finite element mesh representing Ω

In one-dimensional problems, we partition an interval into line elements connected at nodal points at their ends. For two dimensional problems we construct finite element mesh by a collection of triangular or rectangular elements. If the boundary $d\Omega$ is curved, there is some discretization error since the finite element mesh will not perfectly coincide with the given domain Ω . However, as the mesh is refined, this discretization error can be made almost equal to zero.

We can generate a grid by finding a correspondence between points (x,y) in the irregular physical domain and points (ϵ , Π) in the regular computational domain

A conceptual approach to grid generation is to fix the values of ϵ and Π on the physical boundaries first and then locating the interior points by determining the intersection of co-ordinate lines of opposite families drawn between corresponding boundary points.

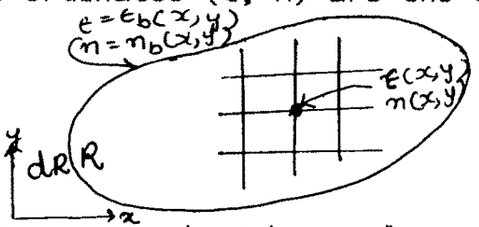
In a sense, the problem of grid generation can be posed as a boundary value problem :-

Given

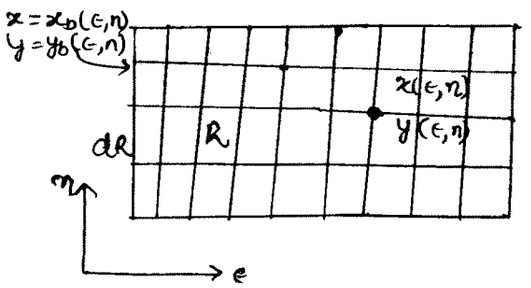
$$\epsilon = \epsilon_b (x, y) \text{ and } \Pi = \Pi_b(x, y)$$

on the boundary dR generate $\epsilon = \epsilon (x, y)$ and $\Pi = \Pi(x, y)$ in the region R bounded by dR . The physical co-ordinates (x, y) ,

typically cartesian are the independent variables and the generalised co-ordinates (ϵ, η) are the dependent variables.



Grid generation as a boundary value problem in the physical domain.



Grid generation as a boundary value problem in the computational domain.

The grid can be generated with less computational effort by working in the computational domain. Thus fixing the location of the points on the boundary gives $x = x_b(\epsilon, \eta)$ and $y = y_b(\epsilon, \eta)$. The generation of the grid in the interior is expressed as the following boundary value problem : given $x = x_b(\epsilon, \eta)$ and $y = y_b(\epsilon, \eta)$ on dR , generate $x = x(\epsilon, \eta)$ and $y = y(\epsilon, \eta)$ in the region bounded by dR.

Since the interior points in the computational domain form a regular grid and the boundaries coincide with co-ordinate lines, the determination of $x(\epsilon, \eta)$ and $y(\epsilon, \eta)$ is easier than working in the irregular physical domain, particularly if a partial differential equation is to be solved to generate the solution, $x(\epsilon, \eta)$ and $y(\epsilon, \eta)$.

In defining the relationship between points in the physical and computational domains i.e. $x = x(\epsilon, \eta)$ and $y = y(\epsilon, \eta)$,

it is necessary that there is a one to one correspondence. It would be unacceptable for a single point in the physical domain to map into two points in the computational domain.

Once the mapping $x = x(\epsilon, \eta)$ and $y = y(\epsilon, \eta)$ has been established, the requirement of a one-to-one mapping can be determined by evaluating the determinant of the transformation Jacobian, $|J|$. For the mapping to be one-to-one, $|J|$ must be finite and non-zero. Depending on how the grid has been generated, $|J|$ can be evaluated at each grid point, analytically or numerically to check for a one-to-one mapping. During development, computer plotting of the grid will quickly locate any points where the mapping is double valued.

An original method to generate a finite element mesh in a planar domain of arbitrary shape.

The characteristics of this generator are :-

1) The mesh density is imposed by the user, it can vary as desired on the contour of the domain and also inside the domain.

2) The number of elements generated can be imposed approximately, what allows to control the number of degrees of freedom of the discretized field.

SECTION (II) : AUTOMATIC MESH GENERATION

Automation of finite element mesh generation holds great benefits for mechanical product development and analysis. In addition to freeing engineers from mundane tasks, automation of mesh generation reduces product cycle design and eliminates human-related errors. Most of the existing mesh generation

methods are either semi-automatic or require specific topological information. A fully automatic mesh generation method can produce quadrilateral or triangular elements. The input includes the region's boundary curves the element size and the mesh density. We begin by first constructing a coarse grid i.e. the region is meshed into subregions. Each subregion is then meshed into smaller subregions. This procedure is repeated a number of times until the final mesh is produced.

The unstructured mesh needed to initiate the adaptive mesh procedure can come from any mesh generator.

The most expedient method was found to use the structured mesh divided into triangles. This mesh is then refined in a control loop using a function which provides the size and the elongation of the desired mesh.

The refinement of the grid is obtained through triangle subdivision. A triangle to the refined is branched into two triangles by cutting it on its longest side. This process is done on all triangles requiring this option and the reconnection of unmatched sides is performed last. Particular attention must be taken for curved boundaries because when a side representing a curved boundary is cut, the new nodes inserted on that side must be relocated on the boundary to remain consistent with the geometric representation.

SECTION (III): **A Boundary Value Problem in Partial Differential Equations**

We consider the partial differential equation

$$\frac{\partial}{\partial x}\left(p\frac{\partial u}{\partial x}\right)+\frac{\partial}{\partial y}\left(p\frac{\partial u}{\partial y}\right)+r=0 \quad \text{in } \Omega \qquad ..(4.1)$$

with the boundary conditions

$$u = g(x,y) \quad \text{on } d\Omega \qquad ....(4.2)$$

where p and r may be constants or fuctions of x and y only. The variational formulation of equation (4.1) requires the functional

$$J = \frac{1}{2}\int\int_{\Omega}\left[p\left(\frac{\partial u}{\partial x}\right)^2 + p\left(\frac{\partial u}{\partial y}\right)^2 - 2ur\right]dx\,dy \qquad ....(4.3)$$

where the boundary conditions (4.2) are to be satisfied. We divide the domain Ω into finite elements. The approximate solution $w(x,y)$ for the whole domain Ω may be written as

$$w(x,y) = \sum_{e=1}^{M} N^e \phi^e = \sum_{i=1}^{N} N_i \phi_i = \mathbf{N}\boldsymbol{\phi} \qquad ....(4.4)$$

where M is the number of elements with N nodes in Ω,

$\mathbf{N} = [N_1 \;\; N_2 \;\; N_3 \;\; ..... \;\; N_N]$ and $\boldsymbol{\phi} = [\phi_1 \;\; \phi_2 \;\; \phi_3 \;\; ..... \;\; \phi_N]^T$,

N_i satisfy hte conditions

$$N_i(x,y) = N_i^e(x,y), \quad \text{if } (x,y) \in \Omega^e$$
$$= 0 \quad , \text{ otherwise} \qquad .....(4.5)$$

and ϕ^e are the nodal values associated with element e.

Substituting equation (4.4) in (4.3), we get

$$J = \frac{1}{2} \iint_{\Omega} \left\{ p \left(\sum_{e=1}^{M} \frac{\partial N^e}{\partial x} \phi^e \right)^2 + p \left(\sum_{e=1}^{M} \frac{\partial N^e}{\partial y} \phi^e \right)^2 - 2r \sum_{e=1}^{M} N^e \phi^e \right\} dx\, dy \quad \ldots\ldots(4.6)$$

Using equation (4.5) we assume that (4.6) can be written in the form

$$J = \sum_{e=1}^{M} J^e \quad \ldots\ldots(4.7)$$

where

$$J^e = \frac{1}{2} \iint_{\Omega^e} \left\{ p \left(\frac{\partial N^e}{\partial x} \phi^e \right)^2 + p \left(\frac{\partial N^e}{\partial y} \phi^e \right)^2 - 2r\, N^e \phi^e \right\} dx\, dy \quad \ldots\ldots(4.8)$$

is the contribution of the element Ω^e to the functional J.

The conditions for the minimization with respect to the nodal values ϕ_i , i = 1 to N gives the following system of equations

$$\frac{\partial J}{\partial \phi_i} = \sum_{e=1}^{M} \frac{\partial J^e}{\partial \phi_i} = 0 \quad , \quad i = 1 \text{ to } N \quad \ldots\ldots(4.9)$$

The equation $\dfrac{\partial J^e}{\partial \phi_i} = 0$ is called the element equation.

Differentiating equation (4.8) with respect to ϕ^e , we get

$$\frac{\partial J^e}{\partial \phi_i} = \frac{1}{2} \iint_{\Omega^e} \left\{ p \left(\frac{\partial N^e}{\partial x}^T \frac{\partial N^e}{\partial x} + \frac{\partial N^e}{\partial y}^T \frac{\partial N^e}{\partial y} \right) \phi^e - r\, N^e \right\} dx\, dy \quad \ldots\ldots(4.10)$$

Thus the element equation becomes

$$A^e \phi^e - b = 0 \qquad \ldots.(4.11)$$

where

$$A^e = \iint_{\Omega^e} p \left(\frac{\partial N^e}{\partial x}^T \frac{\partial N^e}{\partial x} + \frac{\partial N^e}{\partial y}^T \frac{\partial N^e}{\partial y} \right) dx\, dy$$

$$b^e = \iint_{\Omega^e} r N^{eT} \, dx\, dy.$$

Considering the linear three node triangle with nodes i,j,k the linear piecewise approximation can be written as,

$$u^e = N_i u_i + N_j u_j + N_k u_k = N^e \phi^e$$

where $N^e = [\, N_i \quad N_j \quad N_k \,]$ and $\phi^e = [\, u_i \quad u_j \quad u_k \,]$

$$N_i = \frac{1}{2\Delta^e} (a_i + b_i x + c_i y)$$

$$N_j = \frac{1}{2\Delta^e} (a_j + b_j x + c_j y)$$

$$N_k = \frac{1}{2\Delta^e} (a_k + b_k x + c_k y)$$

$$a_i = x_j y_k - x_k y_j \,, \quad a_j = x_k y_i - x_i y_k \,, \quad a_k = x_i y_j - x_j y_i$$

$$b_i = y_j - y_k \,, \quad b_j = y_k - y_i \,, \quad b_k = y_i - y_j$$

$$c_i = x_k - x_j \,, \quad c_j = x_i - x_k \,, \quad c_k = x_j - x_i$$

where

$$\Delta^e = \frac{1}{2}\begin{vmatrix} 1 & x_i & y_i \\ 1 & x_j & y_j \\ 1 & x_k & y_k \end{vmatrix}$$

$$= 2 \text{ (area of element e)}$$

Substituting these equations in equation(4.11) we get

$$A^e \phi^e - b^e = 0$$

where

$$A^e = \frac{p^e}{4\Delta^e}\begin{bmatrix} b_i^2+c_i^2 & b_ib_j + c_ic_j & b_ib_j + c_ic_j \\ b_ib_j + c_ic_j & b_j^2 + c_j^2 & b_jb_k + c_jc_k \\ b_ib_k + c_ic_k & b_jb_k + c_jc_k & b_k^2 + c_k^2 \end{bmatrix}$$

$$b^e = \frac{r^e \Delta^e}{3}\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}, \qquad \phi^e = \begin{bmatrix} u_i \\ u_j \\ u_k \end{bmatrix}$$

PROGRAM LISTING

```
PROGRAM FEM(INPUT,OUTPUT);

{THIS PROGRAM USES AUTOMATIC GENERATION OF MESH POINTS FOR FINITE
 ELEMENT METHOD USING TRIANGULAR ELEMENTS}

 {$M 65520,16384,65360}
 CONST
      NX = 8;  { NUMBER OF NODES ON X AXIS }
      NY = 8; { NUMBER OF NODES ON Y AXIS }
      NN = NX*NY;
 TYPE
       ELEMENTTYPE = RECORD
        ELEMENTNUMBER : INTEGER;
        X_COORDINATES :ARRAY[1..3] OF REAL;
        Y_COORDINATES:ARRAY[1..3] OF REAL
        END;
       MATRIXTYPE = ARRAY [1..NN,1..NN] OF REAL;
       RTYPE=ARRAY[1..NN] OF REAL;
       VECTORTYPE = ARRAY[1..3] OF INTEGER;
 VAR
      MAX,I,J,N,L:INTEGER;
      NELEM,XTOT,YTOT,XNODE,YNODE,NNODE:INTEGER;
      SOL_VECT,XCOORDINATE,YCOORDINATE,F_VECTOR,GLOBAL_VECT:RTYPE;
      X_INITIAL,Y_INITIAL,X_FINAL,Y_FINAL, Q, q1, K:REAL;
      ELEMENT:ELEMENTTYPE;
      GLOBAL_MAT,K_MATRIX:MATRIXTYPE;
```

```
PROCEDURE RCOORDINATES(VAR X1,Y1,N1,N2:INTEGER;
                       VAR XINI,XF,YINI,YF:REAL;
                       VAR XCO,YCO:RTYPE);

{THIS PROCEDURE GENERATES THE X AND Y COORDINATES
 FOR THE ENTIRE MESH}

VAR
    XX,YY:REAL;

BEGIN
     XX:=(XF-XINI)/X1;
     YY:=(YF - YINI)/Y1;
     XCO[1]:=XINI;
     YCO[1]:=YINI;
     FOR L:= 2 TO N1 DO
      BEGIN
       XCO[L]:=XCO[L-1] + XX;
      END;
     FOR I:= 2 TO N2 DO
      BEGIN
       YCO[I]:=YCO[I-1] + YY;
      END;
END;
```

```
PROCEDURE ELEMENT_COORDINATES(VAR NN,X,Y:INTEGER;E:ELEMENTTYPE;
                              VAR XCO,YCO:RTYPE;VAR G,K:MATRIXTYPE;
                              VAR GF,F:RTYPE; R:REAL);

{THIS PROCEDURE CALCULATES THE NODE NUMBERS OF EACH ELEMENT
 AND THEIR COORDINATES}

 VAR
     NODENUMBER,I,J,L,M,P,Q:INTEGER;
     NODE:VECTORTYPE;

BEGIN
WITH E DO
 BEGIN
  NODENUMBER:=1;
  E.ELEMENTNUMBER:=0;
  FOR J:= 1 TO Y DO
   BEGIN
    FOR I:= 1 TO X DO
     BEGIN
      E.ELEMENTNUMBER:=E.ELEMENTNUMBER + 1;
      NODE[1]:=NODENUMBER;
      E.X_COORDINATES[1]:=XCO[I];
      E.Y_COORDINATES[1]:=YCO[J];
      NODE[2]:=NODENUMBER + 1;
      E.X_COORDINATES[2]:=XCO[I+1];
      E.Y_COORDINATES[2]:=YCO[J];
      NODE[3]:= X + NODE[2] + 1;
      E.X_COORDINATES[3]:=XCO[I+1];
      E.Y_COORDINATES[3]:=YCO[J+1];
      NODENUMBER:=NODENUMBER + 1 ;
      KMATRIX(NN,E,R,NODE,F,GF,G,K);
    END;
    NODENUMBER:=NODENUMBER + 1 ;
 END;

    NODENUMBER :=1;
    FOR J:= 1 TO Y DO
    BEGIN
        FOR I:= 1 TO X DO
         BEGIN
          E.ELEMENTNUMBER:=E.ELEMENTNUMBER + 1;
          NODE[1]:=NODENUMBER;
          E.X_COORDINATES[1]:=XCO[I];
          E.Y_COORDINATES[1]:=YCO[J];
          NODE[2]:=NODENUMBER + X + 1;
          E.X_COORDINATES[2]:=XCO[I];
          E.Y_COORDINATES[2]:=YCO[J+1];
          NODE[3]:=NODE[2] + 1;
          E.X_COORDINATES[3]:=XCO[I+1];
          E.Y_COORDINATES[3]:=YCO[J+1];
          NODENUMBER:=NODENUMBER + 1 ;
          KMATRIX(NN,E,R,NODE,F,GF,G,K);
         END;
         NODENUMBER:=NODENUMBER + 1 ;
      END;
    END;
 END;   END;   END;          END;
```

```
PROCEDURE KMATRIX(NN:INTEGER; VAR E:ELEMENTTYPE;R:REAL;
                  NODE : VECTORTYPE; VAR F,GF:RTYPE;
                  VAR G,K:MATRIXTYPE);

{THIS PROCEDURE COMPUTES THE GLOBAL MATRIX FOR ALL THE ELEMENTS}

VAR
    X,Y:ARRAY[1..3] OF REAL;
    A:MATRIXTYPE;
    DELTA,ALPHA:REAL;
    BETA,GAMMA:ARRAY [1..3] OF REAL;
    U,M,L,P,Q,N1,N2,N3:INTEGER;
    FV:ARRAY[1..3] OF REAL;

    BEGIN
      FOR M:=1 TO 3 DO
       BEGIN
        X[M]:=E.X_COORDINATES[M];
        Y[M]:=E.Y_COORDINATES[M];
       END;

DELTA:=X[2]*Y[3]-X[3] *Y[2]+ X[1]*Y[2]-X[1]*Y[3]+X[3]*Y[1]-X[2]*Y[1];
ALPHA:=(X[2]*Y[3] -X[3]*Y[2])/DELTA;
BETA[1]:=(Y[2]-Y[3])/DELTA;
BETA[2]:=(Y[3]-Y[1])/DELTA;
BETA[3]:=(Y[1]-Y[2])/DELTA;
GAMMA[1]:=(X[3]-X[2])/DELTA;
GAMMA[2]:=(X[3]-X[1])/DELTA;
GAMMA[3]:=(X[1]-X[2])/DELTA;

FOR M:=1 TO 3 DO
 BEGIN
  FOR N:= 1 TO 3 DO
   BEGIN
    A[M,N]:=(R/(2*DELTA))*(BETA[M]*BETA[N]+GAMMA[M]*GAMMA[N]);
   END;
   FV[M]:= (1/6) * Q * DELTA;
 END;

N1 :=NODE[1];
N2:=NODE[2];
N3:=NODE[3];
FOR L:= 1 TO 3 DO
 BEGIN
 U:=NODE[L];
 K[U,N1]:=A[L,1];
 K[U,N2]:=A[L,2];
 K[U,N3]:=A[L,3];
 F[U]:=FV[L];
 END;

FOR P:=1 TO NN DO
 BEGIN
  FOR Q:=1 TO NN DO
   BEGIN
    G[P,Q]:=G[P,Q] + K[P,Q];
   END;
   GF[P]:=GF[P] + F[P];
  END;
END;
```

```
PROCEDURE SWAP(VAR N,I,P:INTEGER; VAR A:MATRIXTYPE; F:RTYPE);

{THIS PROCEDURE INTERCHANGES THE ROWS OF THE GLOBAL MATRIX
AND GLOBAL VECTOR AFTER PIVOTING}

VAR
  L:INTEGER;
  T:REAL;

BEGIN
 FOR L:=1 TO N DO
  BEGIN
   T:=A[I,L];
   A[I,L]:=A[P,L];
   A[P,L]:=T;
   T:=F[I];
   F[I]:=F[P];
   F[P]:=T;
  END;
END;
```

```
PROCEDURE GAUSS( N:INTEGER; VAR A:MATRIXTYPE; VAR F, X:RTYPE);

{THIS PROCEDURE USES GUASS ELIMINATION TO SOLVE A SET
 OF SIMULTANEOUS LINEAR SYSTEM OF EQUATIONS}

 VAR
  I,J,K,P,L,M:INTEGER;
  SUM,PIVOT,T:REAL;

BEGIN
     FOR I:= 1 TO N DO
      BEGIN
       PIVOT := ABS(A[I,I]);
       P:=I;
       FOR L:= I+1 TO N DO
        BEGIN
         IF (PIVOT < ABS(A[L,I])) THEN
         BEGIN
          PIVOT:=ABS(A[L,I]);
          P:=L;
         END;
      END;
      IF (P<>I) THEN  SWAP(N,I,P,A,F);
      FOR K:=I+1 TO N DO
       BEGIN
        FOR J:=I+1 TO N DO
         BEGIN
          A[K,J]:=A[K,J] - A[K,I] *(A[I,J]/A[I,I]);
          F[J]:=F[J] - A[K,I] * (A[I,J]/A[I,I]);
        END;
        A[K,I]:=0;
      END;
   END;
   X[N]:=F[N]/A[N,N];
   FOR I:=N-1  DOWNTO 1 DO
   BEGIN
    SUM:= 0;
    FOR K:= I+1 TO N DO
     BEGIN
      SUM:=SUM + A[I,K] * X[K];
     END;
     X[I]:= (F[I] - SUM)/A[I,I];
   END;
   FOR I:= 1 TO N DO
   BEGIN
   WRITELN('The value of the function at node ',I,'  is  ',X[I]:6:2);
  END;
END;         {End of Gauss Elimination}
```

```
{ MAIN PROGRAM BEGINS HERE }
      BEGIN
       WRITELN('THIS IS A PROBLEM OF STEADY STATE HEAT CONDUCTION IN 2-D');
       WRITELN('ENTER THE VALUE OF K');
       READLN(K);
       WRITELN('ENTER THE VALUE OF Q');
       READLN(Q);
       WRITELN( ' ENTER THE NUMBER OF NODES ALONG X_AXIS');
       READLN(XNODE);
       WRITELN('ENTER THE NUMBER OF NODES ALONG Y_AXIS');
       READLN(YNODE);
       NNODE:=XNODE * YNODE;
       XTOT:= XNODE - 1 ;
       YTOT:= YNODE - 1;
       NELEM:=XTOT*YTOT*2;
       WRITELN('THE TOTAL NUMBER OF NODES IN THE DOMAIN ARE ', NNODE);
       WRITELN(' THE X AND Y COORDINATES FOR ALL NODES');
       WRITELN;
       WRITELN('ENTER THE INITIAL VALUES OF X AND Y');
       READLN(X_INITIAL,Y_INITIAL);
       WRITELN('ENTER THE FINAL VALUES OF X AND Y');
       READLN(X_FINAL,Y_FINAL);

       {THIS PART OF THE PROGRAM INITIALISES THE K MATRIX ,GLOBAL MATRIX ,
        GLOBAL VECTOR AND LOAD VECTOR}
        FOR I:=1 TO NNODE DO
         BEGIN
          FOR J:=1 TO NNODE DO
           BEGIN
            K_MATRIX[I,J]:=0;
            GLOBAL_MAT[I,J]:=0;
           END;
           F_VECTOR[I]:=0;
           GLOBAL_VECT[I]:=0;
         END;

       RCOORDINATES(XTOT,YTOT,XNODE,YNODE,X_INITIAL,
           X_FINAL,Y_INITIAL,Y_FINAL,XCOORDINATE,YCOORDINATE);
       ELEMENT_COORDINATES(NNODE,XTOT,YTOT,ELEMENT,XCOORDINATE,YCOORDINATE,
          GLOBAL_MAT,K_MATRIX,GLOBAL_VECT,F_VECTOR,K);
       WRITELN('              Global Matrix            Global Vector ');
       FOR I:=1 TO NNODE DO
        BEGIN
         FOR J:=1 TO NNODE DO
          BEGIN
           WRITE(GLOBAL_MAT[I,J]:6:2);
          END;
          WRITELN ('     ',GLOBAL_VECT[I]:6:2);
         END;
         GAUSS(NNODE,GLOBAL_MAT,GLOBAL_VECT,SOL_VECT);
      END.
```

PROGRAM OUTPUT

```
THIS IS A PROBLEM OF STEADY STATE HEAT CONDUCTION IN 2-D
ENTER THE VALUE OF K? 1
ENTER THE VALUE OF Q? 1
ENTER THE NUMBER OF NODES ALONG X_AXIS? 4
ENTER THE NUMBER OF NODES ALONG Y_AXIS? 4
THE TOTAL NUMBER OF NODES IN THE DOMAIN ARE 16
ENTER THE INITIAL VALUES OF X AND Y? ( 0,0 )
ENTER THE FINAL VALUES OF X AND Y? ( 1,1 )

The value of the function at node 1  is   -2.57
The value of the function at node 2  is   -0.27
The value of the function at node 3  is    0.79
The value of the function at node 4  is    1.21
The value of the function at node 5  is    0.70
The value of the function at node 6  is    1.78
The value of the function at node 7  is   -1.31
The value of the function at node 8  is    1.13
The value of the function at node 9  is    6.91
The value of the function at node 10 is   -1.07
The value of the function at node 11 is   -1.14
The value of the function at node 12 is   -0.16
The value of the function at node 13 is   -3.34
The value of the function at node 14 is   -2.74
The value of the function at node 15 is    2.25
The value of the function at node 16 is     .00
```

GLOBAL VECTOR

$$\begin{bmatrix} -2.67 \\ 0.30 \\ 0.89 \\ 4.74 \\ -0.89 \\ -0.89 \\ 0.30 \\ 0.59 \\ 0.00 \\ 0.89 \\ 1.19 \\ 1.48 \\ -0.89 \\ 1.78 \\ 2.07 \\ 2.37 \end{bmatrix}$$

GLOBAL MATRIX

000.00	-729.00	000.00	000.00	-364.50	000.00	000.00	000.00	000.00	000.00	000.00	000.00	000.00	000.00	000.00	000.00
-729.00	121.50	-688.50	000.00	000.00	-729.00	000.00	000.00	000.00	000.00	000.00	000.00	000.00	000.00	000.00	000.00
000.00	-688.50	162.00	-648.00	000.00	000.00	-688.50	000.00	000.00	000.00	000.00	000.00	000.00	000.00	000.00	000.00
000.00	000.00	-648.00	1296.00	000.00	000.00	000.00	-648.00	000.00	000.00	000.00	000.00	000.00	000.00	000.00	000.00
-364.50	000.00	000.00	000.00	-243.00	121.50	000.00	000.00	-243.00	000.00	000.00	000.00	000.00	000.00	000.00	000.00
000.00	-729.00	000.00	000.00	121.50	-81.00	+81.00	000.00	000.00	-607.50	000.00	000.00	000.00	000.00	000.00	000.00
000.00	000.00	-688.50	000.00	000.00	81.00	-40.50	40.50	000.00	000.00	-567.00	000.00	000.00	000.00	000.00	000.00
000.00	000.00	000.00	-648.00	000.00	000.00	40.50	324.50	000.00	000.00	000.00	-526.50	000.00	000.00	000.00	000.00
000.00	000.00	000.00	000.00	-243.00	000.00	000.00	000.00	-121.50	000.00	000.00	000.00	-121.50	000.00	000.00	000.00
000.00	000.00	000.00	000.00	000.00	-607.50	000.00	000.00	000.00	40.50	-40.50	000.00	000.00	-486.00	000.00	000.00
000.00	000.00	000.00	000.00	000.00	000.00	-567.00	000.00	000.00	-40.50	81.00	-81.00	000.00	000.00	-445.50	000.00
000.00	000.00	000.00	000.00	000.00	000.00	000.00	-526.50	000.00	000.00	-81.00	445.50	000.00	000.00	000.00	-405.00
000.00	000.00	000.00	000.00	000.00	000.00	000.00	000.00	-121.50	000.00	000.00	000.00	-243.00	121.50	000.00	000.00
000.00	000.00	000.00	000.00	000.00	000.00	000.00	000.00	000.00	-486.00	000.00	000.00	121.50	162.00	81.00	000.00
000.00	000.00	000.00	000.00	000.00	000.00	000.00	000.00	000.00	000.00	-445.50	000.00	000.00	81.00	243.00	040.50
000.00	000.00	000.00	000.00	000.00	000.00	000.00	000.00	000.00	000.00	000.00	-405.00	000.00	000.00	040.50	324.00