ASSIGNMENT 1

In this report are included the results obtained in the section C and E. At the end of the document is an annex where the modified codes are copied.

SECTION C:

After implementing the missing parts of the code, it will be explained and analyzed the results obtained in the section C of the exercise for the non-symmetric and tension-only model. As a first point, the missing data should be established:

- It has been assumed q_∞ as: $q_\infty = (3/2) \cdot r_0$.
- The stress path is defined by $\alpha = 250$, $\beta = -350$ and $\gamma = 500$.

A. NON-SYMMETRIC TENSION-COMPRESION DAMAGE MODEL

The results obtained for the three loading/unloading cases will be analyzed in the next points.

a. Uniaxial loading/unloading: SIGMAP=[0,0; 250,0; -350,0; 500,0]

The tensile yield stress is 200MPa and it has assumed that three times this stress as the compression yield stress. These yield stresses can be seen in the following figure (blue lines). The continuous line defines the elastic surface at the beginning of the loading, where the crossing point between the elastic surface and the axis are the yield stresses. The dashed lines show the evolution of the elastic surface due to the loading/unloading path. Furthermore, the red line shows the stress path. In this case the loading is uniaxial, hence the loading stress will be on the axis.

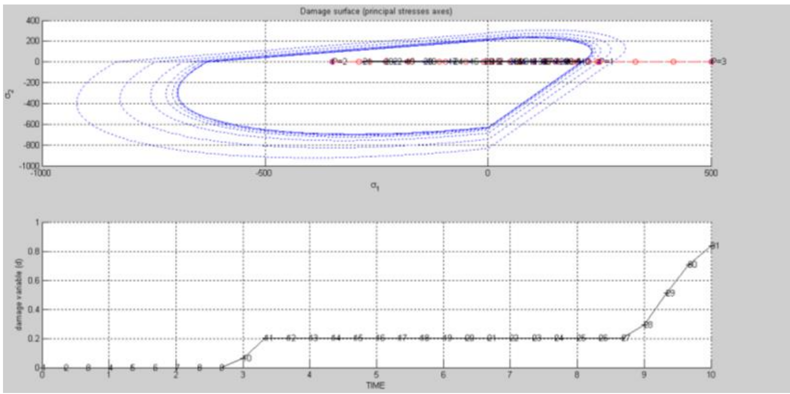


Figure 1. 1 Elastic surface and damage

The loading/unloading process is divided into three phases. In the previous plot is difficult to analyze the loading/unloading path, for this reason it has been plot the stress-strain curve for the three directions.

In the first stage, the element is pulled to over the yielding limit (σ =250 > σ_y = 200). The red line shows a linear first part until the yield limit, which corresponds to the elastic range. After this point, the element plasticizes and the non-linearity can be seen in the curve. The other lines are the stress-strain curve in the other two directions. Although they seem lines, if a zoom is done it could be seen that the lines has a similar form as the red one.

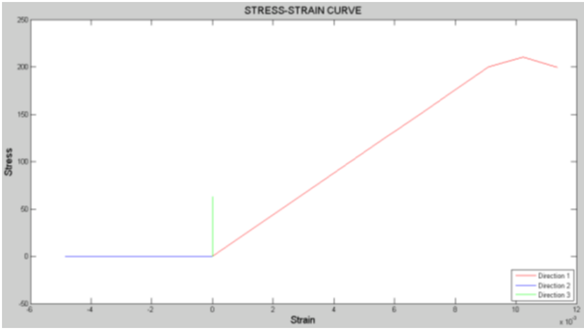


Figure 1. 2 Phase 1: Loading

Then, the element is unloaded but the yield limit has grown so much, that it is not achieved the limits. Therefore, the point will stay in the elastic surface during the unloading and a line will be drawn in the stress-strain curve.

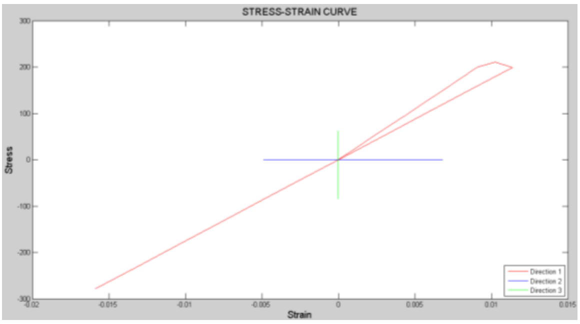


Figure 1. 3 Phase 2: Unloading

Finally, the element is pulled again and the plastic part is much visible and a significant degradation of stiffness can also be observed. The lost of stiffness is applied by the constant "d" (damage), which is drawn in the Figure 1.1.

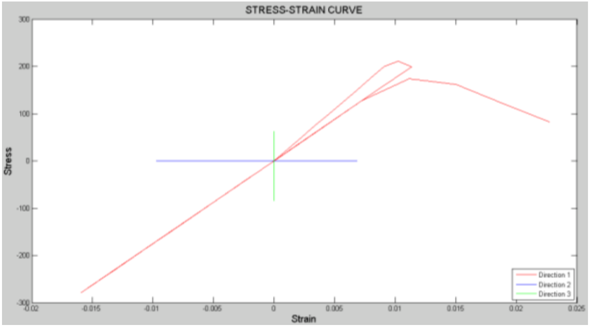


Figure 1. 4 Phase 3: Loading

b. Uniaxial and biaxial loading/unloading: SIGMAP=[0,0; 250,0; -350,-350; 500,500]

In this case, the stresses in the Direction 1 are the same as in the previous point however, the stresses in the Direction 2 has been changed in the second and third phase. Therefore, a uniaxial load is applied upon the element in the first stage and a biaxial loading in the other two stages as it has been plotted in the following figure:

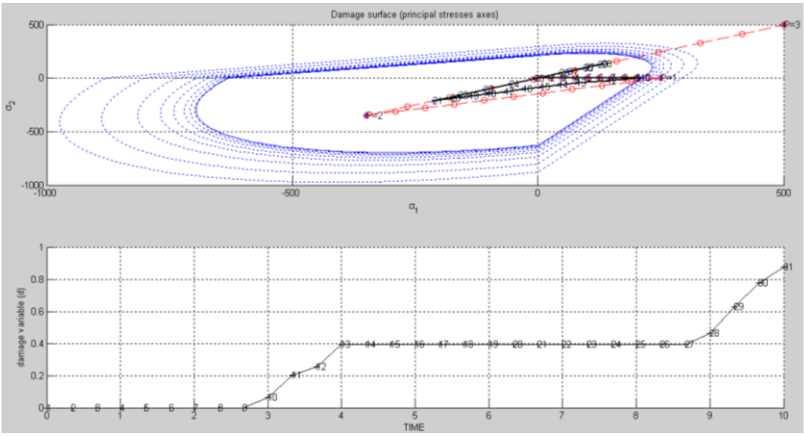
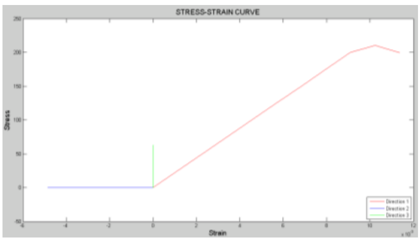
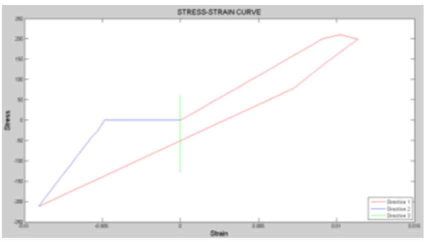


Figure 1. 5 Elastic surface and damage

The three phases of the loading/unloading curve are drawn in the next figures. In the first stage, the element is loaded uniaxially until the yield stress is overcomen. The stresses and strains are the same as in the previous case. Nevertheless, in the second stage, it is applied compression stresses in both directions. The compression stresses have the same value in compression and the blue and red line will concur in the same point because the point is inside the elastic surface. Lastly, it is applied a tensile stress again in both direction and both curves are overlapped (it is only visible the blue line), due to its isotropic behavior.





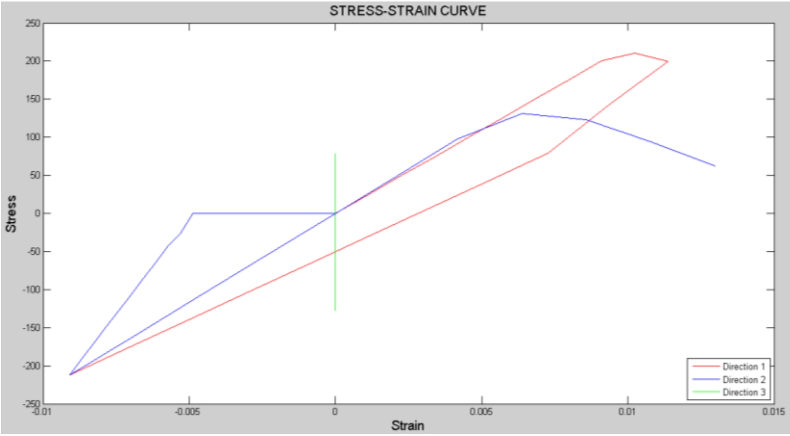


Figure 1. 6 Stress-strain curve

c. Biaxial loading: SIGMAP=[0,0; 250,250; -350,-350; 500,500]

In this case, the loading/unloading is biaxial. In the first phase, the element is loaded until the yielding stress is slightly exceeded. Then, the element is compressed but the point will remain within the elastic surface. Finally, a tensile stress will be applied again.

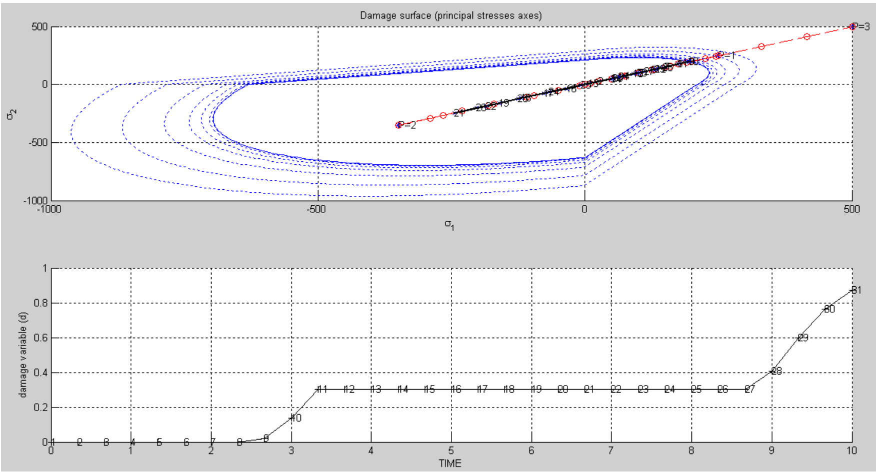
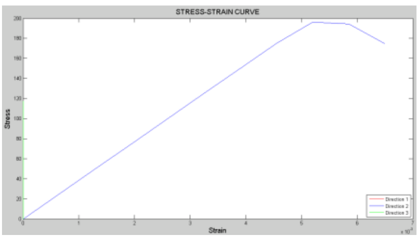
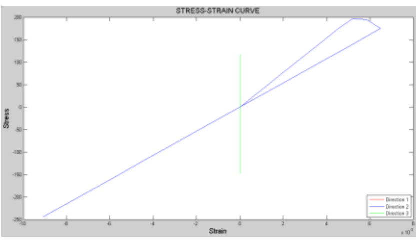


Figure 1. 7 Elastic surface and damage

As the material is isotropic, the stress-strain curve is overlapped in two directions. As in the previous cases, there is an initial elastic-linear part and then it plasticizes. The instant where the material is plasticizing can be clearly seen in the Damage/Time and Stress/Strain plot.





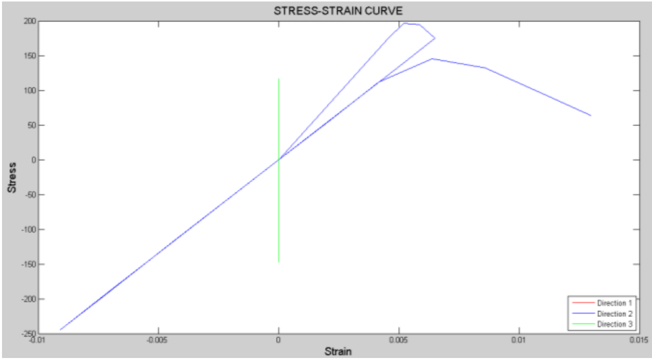


Figure 1. 8 Stress-strain curve

B. TENSION-ONLY DAMAGE MODEL:

a. Uniaxial loading/unloading: SIGMAP=[0,0; 250,0; -350,0; 500,0]

There is not any yield limit for tensile stresses, which means that there is not going to be any damage due to tensile loading. The results will be very similar to the previous case because the tensile yielding stress was so high that the loading was not able to push the point until the limit.

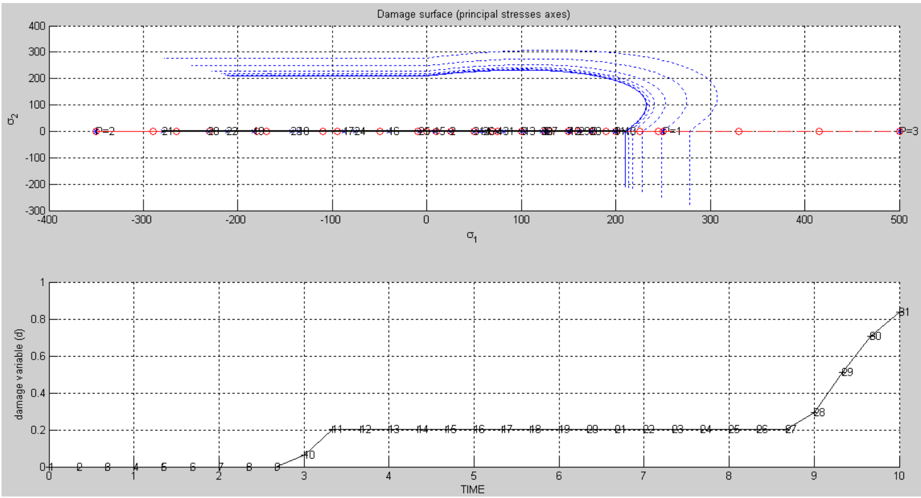


Figure 1. 9 Elastic surface and damage

The stress-strain curves of the both models are the same. However, if the loading point will be pushed in the “Non-symmetric tension-compression damage model” out of the elastic surface while a tensile stress is applied, the surface would change.

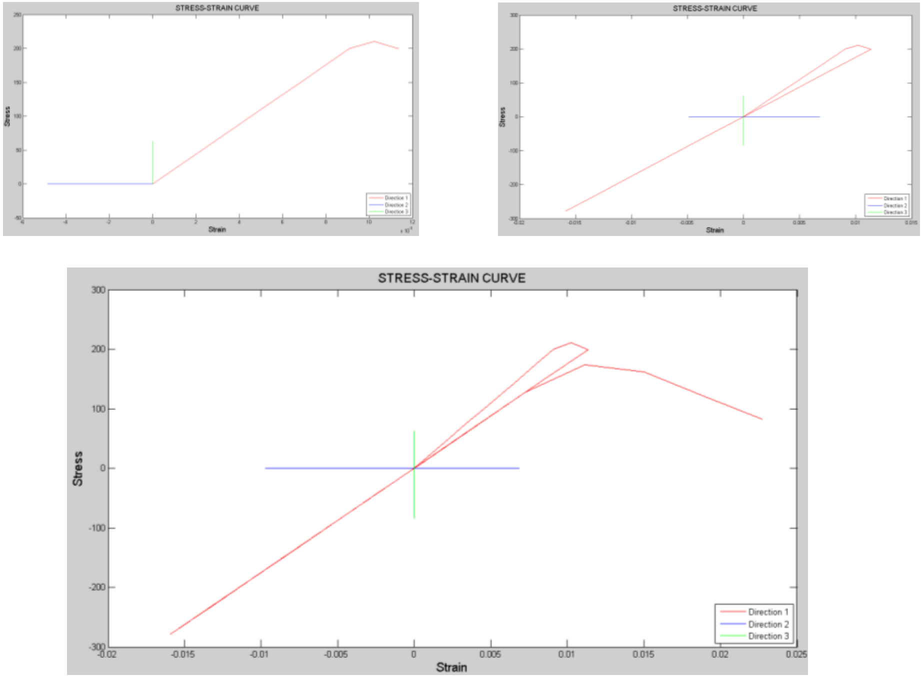


Figure 1. 10 Stress-strain curve

b. Uniaxial and biaxial loading/unloading: SIGMAP=[0,0; 250,0; -350,-350; 500,500]

The Damage/Time plot shows the evolution of the damage. The point would be inside the elastic surface when the damage remains constant. Therefore, the elastic surface will increase between the steps 9-11 and 27-31.

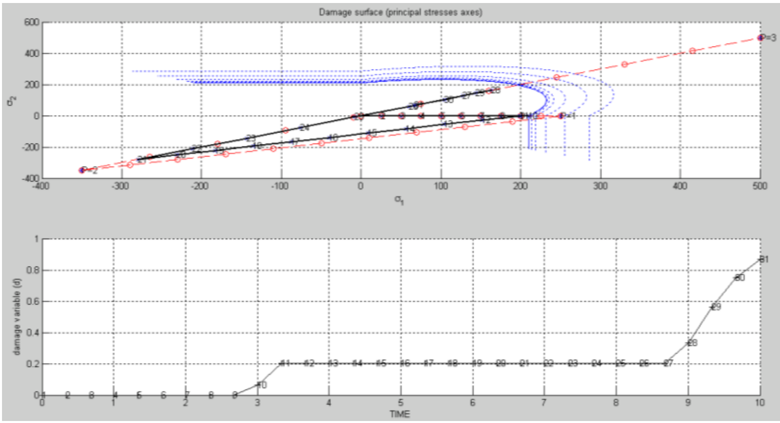


Figure 1. 11 Elastic surface and damage

The stress–strain curve is very helpful to understand the behavior of the point and it can be seen easily the plastifications.

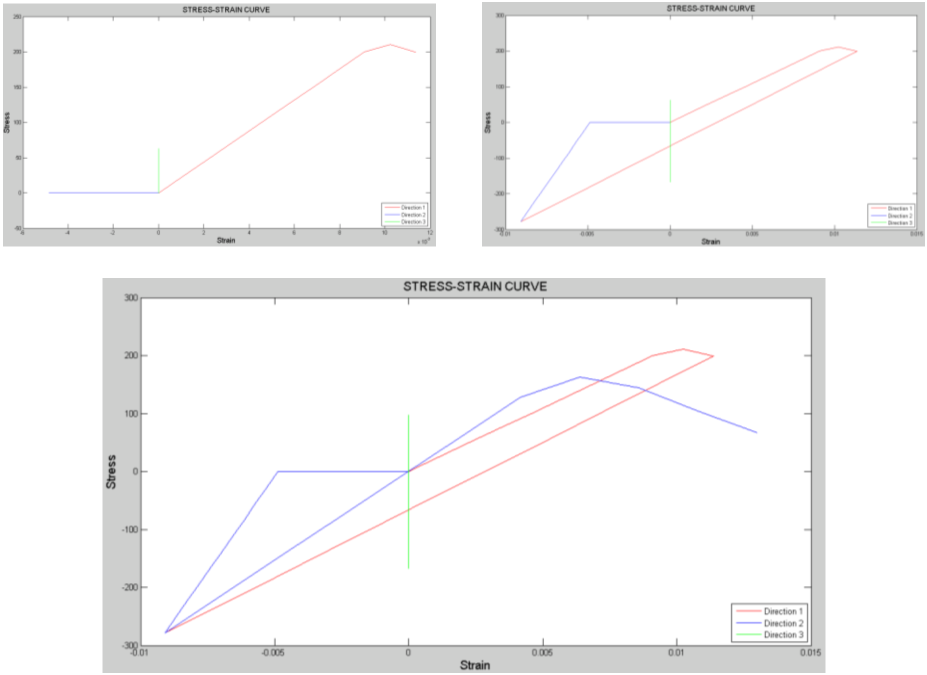


Figure 1. 12 Stress-strain curve

c. **Biaxial loading:** SIGMAP=[0,0; 250,250; -350,-350; 500,500]

As in the previous model the stress-strain curve are overlapped in two directions. The material is only damaged in one direction, in compression. But as in the previous model the material does not reach to the plastification limit in compression, the results must be the same as in that model.

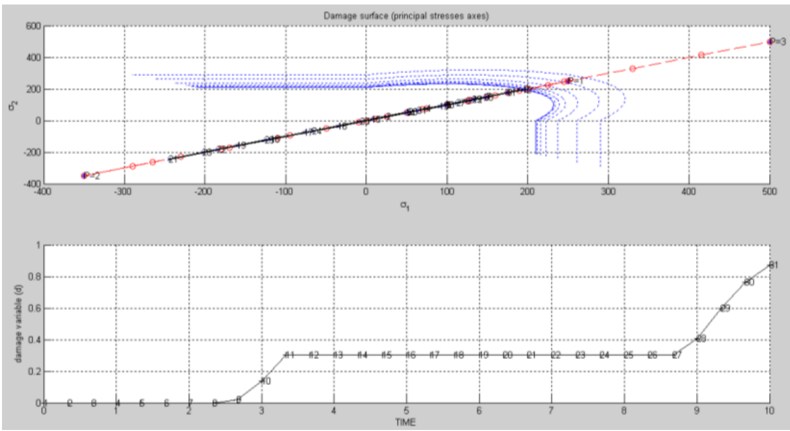
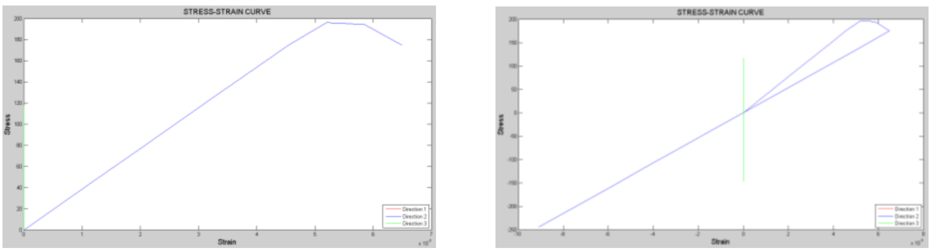


Figure 1. 13 Elastic surface and damage



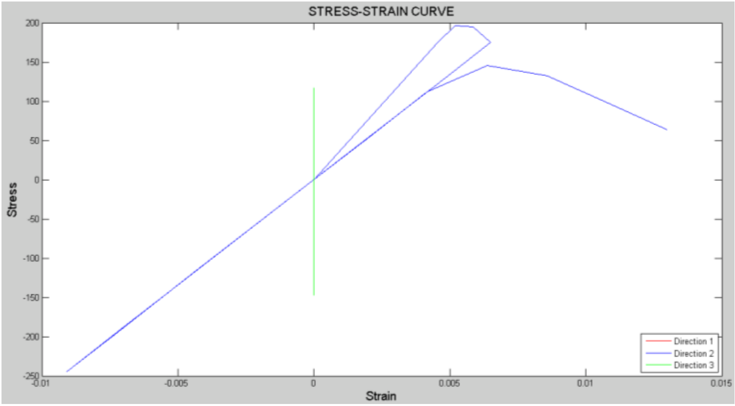


Figure 1. 14 Stress-strain curve

SECTION E:

It will be compared an inviscid model and viscous model to have a general idea about the main differences between both models. They would be loaded until the same level of damage and analyzes the Stress/strain relationship after the loading. It was supposed a viscosity parameter of 0.4 to define the viscous model and the others parameter have been the same for both models.

The main difference can be seen in the plot of the elastic surface. The point can be outside the surface in the viscous model in contrast to the inviscid model, where the point is always inside the surface. The damping dissipates part of the energy introduced into the system, which allow to the point stay out of the surface. Both system have been loaded until it was damaged approximately 50 %. In the inviscid model 500Pa were needed. However, the viscous model needed 680Pa to reach to the same level of damage. This difference is due to the damping. Part of the introduced energy was dissipated as kinetic energy in the viscous model and not all the applied energy is used to reduce the stiffness.

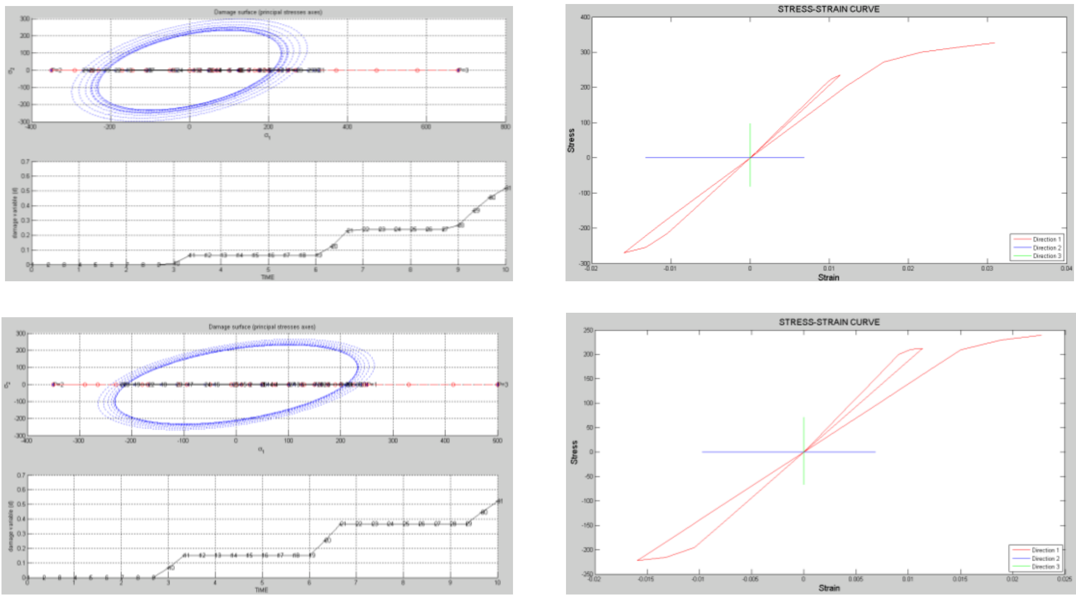


Figure 2. 1 Viscous model (top) and inviscid model (bottom)

After having a general view of the both models, the cases asked in the section E will be show in the following paragraphs:

- **Different viscosity parameters (η):**

The point is loaded uniaxially and the results have been plotted using different viscous model. The elastic range is very similar in all cases, because the viscous coefficient does not start applying until the elastic yield stress is overcomen. The peak stress in the elastic range seems to be different due to the viscosity parameter. Although, the point is outside of the elastic surface due to the viscous coefficient, it still continues in the elastic range. This reason justifies that for greater viscous coefficient the elastic range seems to be longer.

On the other hand, between the plastic range are more differences. As it has been explained before, part of the energy is dissipated by the damping so the higher the damping is, more energy is dissipated. In fact, if it is drawn vertical line for the strain 0.002, it can be seen that the needed stress to deform increase with the viscous coefficient.

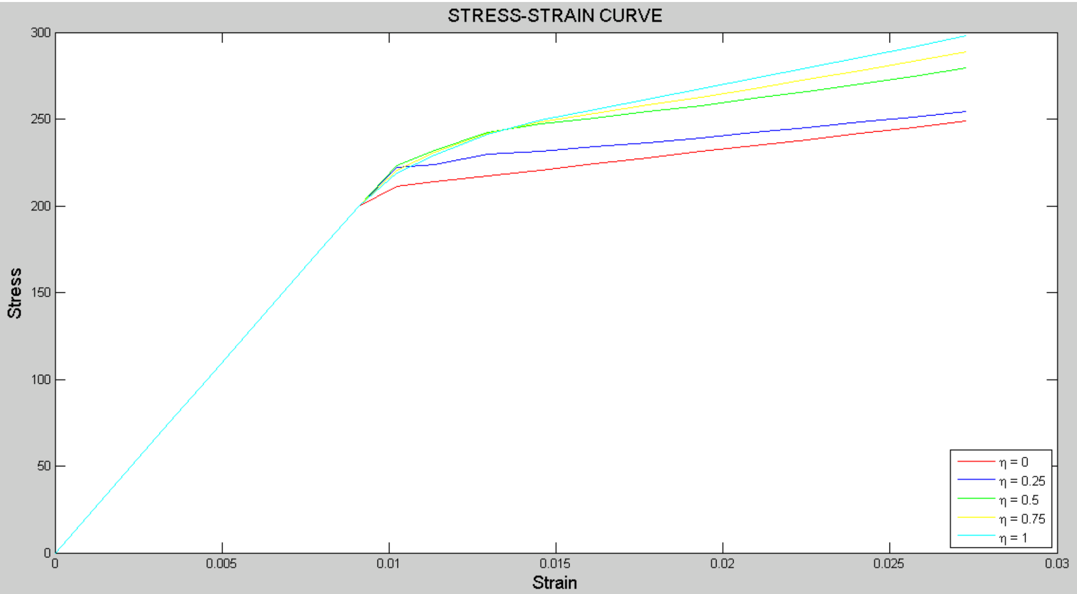


Figure 2. 2 Viscocity coefficient

- **Different strain rate values (dℇ)**

It has been varied the variable istep from 10-100. Logically, the greater the number of steps are, the greater is the computational time that the software need. The solution is more accurate when the time steps are smaller. However, there will be a moment that the reduction of time step will not suppose any significant improvement in the solution. The time step should be defined taking in care the accuracy of the solution and the required computational time.

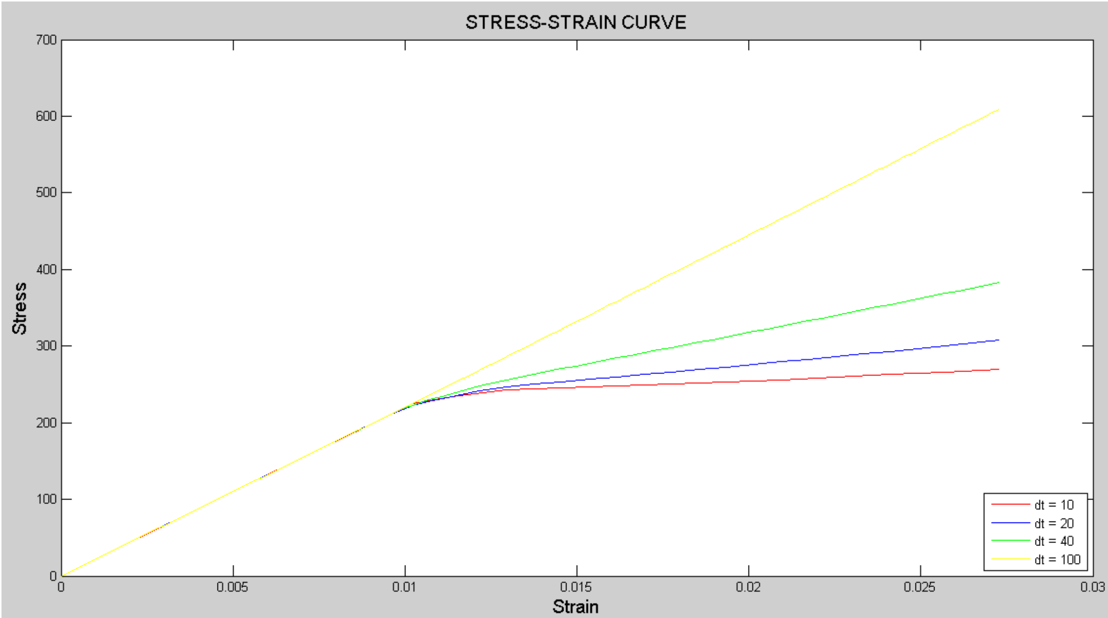


Figure 2. 3 Time-step

- **Different α values: α = ¼, ½, ¾, 1**

The most accurate curve is the one for α = 0.5. Therefore, the closest point to this curve will represent better what happens. This plot shows that for greater α the plastic curve is more inclined than for lower ones. The variation of α affects to the values of C_{11}. Lower values of α will get conservative solutions in contrast, values higher than 0.5 will show results higher than the ideal value.

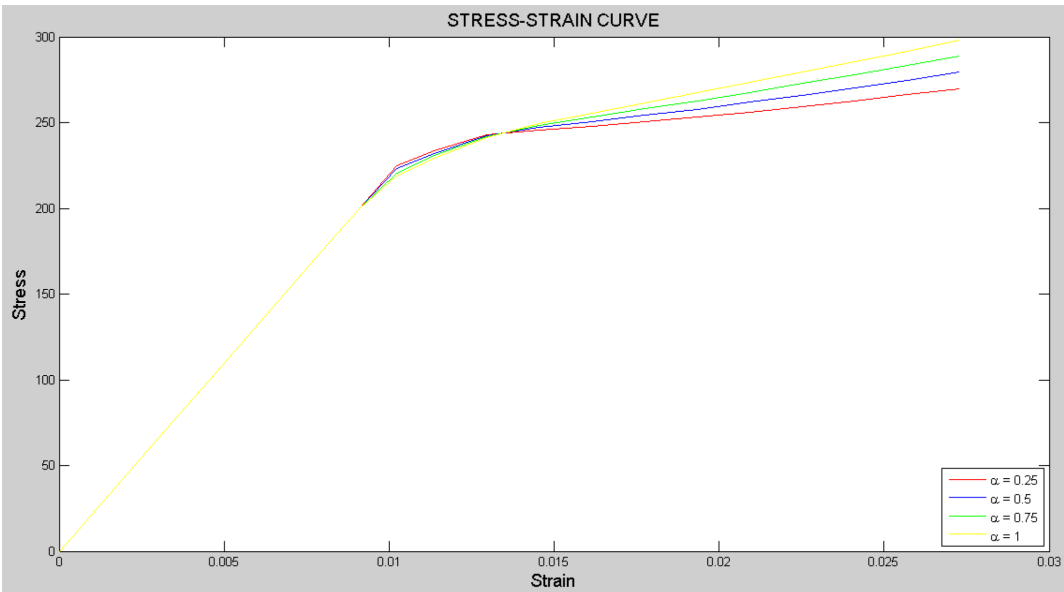


Figure 2. 4 Alpha coefficient

ANNEX

- **main_nointeractive.m**

```
clc
close all
clear
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% Program for modelling damage model
% (Elemental gauss point level)
% -----------------
% Developed by J.A. Hdez Ortega
% 20-May-2007, Universidad PolitÃ©cnica de CataluÃ±a
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%profile on

% ------------------------
% ****************
% INPUTS
% ****************

% YOUNG's MODULUS
% ---------------
YOUNG_M = 20000 ;
% Poisson's coefficient
% -----------------------
POISSON = 0.3 ;
% Hardening/softening modulus
% ---------------------------
HARDSOFT_MOD = 0.1 ;
% Yield stress
% ------------
YIELD_STRESS = 200 ;
% Problem type  TP = {'PLANE STRESS','PLANE STRAIN','3D'}
% ------------------------ = 1             =2          =3
% ------------
ntype= 2 ;
% Model    PTC = {'SYMMETRIC','TENSION','NON-SYMMETRIC'} ;
%                    = 1           = 2           = 3
% ---------------------------------------------------
MDtype =3;
% Ratio compression strength / tension strength
% -------------------------------------------
n = 3 ;
% SOFTENING/HARDENING TYPE
% ------------------------
HARDTYPE = 'LINEAR' ; %{LINEAR,EXPONENTIAL}
% VISCOUS/INVISCID
% ------------------------
VISCOUS = 'NO' ;
% Viscous coefficient ----
% ------------------------
```

```
eta = 0.3 ;
% TimeTotal (initial = 0) ----
% -------------------------
TimeTotal = 10 ; ;
% Integration coefficient ALPHA
% -------------------------
ALPHA_COEFF = 0.5 ;
% Points ---------------------------
% -----------------------------------
nloadstates = 3 ;
SIGMAP = zeros(nloadstates,2) ;
SIGMAP(1,:) =[250 0];
SIGMAP(2,:) =[-350 -0];
SIGMAP(3,:) =[500 0];
% Number of time increments for each load state
% ---------------------------------------
istep = 10*ones(1,nloadstates) ;

% VARIABLES TO PLOT
vpx = 'TIME' ; % AVAILABLE OPTIONS: 'STRAIN_1', 'STRAIN_2'
%                    '|STRAIN_1|', '|STRAIN_2|'
% 'norm(STRAIN)', 'TIME'
vpy = 'damage variable (d)'             % AVAILABLE OPTIONS:
'STRESS_1', 'STRESS_2'
%                    '|STRESS_1|', '|STRESS_2|'
% 'norm(STRESS)', 'TIME', 'DAMAGE VAR.','hardening variable
(q)','damage variable (d)'
%  'internal variable (r)'

%  3) LABELPLOT{ivar}               --> Cell array with the
label string for
%                                    variables of "varplot"
%
LABELPLOT = {'hardening variable (q)','internal variable
(r)','damage variable (d)'};

%%%%%%%%%%%%%%%%%%55 END INPUTS
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% Plot Initial Damage Surface and effective stress path
strain_history =
PlotIniSurf(YOUNG_M,POISSON,YIELD_STRESS,SIGMAP,ntype,MDtype,
n,istep);



E       = YOUNG_M      ;
nu      = POISSON      ;
sigma_u = YIELD_STRESS ;



switch  HARDTYPE
    case 'LINEAR'
```

```
        hard_type = 0  ;
    otherwise
        hard_type = 1  ;
end
switch  VISCOUS
    case 'YES'
        viscpr = 1      ;
    otherwise
        viscpr = 0      ;
end


Eprop   = [E nu HARDSOFT_MOD sigma_u hard_type viscpr eta
ALPHA_COEFF]                ;

% DAMAGE MODEL
% ------------
[sigma_v,vartoplot,LABELPLOT_out,TIMEVECTOR]=damage_main(Epro
p,ntype,istep,strain_history,MDtype,n,TimeTotal);

try; LABELPLOT;catch;LABELPLOT = LABELPLOT_out ; end ;

% PLOTTING
% --------

ncolores = 3 ;
colores =  ColoresMatrix(ncolores);
markers = MarkerMatrix(ncolores) ;
hplotLLL = [] ;

for i = 2:length(sigma_v)
    stress_eig  = sigma_v{i} ; %eigs(sigma_v{i}) ;
    tstress_eig = sigma_v{i-1}; %eigs(sigma_v{i-1}) ;
    hplotLLL(end+1) = plot([tstress_eig(1,1) stress_eig(1,1)
],[tstress_eig(2,2)
stress_eig(2,2)],'LineWidth',2,'color',colores(1,:),'Marker',
markers{1},'MarkerSize',2);
    plot(stress_eig(1,1),stress_eig(2,2),'bx')
    text(stress_eig(1,1),stress_eig(2,2),num2str(i))


    % SURFACES
    % -----
end

% % SURFACES
% % -----
% if(aux_var(1)>0)
%     hplotSURF(i) = dibujar_criterio_dano1(ce, nu,
hvar_n(6), 'r:',MDtype,n );
%     set(hplotSURF(i),'Color',[0 0 1],'LineWidth',1);
% end
```

```
DATA.sigma_v     = sigma_v     ;
DATA.vartoplot   = vartoplot   ;
DATA.LABELPLOT   = LABELPLOT   ;
DATA.TIMEVECTOR = TIMEVECTOR  ;
DATA.strain = strain_history ;

plotcurvesNEW(DATA,vpx,vpy,LABELPLOT,vartoplot) ;

% STRESS/STRAIN CURVE
sigstr1(:,1)=strain_history(:,1);
sigstr2(:,1)=strain_history(:,2);
sigstr3(:,1)=strain_history(:,4);
for i=1:size(strain_history,1)
    sigstr1(i,2)=sigma_v{i}(1,1);
    sigstr2(i,2)=sigma_v{i}(2,2);
    sigstr3(i,2)=sigma_v{i}(3,3);
end

figure(2)
plot(sigstr1(:,1),sigstr1(:,2),'r',sigstr2(:,1),sigstr2(:,2),
'b',sigstr3(:,1),sigstr3(:,2),'g')
title ('STRESS-STRAIN CURVE','FontSize',14)
xlabel('Strain','FontSize',14)
ylabel('Stress','FontSize',14)
legend('Direction 1','Direction 2','Direction 3','Location')
```

- **dibujar_criterio_dano1.m**

```
function hplot =
dibujar_criterio_dano1(ce,nu,q,tipo_linea,MDtype,n)
%*********************************************************************
%*      PLOT DAMAGE SURFACE CRITERIUM: ISOTROPIC MODEL
%*
%*      function [ce] = tensor_elastico (Eprop, ntype)
%*
%*      INPUTS
%*
%*       Eprop(4)    vector de propiedades de material
%*       Eprop(1)=  E------>modulo de Young
%*       Eprop(2)=  nu----->modulo de Poisson
%*       Eprop(3)=  H----->modulo de Softening/hard.
%*       Eprop(4)=  sigma_u----->tensiï¿½n ï¿½ltima
%*       ntype
%*           ntype=1  plane stress
%*           ntype=2  plane strain
%*           ntype=3  3D
%*       ce(4,4)     Constitutive elastic tensor  (PLANE S)
%*       ce(6,6)                                  ( 3D)
%*
%*********************************************************************
*******************
%*        Inverse ce
%*
ce_inv=inv(ce);
c11=ce_inv(1,1);
c22=ce_inv(2,2);
c12=ce_inv(1,2);
c21=c12;
c14=ce_inv(1,4);
c24=ce_inv(2,4);

%*********************************************************************
********************
% POLAR COORDINATES
if MDtype==1
    tetha=[0:0.01:2*pi];

%*********************************************************************
********************
    %* RADIUS
    D=size(tetha);                       %*  Range
    m1=cos(tetha);                       %*
    m2=sin(tetha);                       %*
    Contador=D(1,2);                     %*


    radio = zeros(1,Contador) ;
    s1    = zeros(1,Contador) ;
    s2    = zeros(1,Contador) ;

    for i=1:Contador
        radio(i)= q/sqrt([m1(i) m2(i) 0
nu*(m1(i)+m2(i))]*ce_inv*[m1(i) m2(i) 0 ...
            nu*(m1(i)+m2(i))]');
```

```
        s1(i)=radio(i)*m1(i);
        s2(i)=radio(i)*m2(i);

    end
    hplot =plot(s1,s2,tipo_linea);
```

```
elseif MDtype==2
%   *******************************************************
%     tetha1=[0:0.01:pi]
%     tetha2=[3*pi/2:0.01:2*pi]
%     tetha=[tetha2 tetha1];
   tetha1=[0:0.01:pi/2];
   tetha2=[pi/2+0.01:0.01:pi];
   tetha3=[3*pi/2:0.01:2*pi];
   tetha=[tetha3 tetha1 tetha2];

%******************************************************************
    %* RADIUS
    D=size(tetha);                       %*  Range
    m1=cos(tetha);                       %*
    m2=sin(tetha);                       %*
    Contador=D(1,2);                     %*


    radio = zeros(1,Contador) ;
    s1    = zeros(1,Contador) ;
    s2    = zeros(1,Contador) ;
%     for i=1:size(tetha,2)
%         radio(i)= q/sqrt([m1(i) m2(i) 0
nu*(m1(i)+m2(i))]*ce_inv*[m1(i) m2(i) 0 ...
%             nu*(m1(i)+m2(i))]');
%
%         s1(i)=radio(i)*m1(i);
%         s2(i)=radio(i)*m2(i);
%
%     end

    for i=size(tetha3,2)+1:(size(tetha3,2)+size(tetha1,2))
        radio(i)= q/sqrt([m1(i) m2(i) 0
nu*(m1(i)+m2(i))]*ce_inv*[m1(i) m2(i) 0 ...
            nu*(m1(i)+m2(i))]');

        s1(i)=radio(i)*m1(i);
        s2(i)=radio(i)*m2(i);

    end

    for
i=(size(tetha3,2)+size(tetha1,2))+1:(size(tetha3,2)+size(tetha1,2))
+size(tetha2,2)
        radio(i)= q/sqrt([m1(i) m2(i) 0
nu*(m1(i)+m2(i))]*ce_inv*[m1(i) m2(i) 0 ...
            nu*(m1(i)+m2(i))]');

        s1(i)=radio(i)*m1(i);
```

```
        s2(i)=s2(i-1);

    end
    for i=1:size(tetha3,2)
        radio(i)= q/sqrt([m1(i) m2(i) 0
nu*(m1(i)+m2(i))]*ce_inv*[m1(i) m2(i) 0 ...
            nu*(m1(i)+m2(i))]');

        s1(i)=s1(size(tetha3,2)+1);
        s2(i)=radio(i)*m2(i);

    end

    hplot =plot(s1,s2,tipo_linea);

elseif MDtype==3
tetha1=[0:0.01:pi/2];
tetha2=[pi:0.01:3*pi/2];
tetha=[tetha1 tetha2];

%******************************************************************
    %* RADIUS
    D=size(tetha);                      %*  Range
    m1=cos(tetha);                      %*
    m2=sin(tetha);                      %*
    Contador=D(1,2);                    %*


    radio = zeros(1,Contador) ;
    s1    = zeros(1,Contador) ;
    s2    = zeros(1,Contador) ;

    for i=1:size(tetha1,2)
        radio(i)= q/sqrt([m1(i) m2(i) 0
nu*(m1(i)+m2(i))]*ce_inv*[m1(i) m2(i) 0 ...
            nu*(m1(i)+m2(i))]');

        s1(i)=radio(i)*m1(i);
        s2(i)=radio(i)*m2(i);
    end

    for i=size(tetha1,2)+1:(size(tetha1,2)+size(tetha2,2))
        radio(i)= (q*n)/sqrt([m1(i) m2(i) 0
nu*(m1(i)+m2(i))]*ce_inv*[m1(i) m2(i) 0 ...
            nu*(m1(i)+m2(i))]');

        s1(i)=radio(i)*m1(i);
        s2(i)=radio(i)*m2(i);
    end

    s1=[s1 s1(1)];
    s2=[s2 s2(1)];
    hplot =plot(s1,s2,tipo_linea)
end
%*****************************************************************
return
```

- **damage_main.com**

```
function
[sigma_v,vartoplot,LABELPLOT,TIMEVECTOR]=damage_main(Eprop,nt
ype,istep,strain,MDtype,n,TimeTotal)
global hplotSURF
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%
% CONTINUUM DAMAGE MODEL
% ----------------------
% Given the almansi strain evolution
("strain(totalstep,mstrain)") and a set of
% parameters and properties, it returns the evolution of the
cauchy stress and other  variables
% that are listed below.
%
% INPUTS
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
% ------------------------------------------------------------
-----
% Eprop(1)  = Young's modulus  (E)
% Eprop(2)  = Poisson's coefficient (nu)
% Eprop(3)  = Hardening(+)/Softening(-) modulus (H)
% Eprop(4)  = Yield stress (sigma_y)
% Eprop(5)  = Type of Hardening/Softening law  (hard_type)
%             0 --> LINEAR
%             1 --> Exponential
% Eprop(6)  = Rate behavior (viscpr)
%             0 --> Rate-independent (inviscid)
%             1 --> Rate-dependent   (viscous)
%
% Eprop(7)  = Viscosity coefficient (eta)  (dummy if inviscid)
% Eprop(8)  = ALPHA coefficient (for time integration),
(ALPHA)
%              0<=ALPHA<=1 , ALPHA = 1.0 --> Implicit
%                            ALPHA = 0.0 --> Explicit
%             (dummy if inviscid)
%
% ntype     = PROBLEM TYPE
%             1 : plane stress
%             2 : plane strain
%             3 : 3D
%
% istep = steps for each load state (istep1,istep2,istep3)
%
% strain(i,j) = j-th component of the linearized strain
vector at the i-th
%                step, i = 1:totalstep+1
%
% MDtype      = Damage surface criterion %
%             1 : SYMMETRIC
%             2 : ONLY-TENSION
%             3 : NON-SYMMETRIC
%
```

```
%
% n           = Ratio compression/tension strength (dummy if
MDtype is different from 3)
%
% TimeTotal  = Interval length
%
%  OUTPUTS
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
%  -------------------------------------------------------------
---------
%  1) sigma_v{itime}(icomp,jcomp)  --> Component
(icomp,jcomp) of the cauchy
%                                    stress tensor at step
"itime"
%                                    REMARK: sigma_v is a type
of
%                                    variable called "cell
array".
%
%
%  2) vartoplot{itime}              --> Cell array containing
variables one wishes to plot
%                                     ------------------------
--------------
%   vartoplot{itime}(1) =   Hardening variable (q)
%   vartoplot{itime}(2) =   Internal variable (r)%

%
%  3) LABELPLOT{ivar}                --> Cell array with the
label string for
%                                     variables of "varplot"
%
%          LABELPLOT{1} => 'hardening variable (q)'
%          LABELPLOT{2} => 'internal variable'
%
%
%  4) TIME VECTOR  - >
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% SET LABEL OF "vartoplot" variables  (it may be defined also
outside this function)
% ----------------------------------
 LABELPLOT = {'hardening variable (q)','internal variable'};

E=Eprop(1);
nu = Eprop(2) ;
viscpr = Eprop(6) ;
sigma_u = Eprop(4);


if ntype == 1
    menu('PLANE STRESS has not been implemented yet','STOP');
```

```
    error('OPTION NOT AVAILABLE')
elseif ntype == 3
    menu('3-DIMENSIONAL PROBLEM has not been implemented
yet','STOP');
    error('OPTION NOT AVAILABLE')
else
    mstrain = 4    ;
    mhist   = 6    ;
end

if viscpr == 1
    % VISCOUS MODEL
    eta = Eprop(7) ;
    ALPHA_COEFF = Eprop(8) ;
else
    % INVISCID MODEL
    eta=0;
    ALPHA_COEFF=1;
    Eprop(7)=eta;
    Eprop(8)=ALPHA_COEFF;
end

totalstep = sum(istep) ;

% INITIALIZING GLOBAL CELL ARRAYS
% --------------------------------
sigma_v = cell(totalstep+1,1) ;
TIMEVECTOR = zeros(totalstep+1,1) ;
delta_t = TimeTotal./istep/length(istep) ;

% Elastic constitutive tensor
% ----------------------------
[ce]     = tensor_elastico1 (Eprop, ntype);
% Initz.
% -----
% Strain vector
% -------------
eps_n1   = zeros(mstrain,1);
% Historic variables
% hvar_n(1:4) --> empty
% hvar_n(5) = q --> Hardening variable
% hvar_n(6) = r --> Internal variable
hvar_n   = zeros(mhist,1)   ;

% INITIALIZING  (i = 1) !!!!
% ***********i*
i = 1 ;
r0 = sigma_u/sqrt(E);
hvar_n(5) = r0; % r_n
hvar_n(6) = r0; % q_n
eps_n1 = strain(i,:) ;
sigma_n1 =ce*eps_n1'; % Elastic
sigma_v{i} = [sigma_n1(1)  sigma_n1(3)  0;sigma_n1(3)
sigma_n1(2) 0 ; 0 0  sigma_n1(4)];
```

```
nplot = 3 ;
vartoplot = cell(1,totalstep+1) ;
vartoplot{i}(1) = hvar_n(6) ; % Hardening variable (q)
vartoplot{i}(2) = hvar_n(5) ; % Internal variable (r)
vartoplot{i}(3) = 1-hvar_n(6)/hvar_n(5)  ; %  Damage variable
(d)

for  iload = 1:length(istep)
    % Load states
    for iloc = 1:istep(iload)
        i = i + 1 ;
        TIMEVECTOR(i) = TIMEVECTOR(i-1)+ delta_t(iload) ;
        % Total strain at step "i"
        % ------------------------
        eps_n1 = strain(i,:) ;

        eps_n = strain(i-1,:) ;

%*************************************************************
        %*      DAMAGE MODEL
        %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        [sigma_n1,hvar_n,aux_var] =
rmap_dano1(eps_n1,hvar_n,Eprop,ce,MDtype,n,eps_n,delta_t(iloa
d));
        % PLOTTING DAMAGE SURFACE
        if(aux_var(1)>0)
            hplotSURF(i) = dibujar_criterio_dano1(ce, nu,
hvar_n(6), 'r:',MDtype,n );
            set(hplotSURF(i),'Color',[0 0 1],'LineWidth',1)
;
        end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%*************************************************************
        % GLOBAL VARIABLES
        % ***************
        % Stress
        % ------
        m_sigma=[sigma_n1(1)  sigma_n1(3) 0;sigma_n1(3)
sigma_n1(2) 0 ; 0 0  sigma_n1(4)];
        sigma_v{i} =  m_sigma ;

        % VARIABLES TO PLOT (set label on cell array
LABELPLOT)
        % ----------------
        vartoplot{i}(1) = hvar_n(6) ; % Hardening variable
(q)
        vartoplot{i}(2) = hvar_n(5) ; % Internal variable (r)
        vartoplot{i}(3) = 1-hvar_n(6)/hvar_n(5)  ; %  Damage
variable (d)
    end
```

```
end
```

- **rmap_dano1.m**

```
function [sigma_n1,hvar_n1,aux_var] = rmap_dano1
(eps_n1,hvar_n,Eprop,ce,MDtype,n,eps_n,delta_t)

%**************************************************************
%*
%*Integration Algorithm for a isotropic damage model
%*
%*
%*[sigma_n1,hvar_n1,aux_var] = rmap_dano1
(eps_n1,hvar_n,Eprop,ce)
%*
*
%* INPUTS
%*      eps_n1(4)   strain (almansi)    step n+1
%*      vector R4    (exx eyy exy ezz)
%*      hvar_n(6)   internal variables , step n
%*      hvar_n(1:4) (empty)
%*      hvar_n(5) = r  ; hvar_n(6)=q
%*      Eprop(:)    Material parameters
%*      ce(4,4)     Constitutive elastic tensor
%*
%* OUTPUTS:
%*      sigma_n1(4) Cauchy stress  , step n+1
%*      hvar_n(6)   Internal variables , step n+1
%*      aux_var(3)  Auxiliar variables for computing const.
tangent tensor
%**************************************************************

hvar_n1 = hvar_n;
r_n     = hvar_n(5);
q_n     = hvar_n(6);
E       = Eprop(1);
nu      = Eprop(2);
H       = Eprop(3);
sigma_u = Eprop(4);
hard_type = Eprop(5) ;
sigma_n1 =ce*eps_n1'; % Elastic
sigma_v = [sigma_n1(1)  sigma_n1(3) 0;sigma_n1(3) sigma_n1(2)
0 ; 0 0  sigma_n1(4)];
ALFACOEFF=Eprop(8);
eta=Eprop(7);
%**************************************************************
%*       initializing
%*
 r0 = sigma_u/sqrt(E);
 zero_q=1.d-6*r0;
 q_inf=3*r0/2;
% if(r_n<=0.d0)
%     r_n=r0;
%     q_n=r0;
% end
```

```
%*****************************************************************
*************************
%*       Damage surface
%*
[rtrial_n1] = Modelos_de_dano1 (MDtype,ce,eps_n1,n,q_n,r_n);
rtrial_n=sqrt(eps_n*ce*eps_n')
rtrial=(1-ALFACOEFF)*rtrial_n+ALFACOEFF*rtrial_n1;
%*****************************************************************
*************************
%*   Ver el Estado de Carga
%*
%*   --------->    fload=0 : elastic unload
%*
%*   --------->    fload=1 : damage (compute algorithmic
constitutive tensor)          %*
fload=0;

if(rtrial > r_n)
    %*   Loading

    fload=1;
    delta_r=rtrial-r_n;
    r_n1=((eta-delta_t*(1-
ALFACOEFF))/(eta+ALFACOEFF*delta_t))*r_n+(delta_t/(eta+ALFACO
EFF*delta_t))*rtrial;

    if hard_type == 0
        %  Linear
        q_n1= q_n+ H*delta_r;
    else
        %  EXPONENCIAL
        q_n1=q_inf-(q_inf-q_n)*exp(abs(H)*(1-(rtrial/q_n)));
    end

    if(q_n1<zero_q)
        q_n1=zero_q;
    end

else

    %*     Elastic load/unload
    fload=0;
    r_n1= r_n  ;
    q_n1= q_n  ;


end
% Damage variable
% ---------------
dano_n1   = 1.d0-(q_n1/r_n1);
%  Computing stress
%  ****************
sigma_n1  =(1.d0-dano_n1)*ce*eps_n1';
%hold on
```

```
%plot(sigma_n1(1),sigma_n1(2),'bx')
%**************************************************************
************************
%* Updating historic variables
%*
%  hvar_n1(1:4)  = eps_n1p;
hvar_n1(5)= r_n1 ;
hvar_n1(6)= q_n1 ;
%*
%**************************************************************
%* Auxiliar variables
%*
aux_var(1) = fload;
aux_var(2) = q_n1/r_n1;
%*aux_var(3) = (q_n1-H*r_n1)/r_n1^3;
%**************************************************************
```

- **Modelos_de _dano1.m**

```
function [rtrial] = Modelos_de_dano1 (MDtype,ce,eps_n1,n,q_n,r_n)
%*******************************************************************
%*   Defining damage criterion surface
%*
%*    MDtype=  1      : SYMMETRIC
%*    MDtype=  2      : ONLY TENSION
%*    MDtype=  3      : NON-SYMMETRIC
%*
%* OUTPUT:
%*    rtrial
%*******************************************************************
sigma_n1 =ce*eps_n1'; % Elastic
sigma_v = [sigma_n1(1)  sigma_n1(3) 0;sigma_n1(3) sigma_n1(2) 0 ; 0
0  sigma_n1(4)];

%*******************************************************************

if (MDtype==1)      %* Symmetric
rtrial= sqrt(eps_n1*ce*eps_n1')                       ;

elseif (MDtype==2)  %* Only tension
    d_n=1-q_n/r_n;
    % Diagonalization: Principal stress (eigenvalues)
    eigen_val=rot90(rot90(sort(eig(sigma_v))));
    stress_prin=[eigen_val(1) 0 0;0 eigen_val(2) 0 ;0 0
eigen_val(3)];
    % Principal directions (eigenvectors):unit vector
    [dir_prin,sig_prin]=eig(sigma_v);
    % Apply McAuley bracket:
    stress_Mca=sig_prin;
    sig_pos=zeros(3,3);
    if sig_prin(1,1)<0
        stress_Mca(1,1)=0;
        sig_pos(1,1)=0;
    else
       sig_pos=sig_pos+sig_prin(1,1)*dir_prin(:,1)*dir_prin(:,1)';
    end

    if sig_prin(2,2)<0
        stress_Mca(2,2)=0;
        sig_pos(2,2)=0;
    else
       sig_pos=sig_pos+sig_prin(2,2)*dir_prin(:,2)*dir_prin(:,2)';
    end

    if sig_prin(3,3)<0
        stress_Mca(3,3)=0;
        sig_pos(3,3)=0;
    else
       sig_pos=sig_pos+sig_prin(3,3)*dir_prin(:,3)*dir_prin(:,3)';
    end
    sig_pos_n1 = [sig_pos(1,1) sig_pos(2,2) sig_pos(1,2)
sig_pos(3,3)];
    %Norm in the stress space:
    rtrial=(sqrt(sig_pos_n1*(ce^(-1))*sigma_n1))/(1-d_n);
```

```
elseif (MDtype==3)  %*Non-symmetric
    d_n=1-q_n/r_n;
    % Diagonalization: Principal stress
    eigen_val=rot90(rot90(sort(eig(sigma_v))));
    stress_prin=[eigen_val(1) 0 0;0 eigen_val(2) 0 ;0 0
eigen_val(3)];
    % Apply McAuley bracket:
    stress_Mca=stress_prin;
    if stress_Mca(1,1)<0
        stress_Mca(1,1)=0;
    end
    if stress_Mca(2,2)<0
        stress_Mca(2,2)=0;
    end
    if stress_Mca(3,3)<0
        stress_Mca(3,3)=0;
    end
    % Parameter PHI:
    phi=sum(sum(stress_Mca))/sum(sum(stress_prin));

    %Norm in the stress space:
    rtrial=((phi+(1-phi)/n)*sqrt(sigma_n1'*inv(ce)*sigma_n1))/(1-
d_n);

end
%*************************************************************
return
```