Universitat Politecnica de Catalunya CIMNE
Master in Computational Mechanics

# Computational Solid Mechanics

# Assignment 1

# Continuum Damage Models

Submitted By: Kiran Kolhe

30th March 2020

# Contents


- Details. . . . . . . . . . 1
- Part 1 – Rate Independent Models. . . . . . . . . . 1
  - Implementation of Damage Surfaces. . . . . . . . . . 1
  - Effect of Hardening Modulus . . . . . . . . . . 2
  - Implementation of Damage models for different cases . . . . . . . . . . 2
    - ❖ Case 1. . . . . . . . . . 2
    - ❖ Case 2. . . . . . . . . . 6
    - ❖ Case 3. . . . . . . . . . 9
  - Comparison for Viscid and Inviscid case. . . . . . . . . . 12
- Part 2 – Rate Dependent Models. . . . . . . . . . 13
  - Effect of Viscosity Parameter . . . . . . . . . . 13
  - Effect of Strain Rate . . . . . . . . . . 13
  - Effect of Time Integration Parameter . . . . . . . . . . 14
- Conclusion . . . . . . . . . . 15
- Appendix . . . . . . . . . . 16

# Details:

This assignment aims at the numerical integration of constitutive damage models using MATLAB. Three types of damage models have been proposed in this assignment, namely, the Symmetric model, Tension-only model and the Non-symmetric model. For the information, the Symmetric model has already been implemented and after studying the code and observing the results of this model we have been asked to implement tasks for the Tension-only and Non-symmetric damage models.

In the first part, we have to modify the MATLAB code, considering the rate independence criteria (inviscid case) for the Tension-only and Non-symmetric damage models. We have to implement linear and exponential hardening as well as softening for each of these two proposed models. And lastly, we have to run this developed code for three different loading cases obtaining the stress-strain curves for each of them.

In the second part, the scope has been extended to study the impact of rate dependency (viscid case) on the symmetric model. We have been asked to study the effects for different values of viscosity parameter, strain rate and time-integration parameter on the stress-strain curve.

For the loading paths, the constants alpha, beta and gamma have been chosen conveniently so as to have a good visualization of the results.

# Part 1 – Rate Independent Models

- Implementation of Damage Surfaces:

In order to implement the Tension-only and Non-symmetric damage models for the rate independency criteria, we have to make changes in some files. Initially, we have to modify the code 'modelos_de_dano1.m' which is responsible for defining the damage surface for the 3 types of damage models. In these equations, the rtrial function represents $\tau_{n+1}$ which is the norm at time step 'n+1'.

Then, we have to modify the code 'dibujar_criterio_dano1.m' for the Tension-only and Non-symmetric damage models, which is responsible for plotting the damage surface. In this code, a parameter 'radio' is used which stands for the radial distance of the points on the curve from the origin. The damage surface is plotted in the stress space with x-coordinate as $\sigma_1$ and y-coordinate as $\sigma_2$.

After this, for including the Hardening/Softening law we have to modify the 'rmap_dano1.m' file by referring to the formulas given in the slides. The parameter 'H', which is the hardening modulus, has a value equal to hardening coefficient for linear case and should be calculated for exponential case.

Finally, we have to modify the 'damage_main.m' file which gives the evolution of the stress field. After coding these files, the 'main.m' file is run and the following results in the report have been recorded.

The damage surfaces for the Tension-only and Non-symmetric damage models are shown below in **Fig.1** and **Fig.2**.

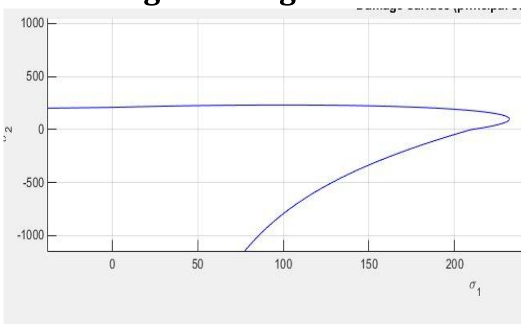


**Fig.1** Damage surface for Tension-only model

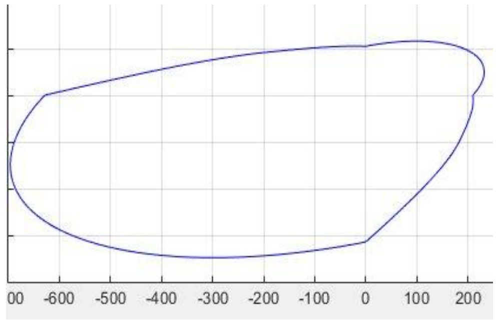


**Fig.2** Damage surface for Non-symmetric model

The elastic region in the Tension-only model shows asymptotic behaviour in the second and fourth quadrant. The Non-symmetric model has a bigger elastic domain in the third quadrant than the first quadrant because damage takes place at higher compressive stresses in the third domain as compared to the tensile stresses in first domain.

- <u>Effect of Hardening Modulus:</u>

The hardening/softening law was implemented for in order to study the linear and exponential response for the models. The results were obtained by plotting the hardening variable 'q' against internal variable 'r' for H=0.5.

In unloading or elastic loading, hardening/softening is not applied to the material hence, the internal variable and hardening variable in consequent time steps are equal. Hardening/softening thus holds true in case of loading only.

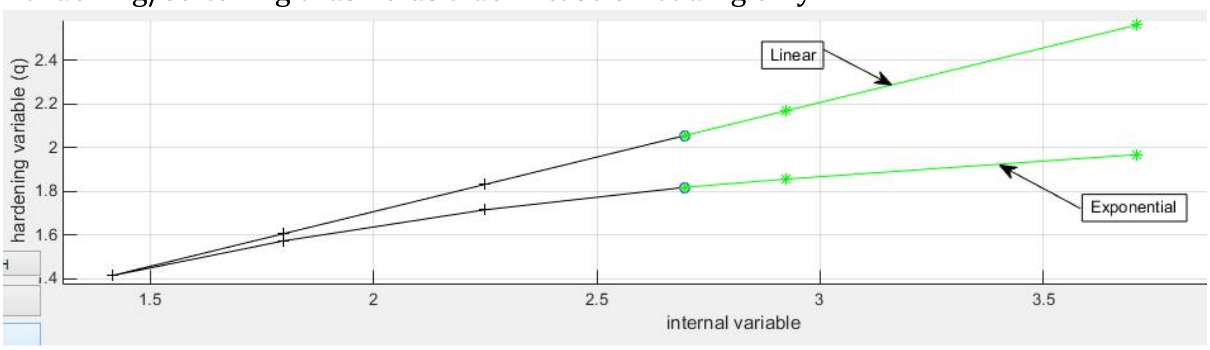


**Fig.3** q vs r plot for comparison of linear and exponential hardening

From **Fig.3**, it can be concluded that the time response in exponential case is slower than that in the linear case.

- <u>Implementation of Damage models for different cases</u>:

The developed codes have been used to assess the correctness of the implementation by applying them to the below 3 cases of loading.

❖ **<u>Case 1</u>**:

Young's Modulus = 20000, Yield Stress = 200, Poisson ratio = 0.3, H = ±0.1

Load path:
$$\Delta\bar{\sigma}_1^{(1)} = 500 \qquad \Delta\bar{\sigma}_2^{(1)} = 0$$
$$\Delta\bar{\sigma}_1^{(2)} = -550 \qquad \Delta\bar{\sigma}_2^{(2)} = 0$$
$$\Delta\bar{\sigma}_1^{(3)} = 700 \qquad \Delta\bar{\sigma}_2^{(3)} = 0$$

- *Tension-only Damage Model*-

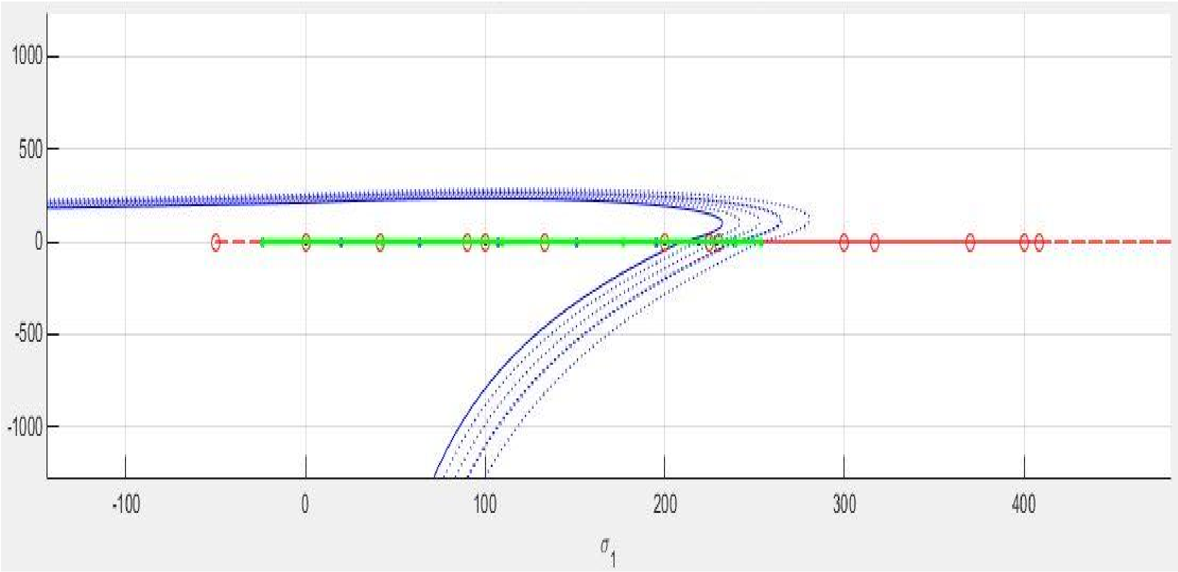


**Fig.4a** Damage Surface (Hardening) for Tension-only model

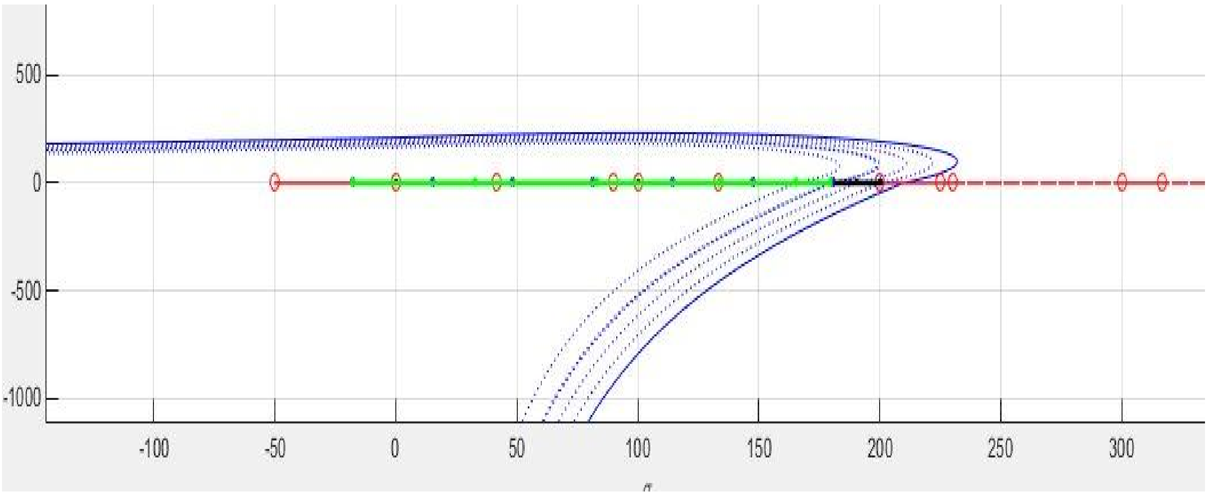


**Fig.4b** Damage Surface (Softening) for Tension-only model

**Fig.4a** and **Fig.4b** represent the damage surfaces for hardening and softening respectively for the uniaxial case of the Tension-only damage model. The dotted blue lines indicate the evolution of the damage surface which expands outwards in case of hardening and reduces inwards in case of softening. The first step is the tensile loading which is then followed by second step of tensile unloading and finally the third step continues after this which is again, tensile loading. As seen in **Fig.4a** the damage surface is crossed in the first step and the material undergoes hardening as it crosses the elastic region.

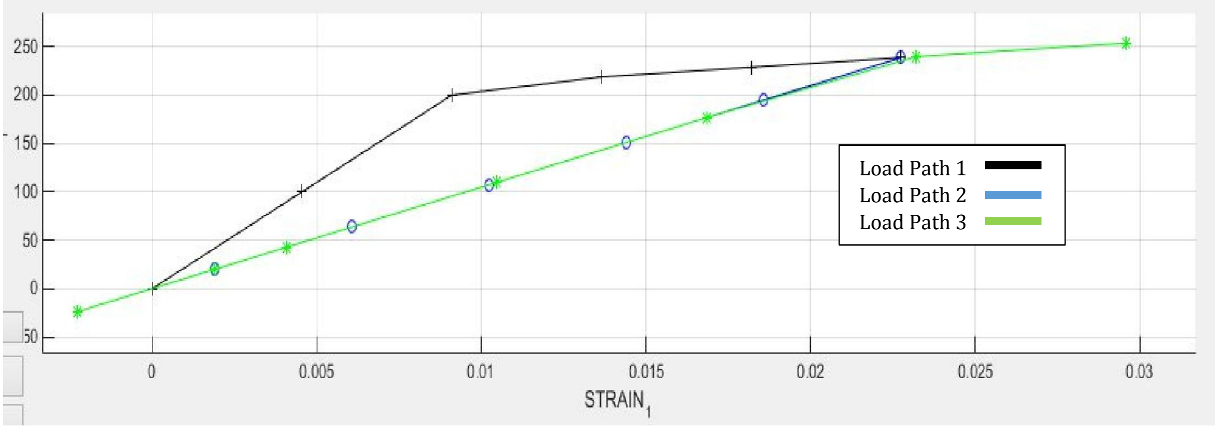


**Fig.5a** Stress-Strain Curve (Hardening) for Tension-only model

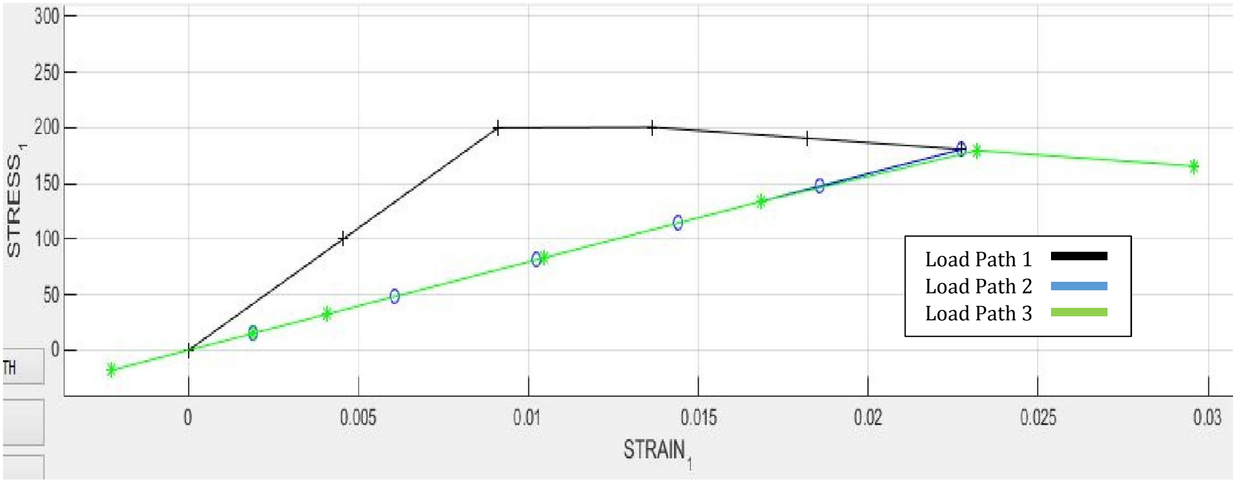


**Fig.5b** Stress-Strain Curve (Softening) for Tension-only model

**Fig.5a** and **Fig.5b** represent the stress-strain curve for hardening (H=0.1) and softening (H= -0.1) respectively for the uniaxial case of the Tension-only damage model. In **Fig.5a** and **Fig.5b** the change in slope of the black line indicates the point where the damage surface is crossed in the first step. The only difference in hardening and softening curves is that while crossing the damage surface, the slope increases in hardening while it decreases in softening.

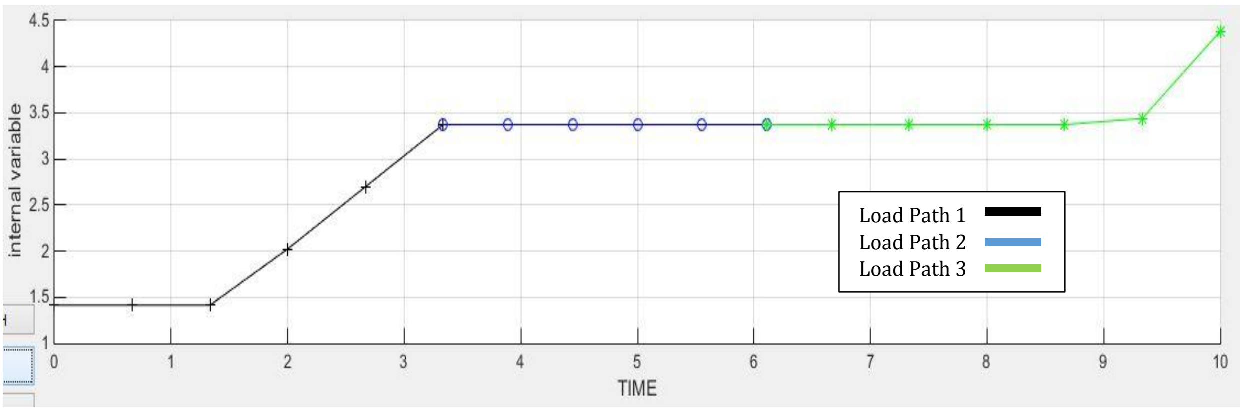


**Fig.6** Internal variable evolution for Tension-only model

And **Fig.6** represents the evolution of internal variable with time for this model. The first step is indicated by black line, which is followed by the second step represented by the blue line, and the green line depicts the third stage of loading. The evolution of internal variable in the second step and the third step takes place with the same slope as in both of these steps the damage surface is not crossed and hence, the elastic regime is maintained.

- *Non-symmetric Damage Model*-

The same results have been obtained for Non-symmetric model.

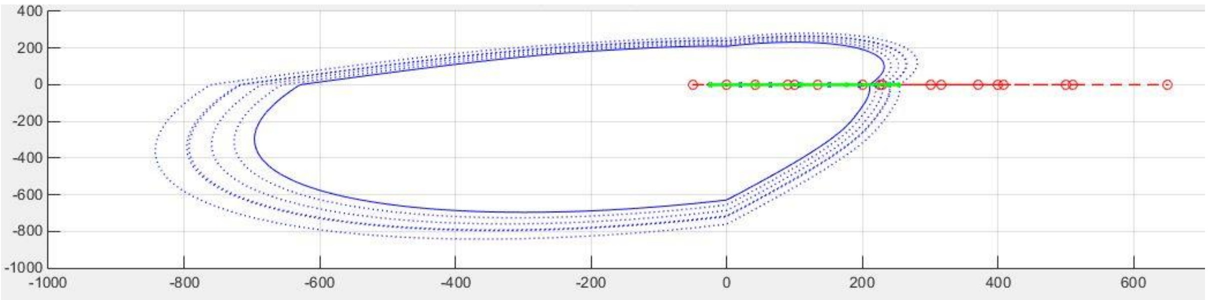

**Fig.7a** Damage Surface (Hardening) for Non-symmetric model

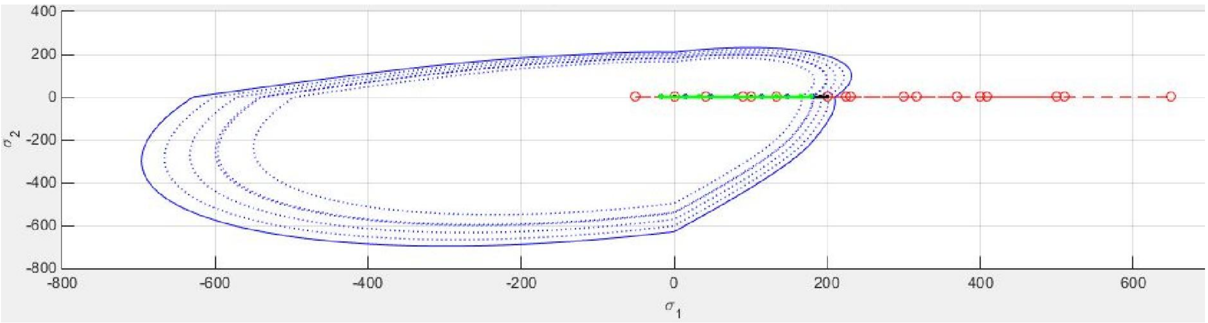

**Fig.7b** Damage Surface (Softening) for Non-symmetric model

**Fig.7a and Fig.7b** represent the damage surfaces for hardening (H=0.1) and softening (H= -0.1) case of the Non-symmetric model. The behavior is observed to be similar to the Tension-only model.

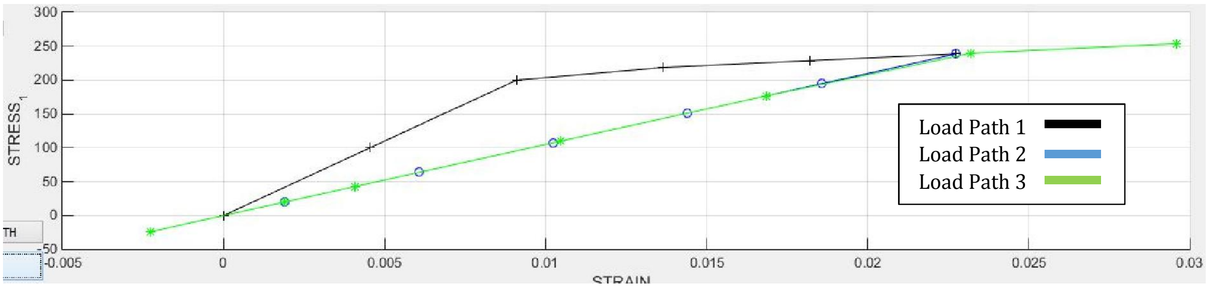


**Fig.8a** Stress-Strain Curve (Hardening) for Non-symmetric model

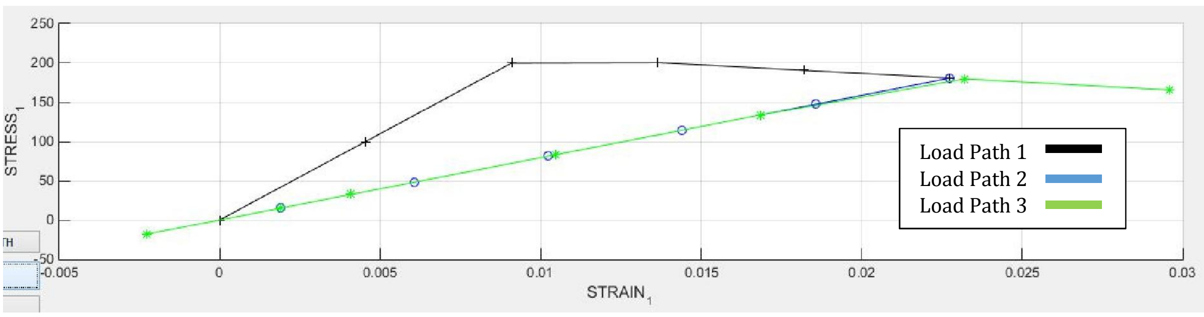


**Fig.8b** Stress-Strain Curve (Softening) for Non-symmetric model

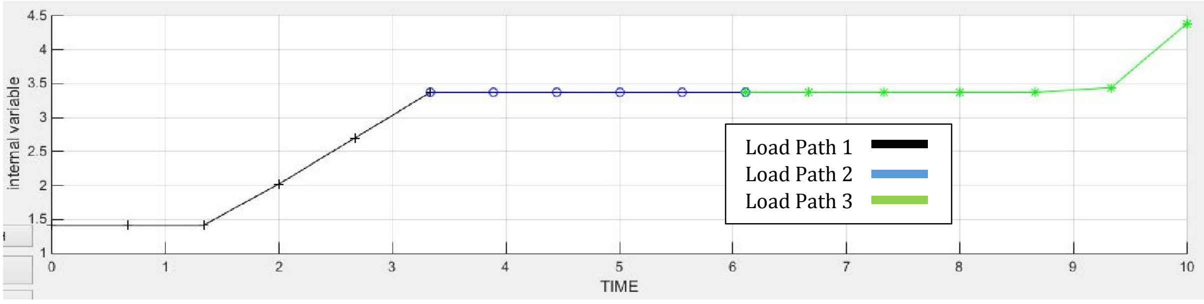


**Fig.9** Internal variable evolution for Non-symmetric model

**Fig.8a** and **Fig.8b** represent the stress-strain curves for hardening and softening case respectively and **Fig.9** shows the evolution of internal variable. The results can be seen somewhat similar with the previous ones.

## ❖ Case 2:

Young's Modulus = 20000, Yield Stress = 200, Poisson ratio = 0.3, H = ±0.1

Load path:

$$\Delta\bar{\sigma}_1^{(1)} = 400 \quad \Delta\bar{\sigma}_2^{(1)} = 0$$
$$\Delta\bar{\sigma}_1^{(2)} = -550 \quad \Delta\bar{\sigma}_2^{(2)} = -550$$
$$\Delta\bar{\sigma}_1^{(3)} = 300 \quad \Delta\bar{\sigma}_2^{(3)} = 300$$

- *Tension-only Damage Model-*

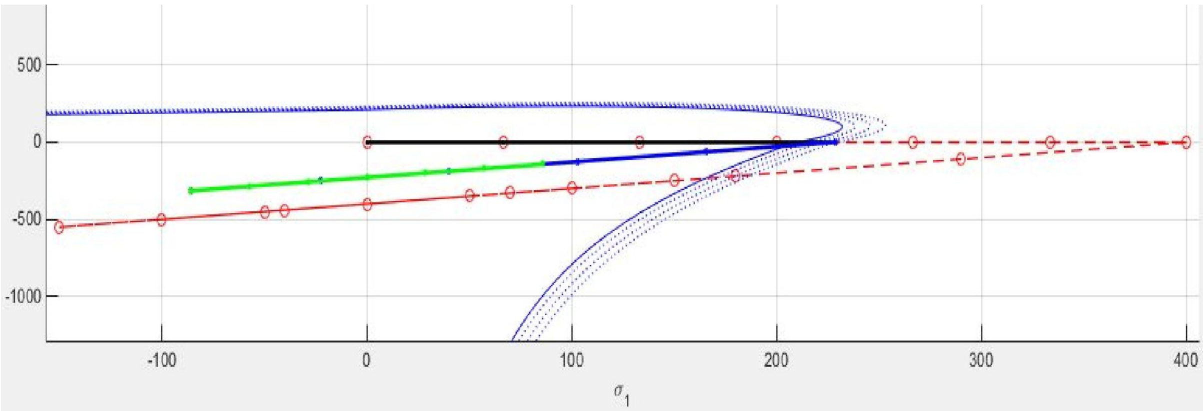

**Fig. 10a** Damage Surface (Hardening) for Tension-only model

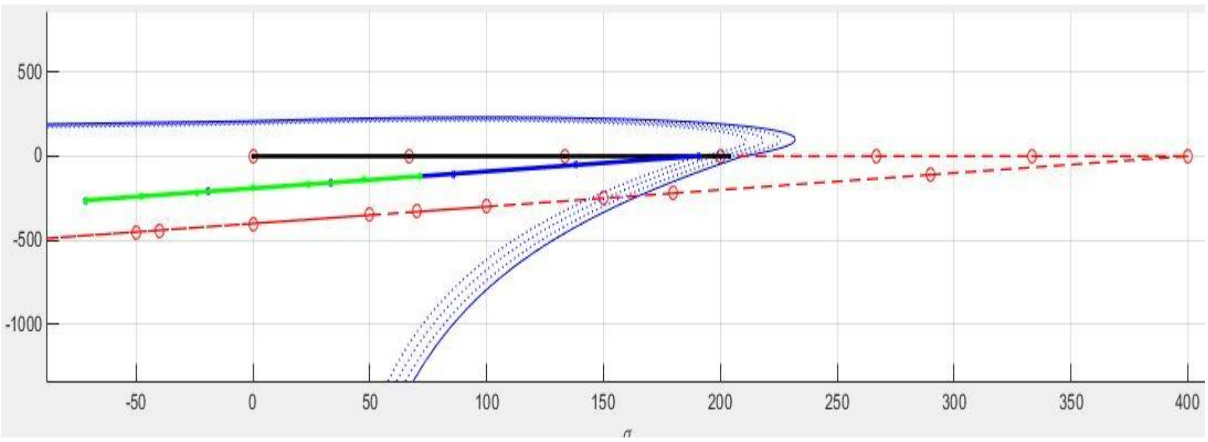

**Fig. 10b** Damage Surface (Softening) for Tension-only model

**Fig.10a** and **10b** represent the damage surfaces for hardening and softening respectively for the Tension-only damage model. The dotted blue lines indicate the evolution of the damage surface which expands in case of hardening and contracts in case of softening. The first step is uniaxial tensile loading which is then followed by second step of biaxial tensile unloading or compression and finally the third step continues after this which is again, biaxial tensile loading. The damage surface is crossed in the first step where the elastic region is crossed and the damage takes place.

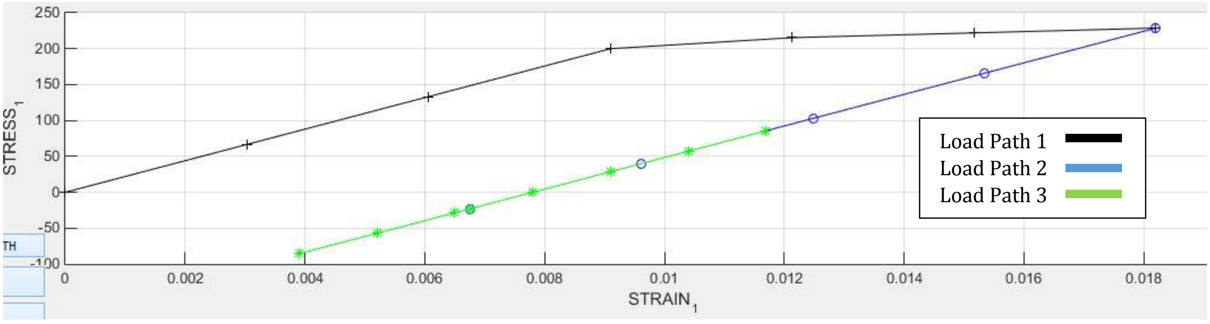


**Fig.11a** Stress-Strain Curve (Hardening) for Tension-only model

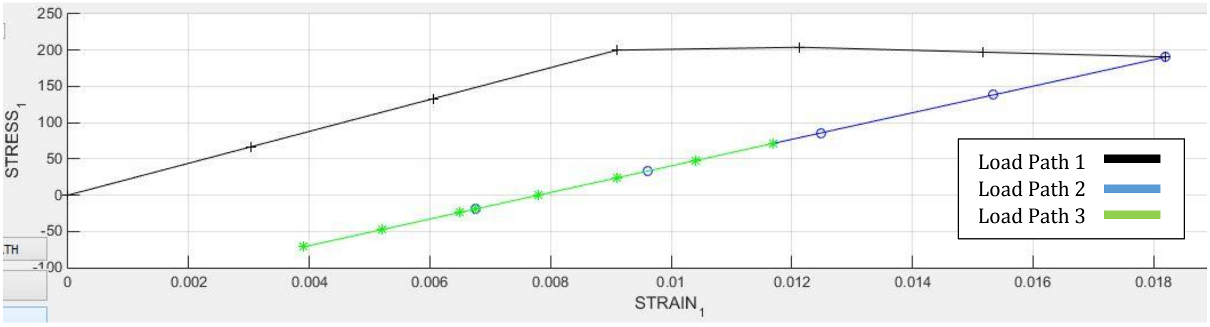


**Fig.11b** Stress-Strain Curve (Softening) for Tension-only model

**Fig.11a** and **11b** represent the stress-strain curve for hardening (H=0.1) and softening (H= -0.1) respectively for the Case 2 of the Tension-only damage model. The first step indicated by the black line is the uniaxial loading which causes damage to the surface. In the second step shown by the blue line, biaxial compression takes place without surpassing the elastic domain because of the property of the Tension-only model. And the third step represented by green line is biaxial tensile loading which again does not cross the elastic regime. The change of slope as indicated in the first case by the black line shows crossing of the damage surface.

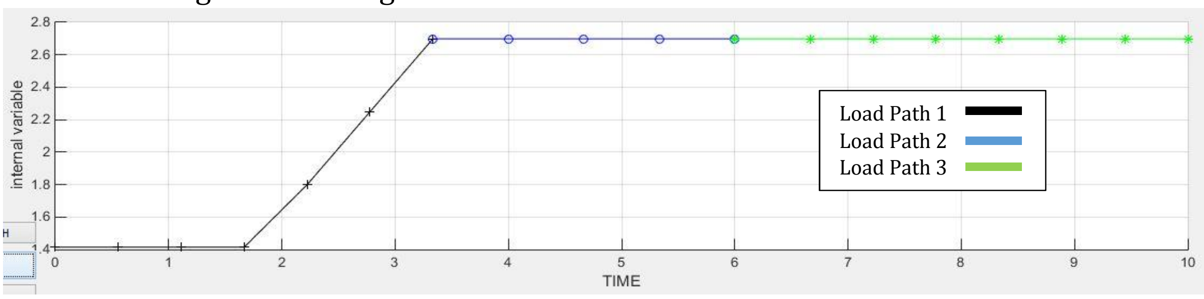


**Fig.12** Internal variable evolution for Tension-only model

**Fig.12** represents the evolution of internal variable with time for this model. The evolution of internal variable in the second step and the third step takes place with the same slope as the elastic regime is not crossed in both of these steps. Hence, the hardening variable also remains constant for these steps.

- *Non-symmetric Damage Model*-

The same results have been obtained for Non-symmetric model.

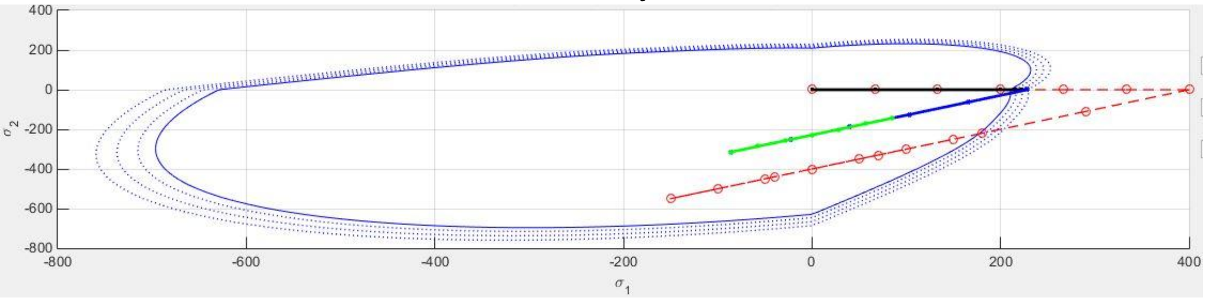


**Fig.13a** Damage Surface (Hardening) for Non-symmetric model

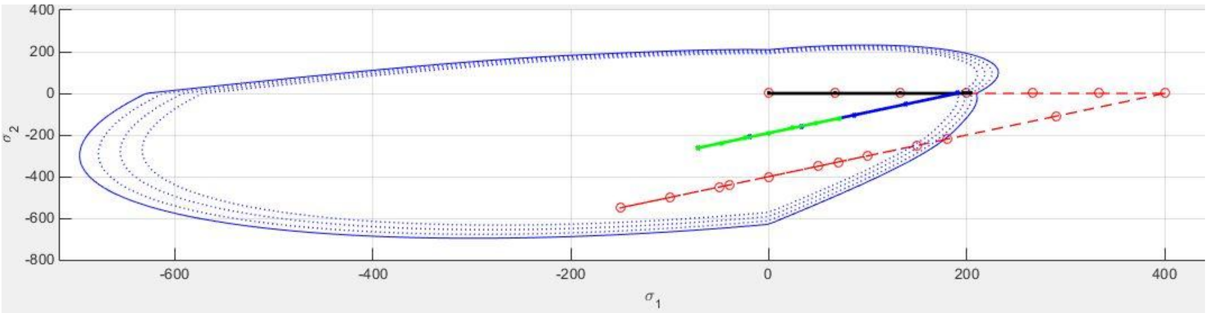


**Fig.13b** Damage Surface (Softening) for Non-symmetric model

**Fig.13a** and **13b** represent the damage surfaces for hardening (H=0.1) and softening (H= -0.1) case of the Non-symmetric model. The behavior is observed to be similar to the Tension-only model.

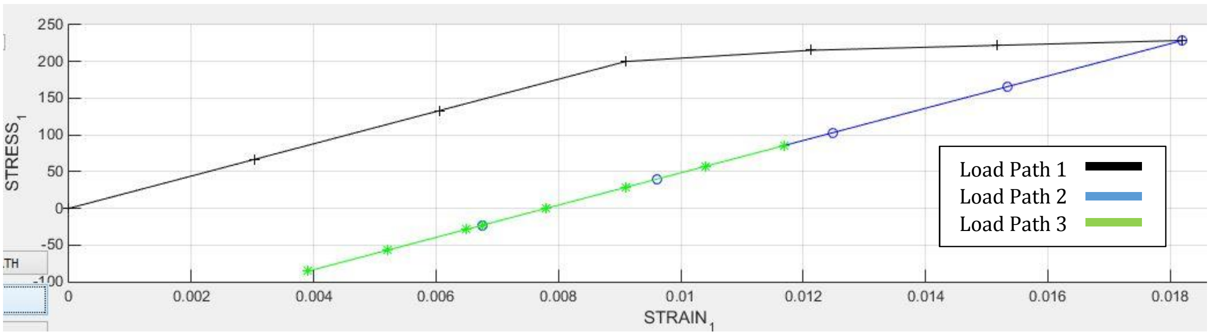


**Fig.14a** Stress-Strain Curve (Hardening) for Non-symmetric model

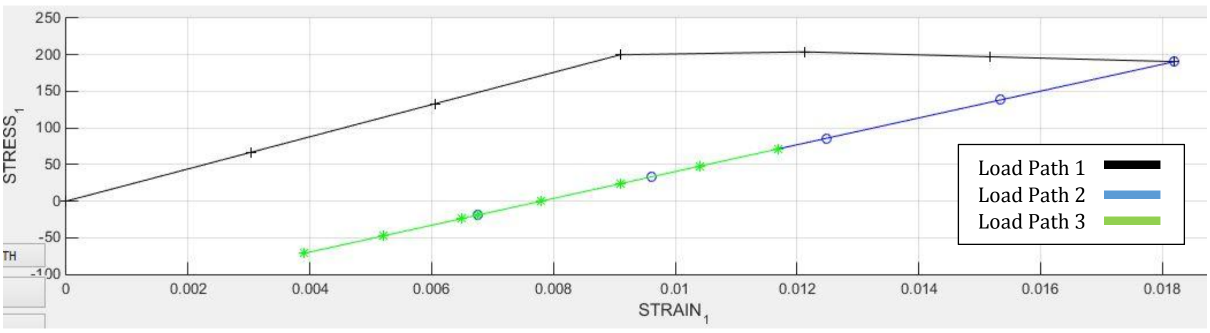


**Fig.14b** Stress-Strain Curve (Softening) for Non-symmetric model

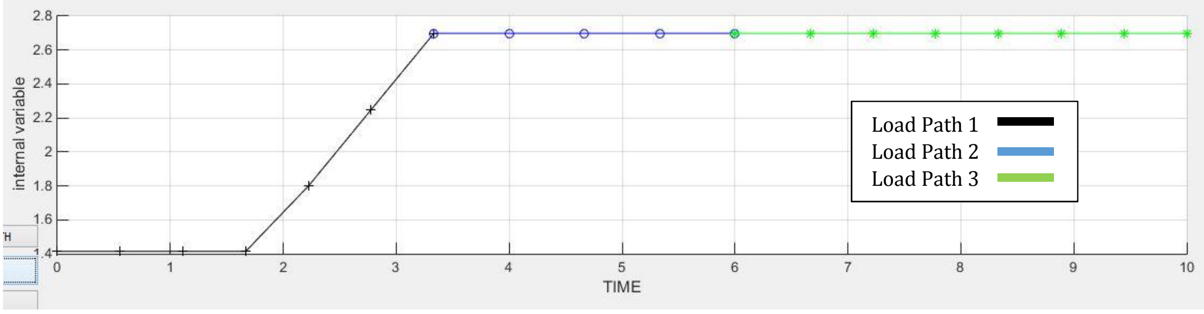


**Fig.15** Internal variable evolution for Non-symmetric model

**Fig.14a** and **14b** represent the stress-strain curves for hardening and softening case respectively and **Fig.15** shows the evolution of internal variable. The results can be seen similar with the Tension-only model.

- **<u>Case 3</u>**:

Young's Modulus = 20000, Yield Stress = 200, Poisson ratio = 0.3, H = ±0.1

Load path: $\Delta\bar{\sigma}_1^{(1)} = 400 \quad \Delta\bar{\sigma}_2^{(1)} = 400$

$$\Delta\bar{\sigma}_1^{(2)} = -550 \quad \Delta\bar{\sigma}_2^{(2)} = -550$$

$$\Delta\bar{\sigma}_1^{(3)} = 300 \quad \Delta\bar{\sigma}_2^{(3)} = 300$$

- *<u>Tension-only Damage Model</u>*-

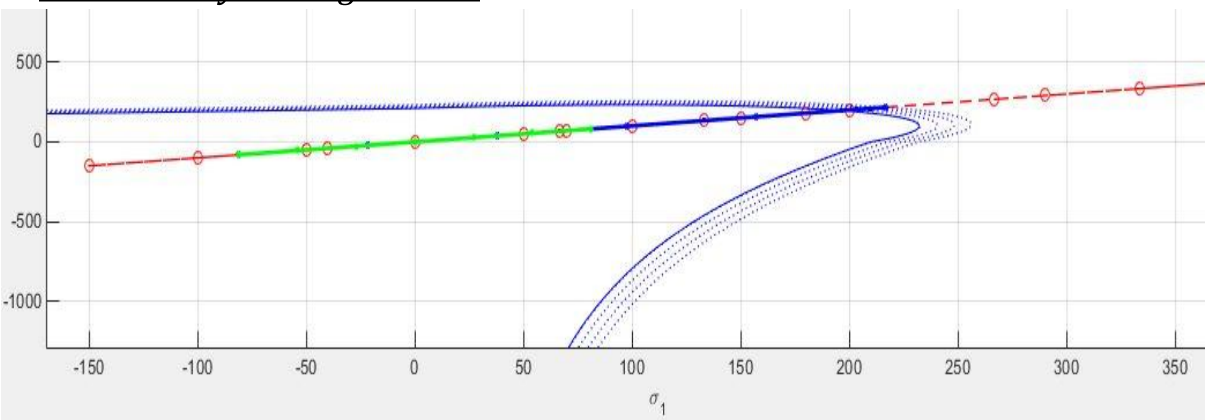


**Fig.16a** Damage Surface (Hardening) for Tension-only model

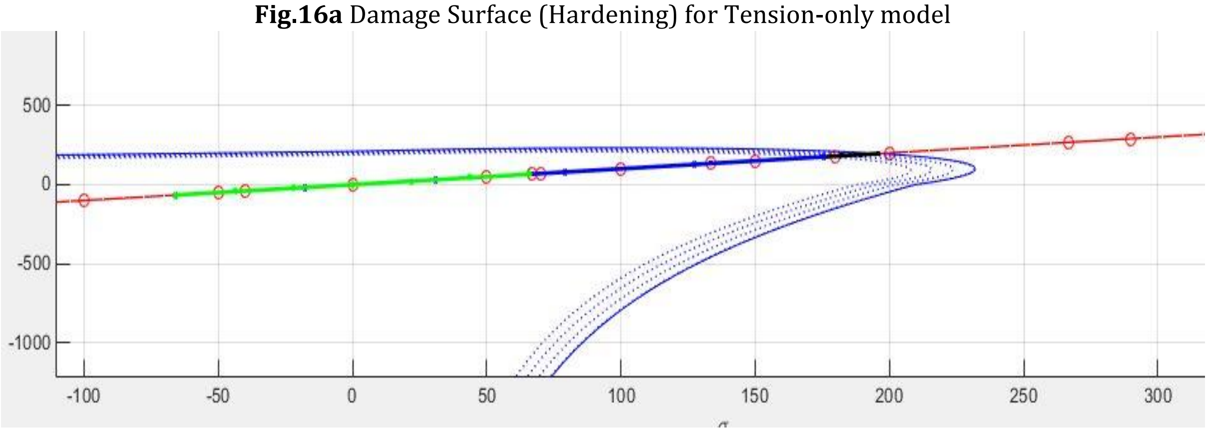


**Fig.16b** Damage Surface (Softening) for Tension-only model

**Fig.16a** and **16b** represent the load path for hardening and softening respectively for the Tension-only damage model. The damage surface is crossed in the first step where the elastic region is crossed and the damage takes place.

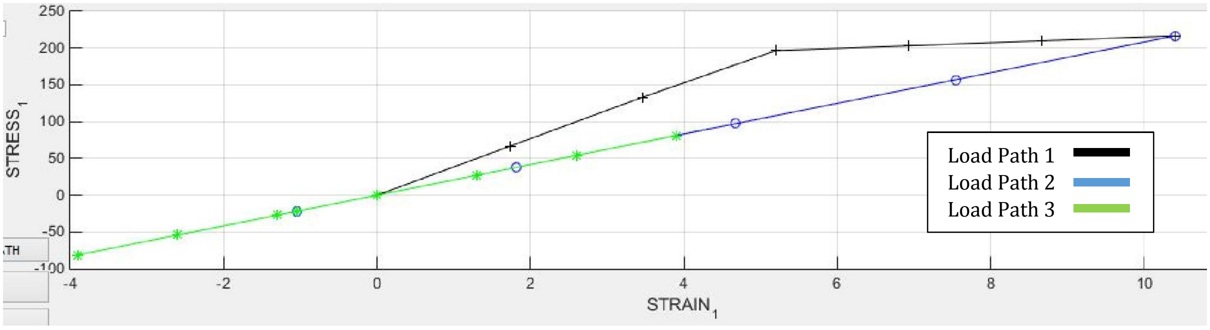


**Fig.17a** Stress-Strain Curve (Hardening) for Tension-only model

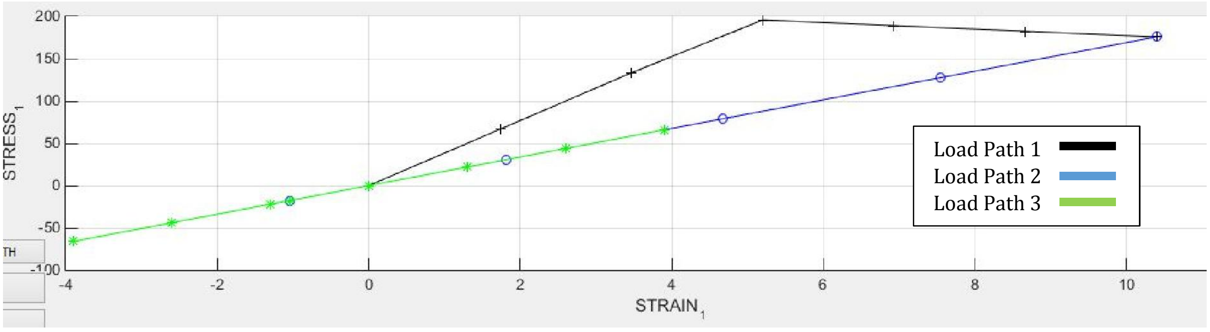


**Fig.17b** Stress-Strain Curve (Softening) for Tension-only model

**Fig.17a** and **17b** represent the stress-strain curve for hardening (H=0.1) and softening (H= -0.1) respectively for the Case 3 of the Tension-only damage model. The first step indicated by the black line is the biaxial loading which crosses the damage surface. The changing of slope in the first step causes change in the curve which represents that elastic limit is overpassed. In the second step shown by the blue line, biaxial compression or biaxial tensile unloading takes place without surpassing the elastic domain. And the third step represented by green line is biaxial tensile loading which again does not cross the elastic regime.

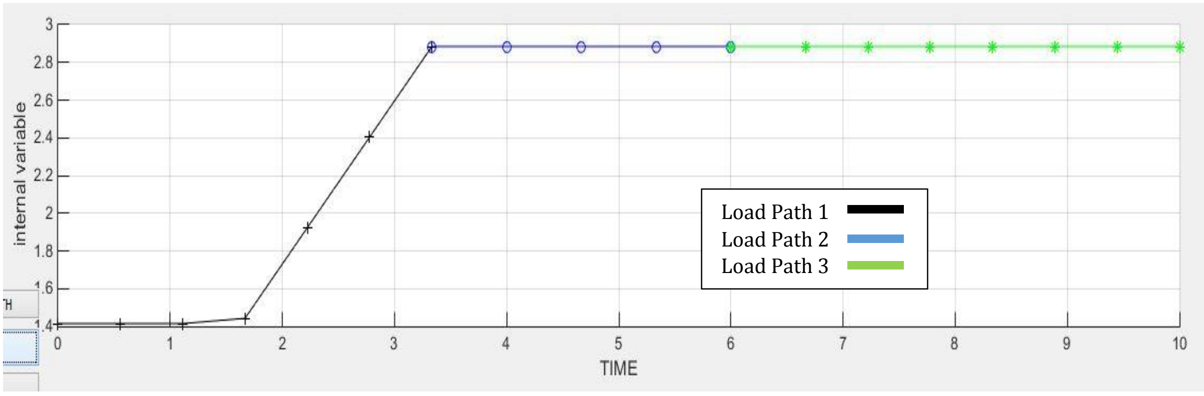


**Fig.18** Internal variable evolution for Tension-only model

And the **Fig.18** represents the evolution of internal variable with time for this model. The evolution of internal variable in the second step and the third step takes place with the same slope as in both steps, the biaxial tensile unloading and the biaxial tensile loading, the elastic regime is not crossed.

- ***Non-symmetric Damage Model*-**

The results for Non-symmetric model have been shown below.

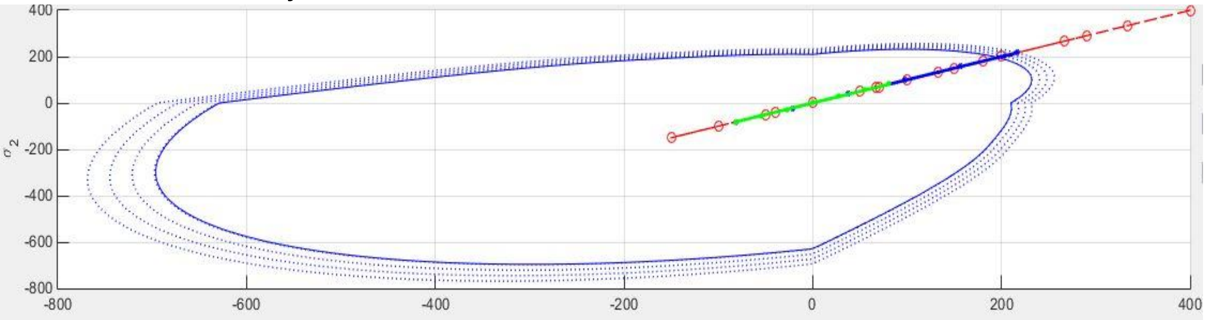


**Fig.19a** Damage Surface (Hardening) for Non-symmetric model

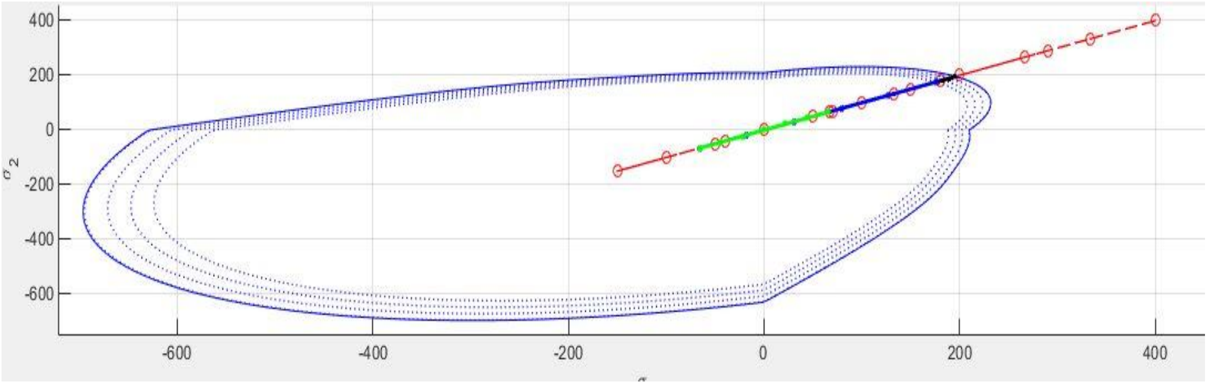


**Fig.19b** Damage Surface (Softening) for Non-symmetric model

**Fig.19a** and **19b** represent the damage surfaces for hardening (H=0.1) and softening (H= -0.1) case of the Non-symmetric model. The behavior is observed to be similar to the Tension-only model. **Fig.20a** and **20b** represent the stress-strain curve for hardening (H=0.1) and softening (H= -0.1) respectively for the Case 3 of the Non-symmetric damage model. And the **Fig.21** represents the evolution of internal variable with time.

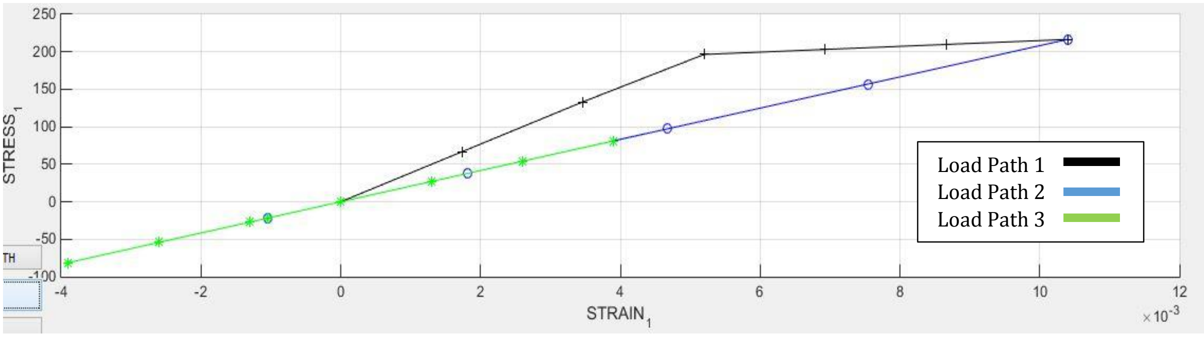


**Fig.20a** Stress-Strain Curve (Hardening) for Non-symmetric model

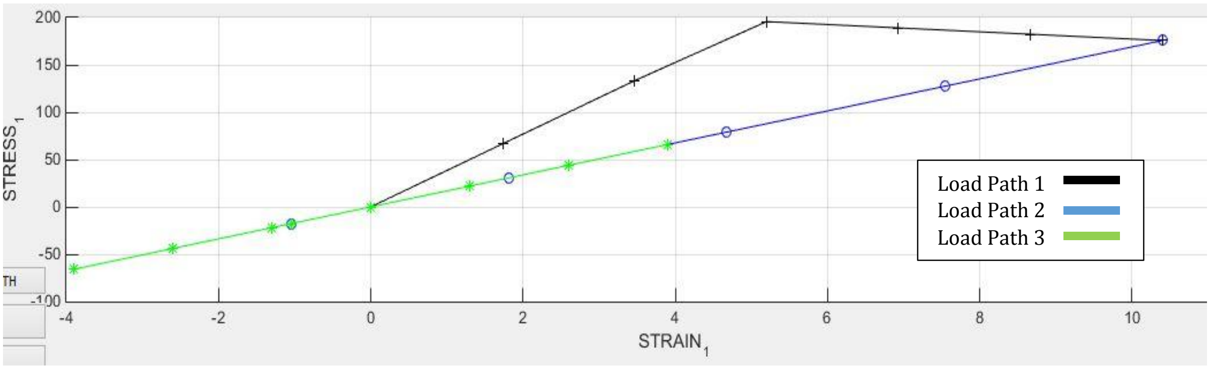


**Fig.20b** Stress-Strain Curve (Softening) for Non-symmetric model

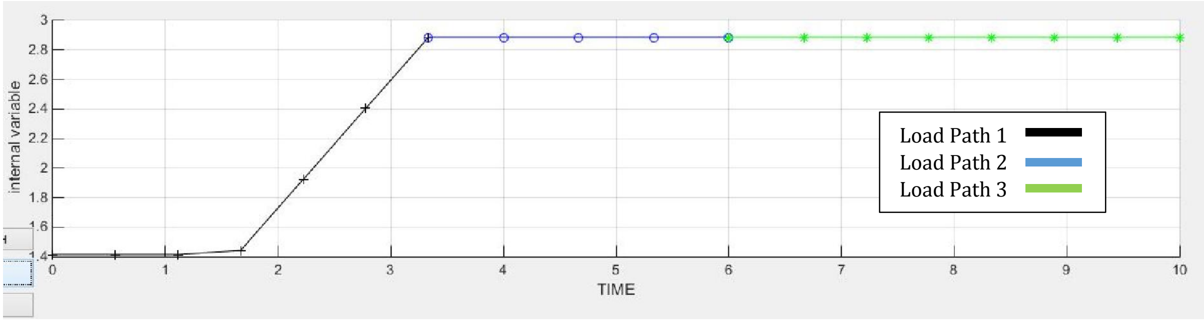


**Fig.21** Internal variable evolution for Non-symmetric model

- <u>Comparison for Viscid and Inviscid Case</u>:

A small comparison between the viscid and inviscid model has been done below in **Fig.22** for the symmetric case. Both the models have same behavior in the elastic region as observed from **Fig.22**. But, as the elastic region is passed, that is when the damage surface has been crossed, it seems that the viscid model tends to have a more steep slope indicating higher stress values while the inviscid model has a less steep slope.

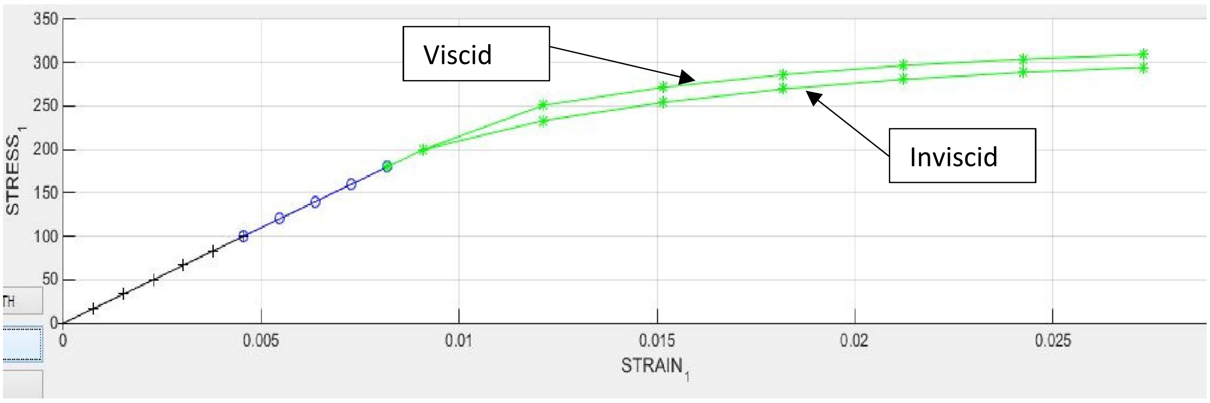


**Fig.22** Viscid/Inviscid case Comparison (stress-strain plot)

## Part 2 – **Rate Dependent Models**

In this part we have to implement the viscous case for symmetric model, and the effect of variation of the following parameters on the stress-strain curve have been visualized below. For this, the loading path selected is:

$$\Delta\bar{\sigma}_1^{(1)} = 100 \qquad \Delta\bar{\sigma}_2^{(1)} = 0$$
$$\Delta\bar{\sigma}_1^{(2)} = 100 \qquad \Delta\bar{\sigma}_2^{(2)} = 0$$
$$\Delta\bar{\sigma}_1^{(3)} = 400 \qquad \Delta\bar{\sigma}_2^{(3)} = 0$$

And Young's Modulus = 20000, Yield Stress = 200, Poisson ratio = 0.3.

- <u>Effect of Viscosity Parameter</u>:

For the above selected load path, the other parameters taken while plotting the below curve are η=0.3, α=0.5, H=0.1.

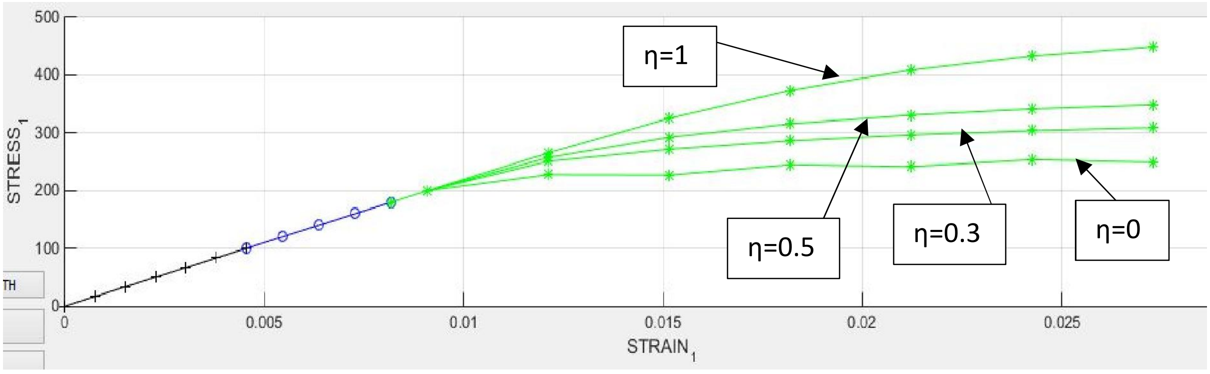


**Fig.23** Stress-Strain graph for different values of η

**Fig.23** represents the effect of different viscosity coefficients on the stress-strain graph. The different values for the viscosity parameter 'η' are taken as 0, 0.3, 0.5 and 1. In the elastic region, the viscosity associated with strain is zero hence we see no variation in the graph in the elastic region for different values of 'η'. But, after the elastic region is passed, higher viscosities account for higher values of stress and because of which we can observe the difference in the above graph in the inelastic region. The more the value of 'η', the more is the stress.

- <u>Effect of Strain Rate:</u>

For the above selected load path, the other parameters taken while plotting the below curve are η=1, α=0.5, H=0.1.

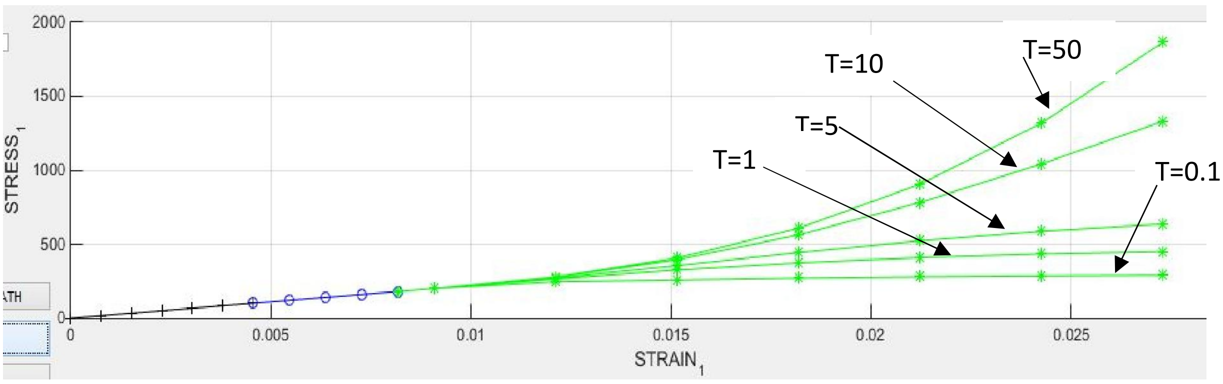


**Fig.24** Stress-Strain graph for different values of strain rate (time int.)

The strain rate variation is done by changing the time interval in the program. In the elastic region, as usual no variation is seen. But, the strain rate varies in the inelastic region and hence we observe variations in graph for different strain rates. Here, because of the property of the rate dependent models, the stresses give different values when the strain rate is changed. It is observed from **Fig.24**, as the time interval is increased, the stress values obtained are lower and more stable graph is obtained.

- <u>Effect of Time Integration Parameter</u>:

For the above selected load path, the other parameters taken while plotting the below curve are η=1, α=0.5, H= -0.5, T=1000

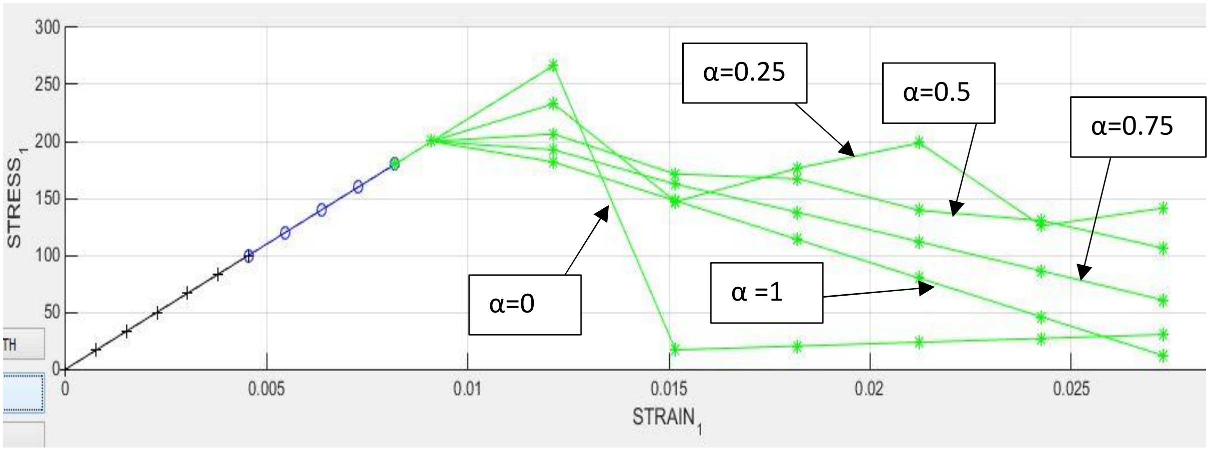


**Fig.25** Stress-Strain graph for different values of α

From **Fig.24**, as we know that, the more the time interval, more stable is the plot. Hence, we select T=1000 in order to get good results for differentiation. The different values of α for which the stress-strain graph is plotted are 0, 0.25, 0.5, 0.75 and 1. In these, we know that α=0 represents Forward Euler Method, α=0.5 represents Crank-Nicholson scheme and α=1 represents the Backward Euler Method.

From **Fig.25**, we can observe that as α increases from 0 to 1, the graph becomes more stable. For α=0, the graph is totally unstable while it stabilizes itself between the values 0.5-1. This tells us that at α=0 as the method is explicit which involves a less stable solution, it should not be preferred over α=1 which represents the implicit approach in order to get a stable solution.

# Conclusion:

This assignment was targeted to study different types of damage models in Damage Mechanics theory. As the scope of this assignment, the Symmetric, Tension-only and Non-symmetric damage models were coded evaluating different loading cases for rate independent (inviscid) and rate dependent (viscid) criteria.

The effect of linear and exponential hardening/softening was also studied on these models and it was visible that linear results seemed to have a faster response as compared to exponential results. Hardening and Softening cases studied had a main prominent difference that graphs had negative slopes for softening while positive for hardening.

The evolution of damage surfaces for the Tension-only and Non-symmetric damage models was observed for different loading paths and their stress-strain graphs were analyzed for the behavior shown. The first case was complete uniaxial loading, second being partly uniaxial and partly biaxial while the third case was completely biaxial loading. The stress-strain graphs depicted a slight change in slope whenever the damage surface was crossed in the load paths.

The Viscous case was implemented for symmetric model and the effect of different parameters such as viscosity coefficient, strain rate and time integration parameter was studied on the stress-strain curves. It was seen that, more the time interval for integration more stable are the results obtained and implicit method ($\alpha$=1) approaches could yield better results than explicit method ($\alpha$=0) approaches. Also, the viscid cases have a more steep slope as compared to the inviscid cases.

Lastly, the codes modified for this exercise have been attached in the appendix section for reference.

# Appendix:

The modified codes for implementation of the damage models has been mentioned in this section. The codes presented below are:
modelos_de_dano1.m, dibujar_criterio_dano1.m, rmap_dano1.m, damage_main.m

**a) modelos_de_dano1.m**

```
function [rtrial] = Modelos_de_dano1 (MDtype,ce,eps_n1,n)
%***********************************************************************
************
%*          Defining damage criterion surface
%*
%*
%*
%*
%*                         MDtype=  1       : SYMMETRIC
%*
%*                         MDtype=  2       : ONLY TENSION
%*
%*                         MDtype=  3       : NON-SYMMETRIC
%*
%*
%*
%*
%*
%* OUTPUT:
%*
%*                         rtrial
%*
%***********************************************************************
************



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

elseif (MDtype==2)  %* Only tension
sigma_n=(eps_n1*ce);
sigma_nplus=sigma_n.*(sigma_n>0);

rtrial= sqrt(sigma_nplus*eps_n1');

elseif (MDtype==3)  %*Non-symmetric
sigma_n=(eps_n1*ce);
sigma_nplus=sigma_n.*(sigma_n>0);
sigma_nabs=abs(sigma_n);
teta_ratio=sum(sigma_nplus)/sum(sigma_nabs);

rtrial= (teta_ratio+(1-teta_ratio)/n)*sqrt(eps_n1*ce*eps_n1');

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

**b) 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

tetha=[(-pi/2)*0.9999:0.01:pi*0.9999];
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
    sigma_n=[m1(i) m2(i) 0 nu*(m1(i)+m2(i))];
    sigma_nplus=sigma_n.*(sigma_n>0);
    radio(i)= q/sqrt(sigma_nplus*ce_inv*sigma_n');

    s1(i)=radio(i)*m1(i);
    s2(i)=radio(i)*m2(i);
end
hplot =plot(s1,s2,tipo_linea);

elseif MDtype==3

tetha=[0:0.01:2*pi];
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
    sigma_n=[m1(i) m2(i) 0 nu*(m1(i)+m2(i))];
    sigma_nplus=sigma_n.*(sigma_n>0);
    sigma_nabs=abs(sigma_n);
    teta_ratio=sum(sigma_nplus)/sum(sigma_nabs);

    radio(i)= (q/sqrt(sigma_n*ce_inv*sigma_n'))/(teta_ratio+(1-
teta_ratio)/n);
    s1(i)=radio(i)*m1(i);
    s2(i)=radio(i)*m2(i);
end
hplot =plot(s1,s2,tipo_linea);
end
return
```

### c) 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);
viscpr = Eprop(6);
eta = Eprop(7);
alpha = Eprop(8);
%*************************************************************************
***********


%*************************************************************************
***********
```

```
%*       initializing                                                 %*
 r0 = sigma_u/sqrt(E);
 zero_q=1.d-6*r0;
% if(r_n<=0.d0)
%     r_n=r0;
%     q_n=r0;
% end
%*********************************************************************
***********


%*********************************************************************
***********
%*       Damage surface
%*
[rtrial_prev] = Modelos_de_dano1 (MDtype,ce,eps_n,n);
[rtrial] = Modelos_de_dano1 (MDtype,ce,eps_n1,n);
rtrial_n_alpha = rtrial_prev*(1-alpha)+rtrial*alpha;
%*********************************************************************
***********


%*********************************************************************
***********
%*   Ver el Estado de Carga
%*
%*   --------->    fload=0 : elastic unload
%*
%*   --------->    fload=1 : damage (compute algorithmic constitutive
tensor)          %*
q_inf=2;
A=1;
fload=0;
if viscpr == 0
    if(rtrial > r_n)

    %*   Loading

    fload=1;
    delta_r=rtrial-r_n;
    r_n1= rtrial  ;
    if hard_type == 0
        %  Linear
        q_n1= q_n+ H*delta_r;
    else
        H_n= A*(q_inf-r0)/r0*exp(A*(1-r_n/r0));
        q_n1= q_n+ H_n*delta_r ;
    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
else
    if rtrial_n_alpha > r_n
        fload=1;        %loading
        delta_r=rtrial_n_alpha-r_n;

        % computation of r at the step n+1
        r_n1 = (eta - delta_t*(1-alpha))/(eta + alpha*delta_t)*r_n +
(delta_t/(eta + alpha*delta_t))*rtrial_n_alpha;

        if hard_type==0
            % linear
            H_n1 = H;
            q_n1= q_n+ H_n1*delta_r;
        else
            H_n= A*(q_inf-r0)/r0*exp(A*(1-r_n/r0));
            q_n1= q_n+ H_n*delta_r;
        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
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')

%*************************************************************************
***********
%Ce_tang_1

if viscpr == 1
    if rtrial_n_alpha > r_n

        % Constitutive Tangent Matrix Algorithm
        Ce_alg_n1 = (1.d0-
dano_n1)*ce+((alpha*delta_t)/(eta+alpha*delta_t))*(1/rtrial_n_alpha)*((H_n1
*r_n1-q_n1)/(ce*eps_n1')'*(ce*eps_n1'));
        C_alg = Ce_alg_n1(1,1);

        % Constitutive Tangent Matrix
        Ce_tan_n1 = (1.d0-dano_n1)*ce;
        C_tan = Ce_tan_n1(1,1);
    else
```

```
        % Constitutive Tangent Matrix Algorithm
        Ce_alg_n1=(1.d0-dano_n1)*ce;
        C_alg = Ce_alg_n1(1,1);

        % Constitutive Tangent Matrix
        Ce_tan_n1 = Ce_alg_n1;
        C_tan = C_alg;
    end
else
    if rtrial > r_n
        Ce_tan_n1 = (1.d0-dano_n1)*ce+(1/rtrial)*((H_n1*r_n1-
q_n1)/(r_n1^2))*((ce*eps_n1')'*(ce*eps_n1'));
        C_tan = Ce_tan_n1(1,1);
    else
        Ce_tan_n1 = (1.d0-dano_n1)*ce;
        C_tan = Ce_tan_n1(1,1);
    end
end



%**************************************************************************
***********
%* Updating historic variables
%*
%  hvar_n1(1:4)  = eps_n1p;
hvar_n1(5)= r_n1 ;
hvar_n1(6)= q_n1 ;
%**************************************************************************
***********

if viscpr == 1
    hvar_n1(8)= C_alg;
    hvar_n1(9)= C_tan;
end


%**************************************************************************
***********
%* Auxiliar variables
%*
aux_var(1) = fload;
aux_var(2) = q_n1/r_n1;
%*aux_var(3) = (q_n1-H*r_n1)/r_n1^3;
%**************************************************************************
**********
```

### d) damage_main.m

```
function
[sigma_v,vartoplot,LABELPLOT,TIMEVECTOR]=damage_main(Eprop,ntype,istep,stra
in,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

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,:) ;

        % Total strain at step "i-1"
        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);
        % 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
```