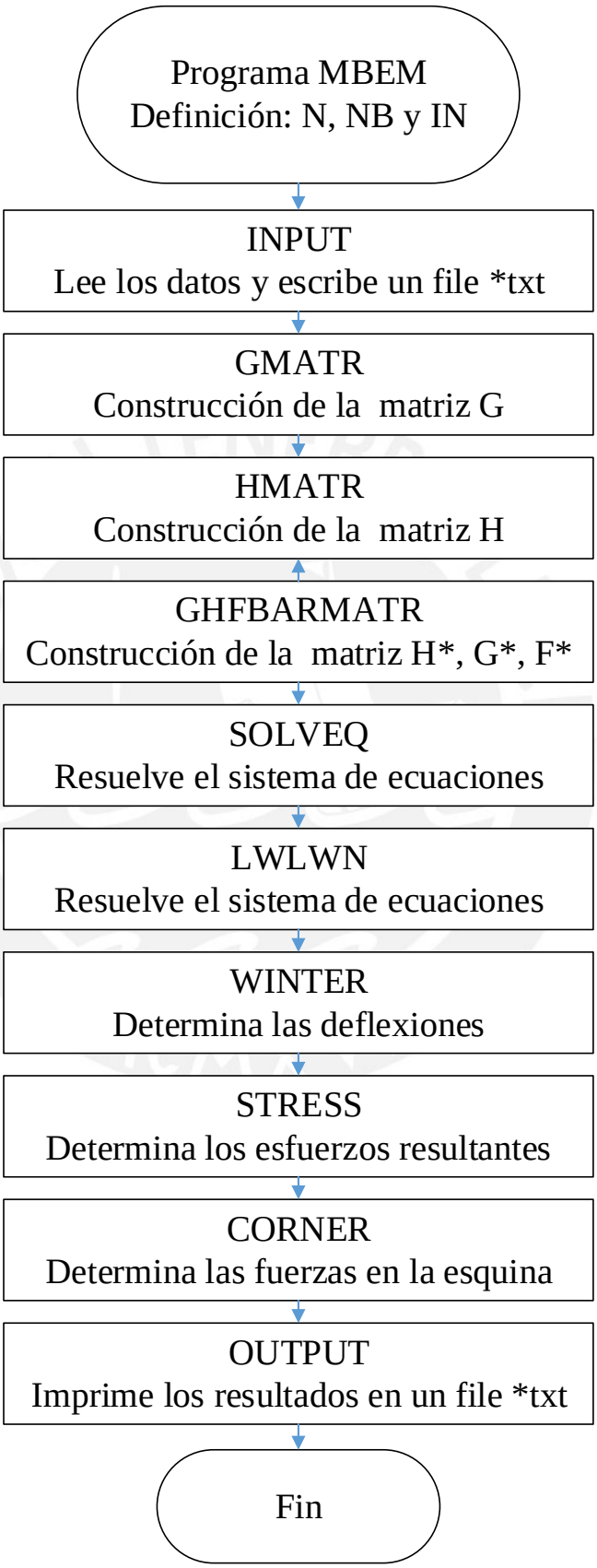
TESIS PUCP

Anexo A: Diagrama de flujo del programa MBEM en Matlab



Tesis publicada con autorización del autor
No olvide citar esta tesis

TESIS PUCP

PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

Anexo B: Código de programa MBEM para el análisis de losas

El profesor John T. Katsikadelis de la Universidad Técnica Nacional de Atenas, Grecia desarrolló el programa PLBECOM en leguaje FORTRAN, el cual resuelve problemas de losas empleando el método de los elementos de contorno con elementos constantes. Al respecto, el programa en mención se traduce en el lenguaje MATLAB, el cual se le denomina MBEM, cuya denominación se asemeja al programa MFEM para elementos finitos.

El programa define los parámetros N, NC, NFC, NB y IN, los cuales especifican el número de elementos en el contorno, numero de esquinas, numero de esquinas con lados adyacentes libres, numero de contornos y número de puntos internos, respectivamente. Asimismo, el programa define los parámetros NPS (número de puntos interno para soporte), NLS (número de líneas para soporte), NELS (número de elemento en las líneas para soporte), NCL (número de cargas concentradas), NLL (número de cargas lineales) y NELL (número de elementos en las cargas lineales).

Seguidamente, el programa contiene subrutinas:

GMATR: ensambla la matriz $[G]$ definido por la ecuación 3.144.

HMATR: ensambla la matriz $[H]$ definido por la ecuación 3.143.

HGFBBARMATR: ensambla las matrices $[\overline{H}]$, $[\overline{G}]$ y $[\overline{F}]$ definido por la ecuación 3.148a, b y c.

FLOAD: ensambla el vector carga $\{F\}=\{F_1\ F_2\ F_3\}^T$ definido por la ecuación 3.145.

SOLVEQ: resuelve el sistema de ecuaciones lineales 3.142 y 3.147, y ensambla los vectores $\{W\}$, $\{WN\}$, $\{MN\}$, $\{VN\}$, $\{WC\}$ y $\{RC\}$.

LWLWN: determina el Laplaciano $\{LW\}$ y su derivada $\{LWN\}$ en el contorno para la solución de las integrales de las ecuaciones 3.132 y 3.133.

WINTER: determina las deflexiones w en los IN puntos internos mediante la ecuación 3.130.

STRESS: determina las fuerzas internas M_x, M_y, M_{xy}, Q_x y Q_y en los puntos internos mediante la ecuación 3.34, 3.42 y 3.136a, b, c, d, e .

CORNER: determina las fuerzas resultantes R_k en las esquinas.

Tesis publicada con autorización del autor
No olvide citar esta tesis

Fase preprocesamiento

El pre proceso es la primera etapa del programa MBEM cosiste en la definición de la geometría.

```
fprintf('********* PRE PROCESING ********** \n');
inputfile=input('Enter file name for plate: ','s');
fid = fopen (inputfile,'w');
user=input('User name: ','s');
title=input('Title: ','s');
%=========================================================================
% ELASTIC CONSTANTS
%=========================================================================
el=2.17*10^8;  % Modulus of elasticity
pn=0.15;       % Poisson ratio
%=========================================================================
% PLATE THICKNESS
%=========================================================================
hplate=0.25;
%=========================================================================
% BOUNDARY DATA
%=========================================================================
nb=1;                          % number of boundaries
%
ns=4;                          % number of sides.
```

Asimismo, se define las condiciones de apoyo, para tal fin se asignó un código de valores de 0 y 1, de acuerdo a la condición de apoyo.

```
%=========================================================================
% ASSIGN boundary conditions at the nbe boundary nodal points
%=========================================================================
% Boundary conditions for each side:
% ********************************************************************
% *                    |  a1   |  a2  |  a3  |  b1   |  b2   |  b3  | *
% ********************************************************************
% * Simply supported   |  1    |  0   |  w*  |   0   |   1   |  Mn* | *
% * Clamped            |  1    |  0   |  w*  |   1   |   0   |  wn* | *
% * Free               |  0    |  1   |  Vn* |   0   |   1   |  Mn* | *
% * Guided             |  0    |  1   |  Vn* |   0   |   1   |  wn* | *
% * Elastic support    |  kT   |  1   |  Vn* |   kR  |   1   |  Mn* | *
% ********************************************************************
%    kT and kR are the transverse and rotational spring stiffnesses
%    The star designates prescribed quantity
% abs:= array including the values of a1,a2,a3,b1,b2,b3 of each side
%----------------abs-------------------|
%        | 1  |  2 | 3  |  4 |  5 |  6 |
%--------------------------------------
% side 1 | a1 | a2 | a3 | b1 | b2 | b3 |
% side 2 | ...                         |
% ...
% side ns| ...                         |
%---------------------------------------|
%abs=zeros(ns,6);
abs=[ 1,0,0, 1,0,0;
      1,0,0, 1,0,0;
```


Tesis publicada con autorización del autor
No olvide citar esta tesis

Además, se asigna el número de elementos para la división, y las condiciones de carga.

```
%=========================================================================
% GENERATION of the coordinates of the IN interior nodal points xin,yin
% where deflections and the stress resultants will be evaluated.
%=========================================================================
% First a rectangular mesh with nrow X ncol nodal points is applied on the
% domain of the  plate.Then the function checkin.m detects the nodal points
% inside the domain. Points at a distance from the boundary smaller than a
% prescribed value "dist" are excluded.
% -------------------------------------------------------------------------
% CALL function meshpoints to generate nrow X ncol nodal points
nrow=21;
ncol=21;
[xint,yint,mt]=meshpoints(nrow,ncol,xl,yl);
```

```
%=========================================================================
% SURFACE LOADS
%=========================================================================
% fx,fy,f0 : coefficients of the linear load function f=fx*x+fy*y+f0
fx=0;
fy=0;
f0=5000;
```

La división de la losa se muestra en un gráfico, donde se detalla el número de elementos de contorno y los puntos en los cuales se quiere conocer los esfuerzos resultantes.

```
%=========================================================================
% PLOT GEOMETRY and NODAL POINTS
%=========================================================================
% CALL function plotplate to view geometry and nodal points
plotplate;
```

Fase procesamiento

El proceso es la segunda etapa del programa, que consiste en la resolución de los sistemas de ecuaciones.

```
%   SUBROUTINE  INPUTPL(XL,YL,XM,YM,JN,KN,N,NC,NFC,XC,YC,XFC,YFC
%    &    ,CCODE,NB,ANGLE,ICOR,IN,XIN,YIN,XSL,YSL
%     &    ,XSM,YSM,NLS,NELS,JLS,KLS
%      &    ,NPS,XPS,YPS,XCL,YCL,NCL,FCL,NELL,NLL,XLL,YLL,JL,KL,FLL,PN
%      &    ,EL,H,A1,A2,A3,B1,B2,B3,C1,C2,C3,CURV,FX,FY,F0)
%   This subroutine reads the data from the input file and writes
%   them in the output file
% Read the number of the last element of each boundary
NL=zeros(1,NB);
for i=1:NB
    NL(i)=nl(i);
end
```

En esta etapa se construye la matriz G que contiene las condiciones de esfuerzos y desplazamientos. Asimismo, se ensambla la matriz H.

Tesis publicada con autorización del autor
No olvide citar esta tesis

```
493      %   GMATRPL(XL,YL,XM,YM,XFC,YFC,JN,KN,N,NFC,XSL,YSL (2)
494      %   1 ,XSM,YSM ,JLS,KLS,NELS,NLS,XPS,YPS,NPS,G,UCL,LUCL,UCP,LUCP)
495      % This subroutine computes the elements of G matrix
496      % Compute the G matrix
497 -    G=zeros(2*N+NFC+NELS+NPS);
498 -    for i=1:N
499 -        X0=XM(i);
500 -        Y0=YM(i);
501 -        XA1=xl(JN(i));
502 -        YA1=yl(JN(i));
503 -        XA2=xl(KN(i));
504 -        YA2=xl(KN(i));
505 -        SAL=sqrt((XA1-XA2)^2+(YA1-YA2)^2);
506 -        AAX=(XA2-XA1)/2;
507 -        AAY=(YA2-YA1)/2;
508 -        ENAX=2*AAY/SAL;
509 -        ENAY=-2*AAX/SAL;
510 -        for j=1:N
511 -            X1=XL(JN(j));
512 -            Y1=YL(JN(j));
513 -            X2=XL(KN(j));
514 -            Y2=YL(KN(j));
515 -            SL=sqrt((X1-X2)^2+(Y1-Y2)^2);
516 -            if i==j
517 -                G(i,j)=SL^3/96/pi/D*(log(SL/2)-1/3);
518 -                G(i,j+N+NFC)=0;
519 -                G(i+N+NFC,j)=0;
520 -                G(i+N+NFC,j+N+NFC)=-(SL/8/D/pi-SL/4/D/pi*log(SL/2));
521 -            else
522 -                [U,UN,U1N,U1]=RLINTG(X0,Y0,X1,Y1,X2,Y2,ENAX,ENAY,D);%call
523 -                G(i,j)=U;
524 -                G(i,j+N+NFC)=-UN;
525 -                G(i+N+NFC,j)=U1;
526 -                G(i+N+NFC,j+N+NFC)=-U1N;
527 -            end
528 -        end
```

```
674      %   SUBROUTINE HMATRPL(XL,YL,XM,YM,XC,YC,JN,KN,ICOR,N,NFC,XSM,YSM (5)
675      %   1         ,NELS,XPS,YPS,NPS,ANGLE,H,CURV)
676      % This subroutine computes the elements of H matrix
677 -    H=zeros(2*N+NFC+NELS+NPS,2*N+NFC);
678 -    for i=1:N
679 -        X0=XM(i);
680 -        Y0=YM(i);
681 -        XA1=XL(JN(i));
682 -        YA1=YL(JN(i));
683 -        XA2=XL(KN(i));
684 -        YA2=YL(KN(i));
685 -        SAL=sqrt((XA1-XA2)^2+(YA1-YA2)^2);
686 -        AAX=(XA2-XA1)/2;
687 -        AAY=(YA2-YA1)/2;
688 -        ENAX=2*AAY/SAL;
689 -        ENAY=-2*AAX/SAL;
690 -        for j=1:N
691 -            X1=XL(JN(j));
692 -            Y1=YL(JN(j));
693 -            X2=XL(KN(j));
694 -            Y2=YL(KN(j));
```


Tesis publicada con autorización del autor
No olvide citar esta tesis

```
695 -           SL=sqrt((X1-X2)^2+(Y1-Y2)^2);
696 -           CUR=CURV(j);
697 -           if i==j
698 -               H(i,j)=0.5+(1-pn)/4/pi*CUR*SAL;%need PN, CURV
699 -               H(i,j+N+NFC)=-(-(1+pn)/4/pi*SL*log(SL/2)+(1-pn)/8/pi*SL);
700 -               H(i+N+NFC,j)=-(1+pn)/pi/SL;
701 -               H(i+N+NFC,j+N+NFC)=-(-0.5);
702 -           else
703 -               [UV,UM,U1V,U1M]=RLINTH(X0,Y0,X1,Y1,X2,Y2,CUR,pn,ENAX,ENAY);
704 -               H(i,j)=UV;
705 -               H(i,j+N+NFC)=-UM;
706 -               H(i+N+NFC,j)=U1V;
707 -               H(i+N+NFC,j+N+NFC)=-U1M;
708 -           end
709 -       end
```

Ensambla las matrices $\overline{H}$, $\overline{G}$ y $\overline{F}$.

```
918       %=====================================================================
919       %   SUBROUTINE HGBARMATR(N,NFC,A1,A2,A3,B1,B2,B3,C1,C2,C3,HBAR,GBAR (8)
920       %   1            ,FBAR)
921       %  This subroutine produces the matrices Hbar,Gbar of Eq. 2.139
922 -     HBAR=zeros(2*N+NFC);
923 -     GBAR=zeros(2*N+NFC);
924 -     FBAR=zeros(1,2*N+NFC);
925 -     for i=1:N
926 -         HBAR(i,i)=A1(i);
927 -         HBAR(i+N+NFC,i+N+NFC)=B1(i);
928 -         GBAR(i,i)=A2(i);
929 -         GBAR(i+N+NFC,i+N+NFC)=B2(i);
930 -         FBAR(i)=A3(i);
931 -         FBAR(i+N+NFC)=B3(i);
932 -     end
933 -     if NFC>0
934 -         for i=1:NFC
935 -             HBAR(i+N,i+N)=C1(i);
936 -             GBAR(i+N,i+N)=C2(i);
937 -             FBAR(i+N)=C3(i);
938 -         end
939 -     end
```

Resuelve el sistema de ecuaciones $H\begin{Bmatrix} w \\ w_c \\ w_n \end{Bmatrix} = G\begin{Bmatrix} V \\ R \\ M \end{Bmatrix} + F$ y $\overline{H}\begin{Bmatrix} w \\ w_c \\ w_n \end{Bmatrix} = \overline{G}\begin{Bmatrix} V \\ R \\ M \end{Bmatrix} + \overline{F}$

```
940       %=====================================================================
941       %   SUBROUTINE SOLVEQPL(N,NFC,NELS,NPS,H,G,F,HBAR,GBAR,FBAR
942       %   &   ,W,WC,WN,VN,MN,SL,SP,TW)
943       %  This subroutine solves the system of Eq. 3.141 and 3.147
944 -     [FLOAD,FLOADL]=FLOAD2D(XL,YL,XM,YM,JN,KN,N,NFC,XFC,YFC,NELS,...
945           XSM,YSM,XPS,YPS,NPS,D,FX,FY,F0);
946 -     A=zeros(4*N+2*NFC+NELS+NPS);
947 -     B=zeros(1,4*N+2*NFC+NELS+NPS);
948 -     for i=1:2*N+NFC+NPS+NELS
949 -         for j=1:2*N+NFC
950 -             A(i,j)=H(i,j);
951 -         end
```


Tesis publicada con autorización del autor
No olvide citar esta tesis

```
952 -       for j=1:2*N+NFC+NPS+NELS
953 -           A(i,j+2*N+NFC)=-G(i,j);
954 -       end
955 -       B(i)=FLOAD(i);
956 -   end
957
958 - for i=1:2*N+NFC
959 -       for j=1:2*N+NFC
960 -           A(i+2*N+NFC+NELS+NPS,j)=HBAR(i,j);
961 -           A(i+2*N+NFC+NELS+NPS,j+2*N+NFC)=GBAR(i,j);
962 -       end
963 -       B(i+2*N+NFC+NELS+NPS)=FBAR(i);
964 -   end
965       %[A,B]=LEQS(4*N+2*NFC+NELS+NPS);
966 -     X=A\B';
967 -     W=zeros(1,N);
968 -     WN=zeros(1,N);
969 -     VN=zeros(1,N);
970 -     MN=zeros(1,N);
971 - for i=1:N
972 -       W(i)=X(i);
973 -       WN(i)=X(i+N+NFC);
974 -       VN(i)=X(i+2*N+NFC);
975 -       MN(i)=X(i+3*N+2*NFC);
976 -   end
977 -     WC=zeros(1,NFC);
978 -     TW=zeros(1,NFC);
979 - for i=1:NFC
980 -       WC(i)=X(i+N);
981 -       TW(i)=X(i+3*N+NFC);
982 -   end
```

Determina el Laplaciano $\{LW\}$ y su derivada $\{LWN\}$.

```
992       %=====================================================================
993       %   SUBROUTINE LWLWN(N,XL,YL,JN,KN,XM,YM,WB,WN,FLOAD2D (10)
994       %   1             ,NELS,UCL,LUCL,NPS,UCP,LUCP,SL,SP,LW,LWN)
995       % This subroutine computes the Laplacian of the deflection w and
996       % its normal derivative at the boundary
997       %V=zeros(N,N);
998       %LV=zeros(N,N);
999       %LVN=zeros(N,N);
1000 -     [V,VN,LV,LVN]=KERNLMATR(XL,YL,XM,YM,JN,KN,N,N,D);
1001 -   for i=1:N
1002 -       for j=1:N
1003 -           V(i,j)=D*V(i,j);
1004 -           VN(i,j)=D*VN(i,j);
1005 -           LV(i,j)=D*LV(i,j);
1006 -           LVN(i,j)=D*LVN(i,j);
1007 -       end
1008 -   end
1009 -   for i=1:N
1010 -       LVN(i,i)=LVN(i,i)-0.5;
1011 -   end
1012 -     F1=zeros(1,N);
1013 -     F2=zeros(1,2*N);
```


Tesis publicada con autorización del autor
No olvide citar esta tesis

```
for i=1:N
    for j=1:N
        F1(i)=F1(i)+LVN(i,j)*W(j)-LV(i,j)*WN(j);
    end
    for j=1:NELS
        F2(i)=F2(i)+UCL(i,j)*SL(j);
        F2(i+N)=F2(i+N)+LUCL(i,j)*SL(j);
    end
    for j=1:NPS
        F2(i)=F2(i+N)+UCP(i,j)*SP(j);
        F2(i+N)=F2(i+N)+LUCP(i,j)*SP(j);
    end
end
FLOAD22D=zeros(1,2*N);
AA=zeros(2*N);
for i=1:N
    FLOAD22D(i)=FLOAD22D(i)+F1(i)+F2(i);
    FLOAD22D(i+N)=FLOAD22D(i+N)+F2(i+N);
    for j=1:N
        AA(i,j)=V(i,j);
        AA(i,N+j)=-VN(i,j);
        AA(i+N,j)=LV(i,j);
        AA(i+N,j+N)=-LVN(i,j);
    end
end
%[A,FLOAD2D,LSING]=LEQS(2*N);
XX=AA\FLOAD22D';
LWN=zeros(1,N);
LW=zeros(1,N);
for i=1:N
    LWN(i)=XX(i);
    LW(i)=XX(i+N);
end
```

Determina las deflexiones en los puntos internos del dominio.

```
%=====================================================================
%   SUBROUTINE WINTERPL(N,XL,YL,XM,YM,JN,KN,XIN,YIN,IN,XSL,YSL,JLS (14)
%   1  ,KLS,NELS,NLS,XPS,YPS,NPS,XLL,YLL,JL,KL,NLL,NLT,FLL,XCL,YCL
%     1  ,NCL,FCL,WB,WN,LW,LWN,SL,SP,W,WX,WY,WXX,WYY,WXY,LWX,LWY)
%   This subroutine computes the values of w,wx,wy,wxx,wyy,wxy,Lwx,Lwy
%   at the IN internal points
%  Compute matrices for line supports
[P1,PX,PY,PXX,PYY,PXY,LPX,LPY]=FLOADIN(XL,YL,XM,YM,JN,KN,N,IN,...
XIN,YIN,xll,yll,JN,KN,NLT,NLL,FLL,XCL,YCL,NCL,FCL);
if NLS>0%not used
    [UL,ULX,ULY,ULXX,ULYY,ULXY,LULX,LULY]=UINDERLMATR(XIN,YIN,IN,...
    XSL,YSL,JSL,KSL,NLS,NELS);
    for i=1:IN
        for j=1:NELS
            P1(i)=P1(i)+UL(i,j)*SL(j);
            PX(i)=PX(i)+ULX(i,j)*SL(j);
            PY(i)=PY(i)+ULY(i,j)*SL(j);
            PXX(i)=PXX(i)+ULXX(i,j)*SL(j);
            PXY(i)=PXY(i)+ULXY(i,j)*SL(j);
            PYY(i)=PYY(i)+ULYY(i,j)*SL(j);
            LPX(i)=LPX(i)+LULX(i,j)*SL(j);
            LPY(i)=LPY(i)+LULY(i,j)*SL(j);
        end
```


Tesis publicada con autorización del autor
No olvide citar esta tesis

```
1073      %  Compute matrices for point supports
1074 -    if NPS>0%not used
1075 -        [UP,UPX,UPY,UPXX,UPYY,UPXY,LUPX,LUPY]=UINDERPMATRIN(XPS,YPS,...
1076          NPS,XIN,YIN,IN);
1077 -        for i=1:IN
1078 -            for j=1:NPS
1079 -                P1(i)=P1(i)+UP(i,j)*SP(j);
1080 -                PX(i)=PX(i)+UPX(i,j)*SP(j);
1081 -                PY(i)=PY(i)+UPY(i,j)*SP(j);
1082 -                PXX(i)=PXX(i)+UPXX(i,j)*SP(j);
1083 -                PXY(i)=PXY(i)+UPXY(i,j)*SP(j);
1084 -                PYY(i)=PYY(i)+UPYY(i,j)*SP(j);
1085 -                LPX(i)=LPX(i)+LUPX(i,j)*SP(j);
1086 -                LPY(i)=LPY(i)+LUPY(i,j)*SP(j);
1087 -            end
1088 -        end
1089 -    end
```

```
1090 -    [V,VN,LV,LVN]=KERNLMATR(XL,YL,XIN,YIN,JN,KN,N,IN,D);
1091 -    [VX,VNX,LVX,LVNX,VY,VNY,LVY,LVNY]=KERNLMATR1(XL,YL,XIN,YIN,JN,KN,N,IN,D);
1092 -    [VXX,VNXX,LVXX,LVNXX,VYY,VNYY,LVYY,LVNYY,VXY,VNXY,LVXY,LVNXY]=...
1093          KERNLMATR2(XL,YL,XIN,YIN,JN,KN,N,IN,D);
1094 -    W=-V.*LWN+LVN.*WB+VN.*LW-LV.*WN;
1095 -    WX=-VX*LWN+LVNX*WB+VNX*LW-LV*WN;
1096 -    WY=-VY*LWN+LVNY*WB+VNY*LW-LVY*WN;
1097 -    WXX=-VXX*LWN+LVNXX*WB+VNXX*LW-LVXX*WN;
1098 -    WXY=-VXY*LWN+LVNXY*WB+VNXY*LW-LVXY*WN;
1099 -    WYY=-VYY*LWN+LVNYY*WB+VNYY*LW-LVYY*WN;
1100 -    LWX=-LVX*LWN+LVNX*LW;
1101 -    LWY=-LVY*LWN+LVNY*LW;
1102
1103 -    for i=1:IN
1104 -        W(i)=P1(i)+D*W(i);
1105 -        WX(i)=PX(i)+D*WX(i);
1106 -        WY(i)=PY(i)+D*WY(i);
1107 -        WXX(i)=PXX(i)+D*WXX(i);
1108 -        WXY(i)=PXY(i)+D*WXY(i);
1109 -        WYY(i)=PYY(i)+D*WYY(i);
1110 -        LWX(i)=LPX(i)+D*LWX(i);
1111 -        LWY(i)=LPY(i)+D*LWY(i);
1112 -    end
```

Obtención de las fuerzas internas resultantes en el dominio de la losa.

```
1566      %=====================================================================
1567      %   SUBROUTINE STRESSPL(IN,WXX,WYY,WXY,LWX,LWY,MX,MY,MXY,QX,QY)
1568      %   This subroutine computes the stress resultants at the IN internal
1569      %   points
1570 -    MX=zeros(1,IN);
1571 -    MY=zeros(1,IN);
1572 -    MXY=zeros(1,IN);
1573 -    QY=zeros(1,IN);
1574 -    QX=zeros(1,IN);
1575 -    for i=1:IN
1576 -        MX(i)=-D*(WXX(i)+PN*WYY(i));
1577 -        MY(i)=-D*(WYY(i)+PN*WXX(i));
1578 -        MXY(i)=D*(1-PN)*WXY(i);
1579 -        QY(i)=-D*LWY(i);
1580 -        QX(i)=-D*LWX(i);
1581 -    end
```


Tesis publicada con autorización del autor
No olvide citar esta tesis

TESIS PUCP

PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

Fase posprocesamiento

El post proceso es la tercera etapa del programa, esta etapa consiste en la salida de resultados, los cuales pueden ser en archivo txt y gráficos.

```
2537      %*********************************POST PROCESS*****************************
2538 -    [X,Y]=meshgrid(0:1:10,0:1:10);
2539 -    contour(X,Y,W)
2540 -    hold on
2541 -    contour(X,Y,MX)
```


Tesis publicada con autorización del autor
No olvide citar esta tesis