Computational Solid Mechanics

(Master of Science in Computational Mechanics)

Assignment 1

Assignment on Continuum Damage Models

Submitted by,

Krupesh Beekanahalli Shivaprakash

Part -1 (Rate Independent Models)

a) (Implementation of Non symmetric tension compression model and tension only model)

An inviscid damage model is considered. In the supplied MATLAB code the integration algorithms (rate independent and plane strain case) are implemented for 'non symmetric tension compression model 'and 'tension only model'. The major part of the modified (added) MATLAB code is attached in annex 1.

Consider, Non symmetric model Tension Compression model. An arbitrary loading path with arbitrary material properties is considered. The plot of damage surface is shown in figure A1. The ratio of compressive strength to tensile strength, n= 2 for the current plot. The Elastic domain is larger in the negative quadrant. The corresponding of evolution of damage variable over time is shown in figure A2. As shown in figure, the damage variable never decreases.

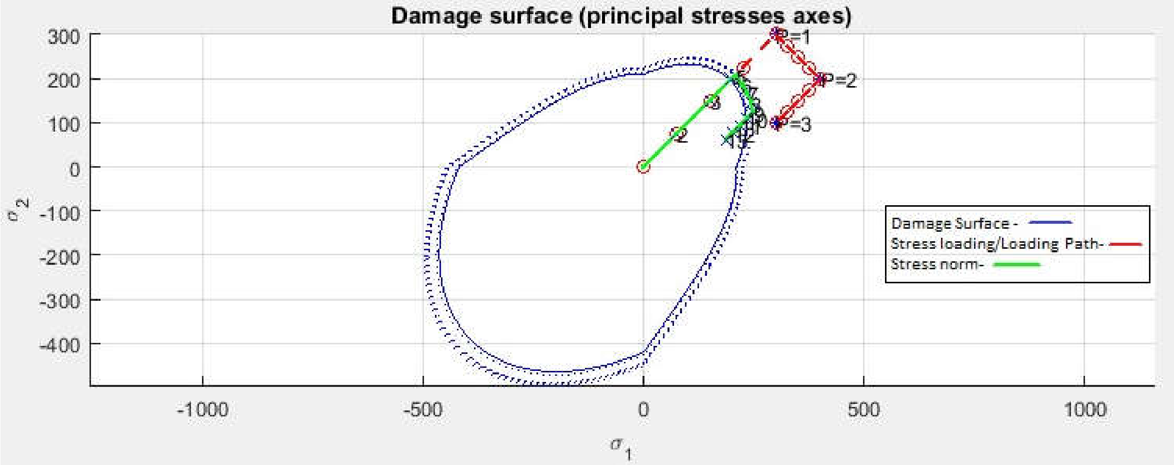

Figure A1: Damage Surface of Non symmetric Model

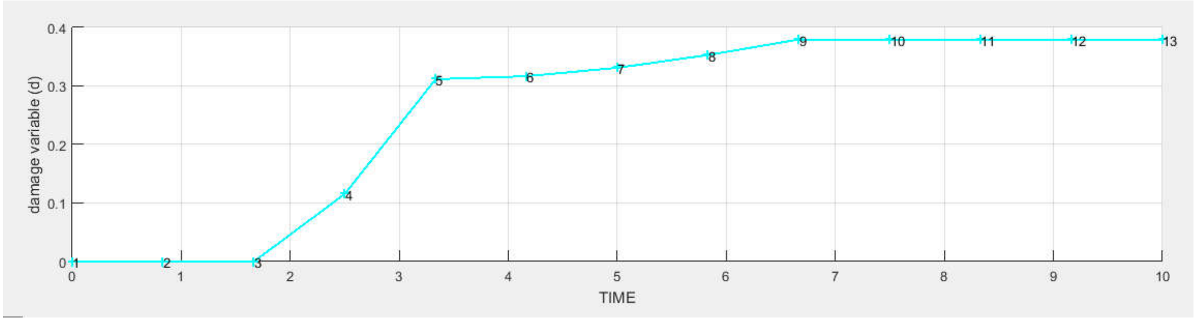

Figure A2: Plot of damage variable vs. time for Non symmetric Model

Consider, Tension only model. A sample loading path is considered in shown in figure A3 and A4. As we are considering the positive counterpart of stresses, the size of elastic domain is normal in pure tensile region, where as its infinite in pure compressive region. If we compare this model with non symmetric case, here the ratio of compressive strength to tensile strength is infinite. Hence, pure compressive state is always elastic. The corresponding of evolution of damage variable over time is shown in figure A5.

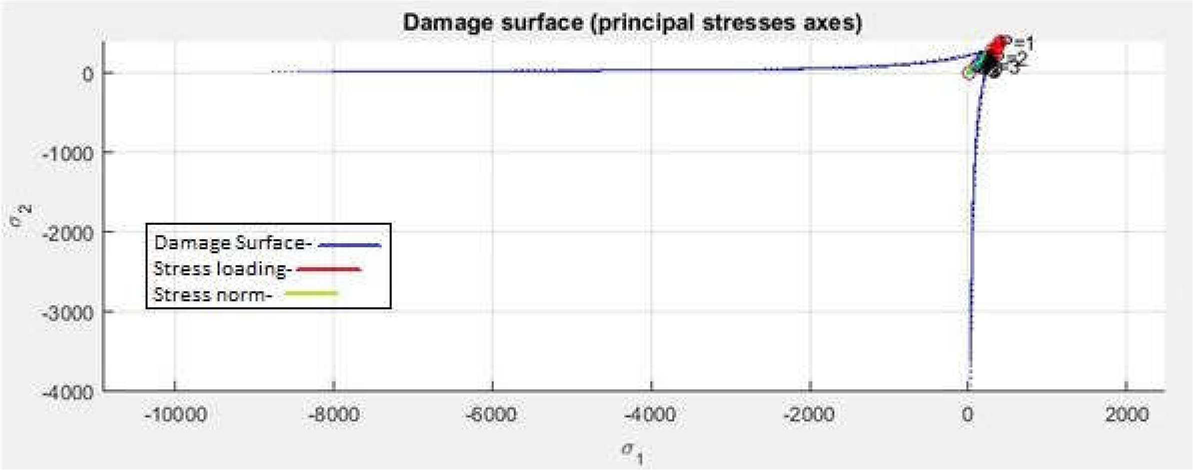

Figure A3: Damage Surface of Tension only Model

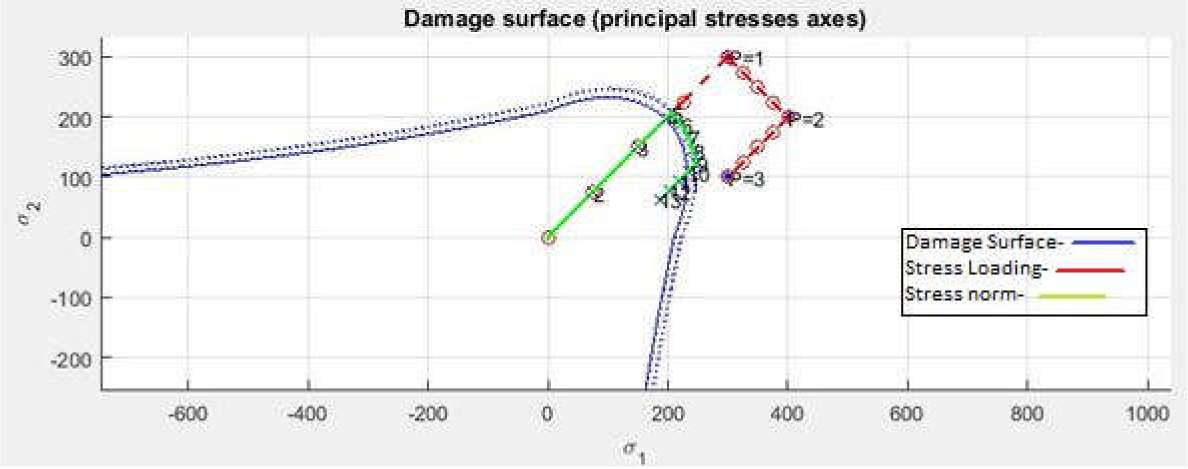

Figure A4: Damage Surface of Tension only Model (Maximized near loading path)

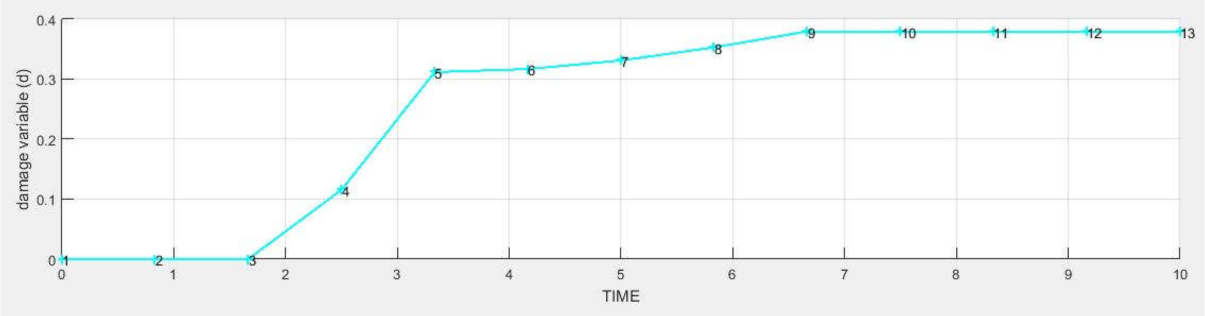

Figure A5: Plot of damage variable vs. time for Tension only Model

b) (linear and exponential hardening/softening (H<0 and H>0))

The effect of hardening and softening is illustrated in this part. MATLAB code is implemented for Exponential hardening and softening. The added code is attached in Annexure 2.

The previous implemented models (in section a)) are studied for Linear and exponential hardening / softening cases. The variations of hardening variable q(r) versus internal variable r are similar for all three models. They are as shown in following figures.

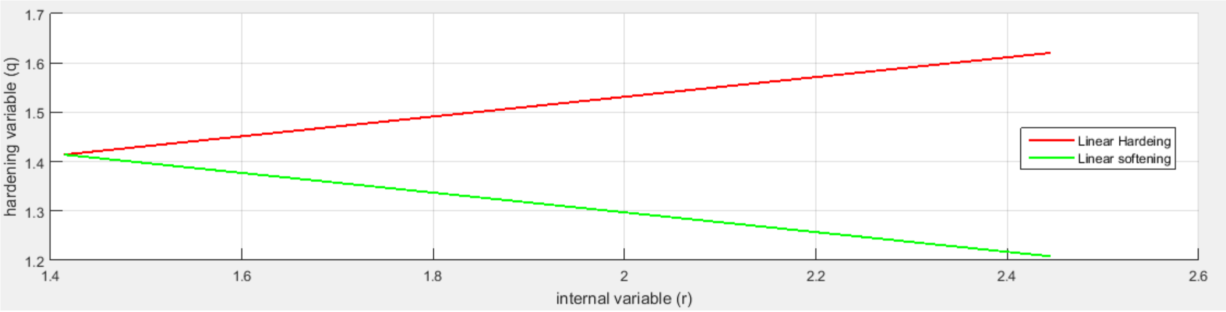

Figure B1: Graph of q(r) vs. r for Linear Hardening / Softening

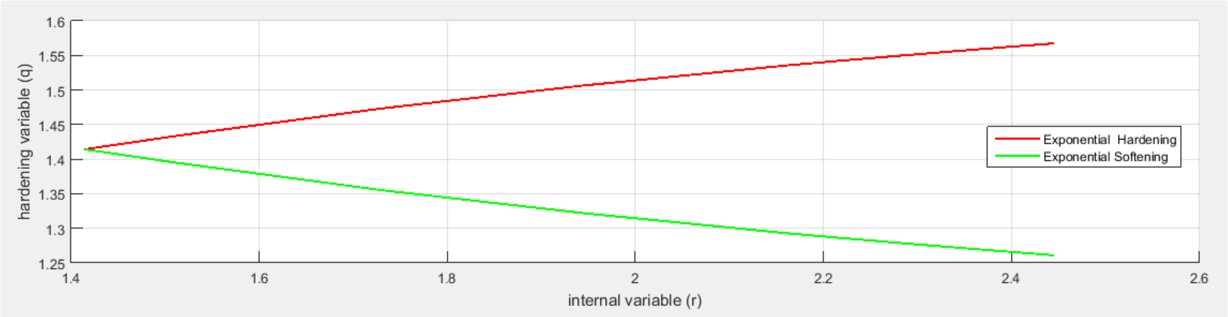

Figure B2: Graph of q(r) vs. r for Exponential Hardening / Softening

The plots of damage surfaces for Linear Hardening for Tension only model and Non symmetric Tension Compression models are shown in following figures. An arbitrary loading path is considered. The Hardness Modulus considered is, H=0.1 for hardening and H=-0.1 for softening. It’s clear from the graph that, on pure loading the damage surface expands in case of hardening and shrinks in case of softening.

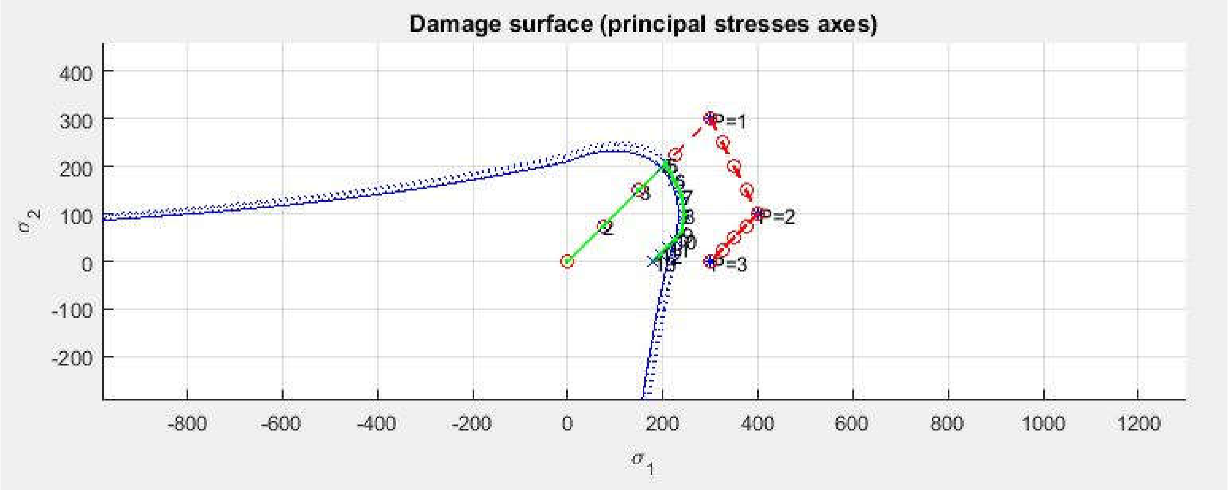

Figure B3: Linear Hardening in Tension only model

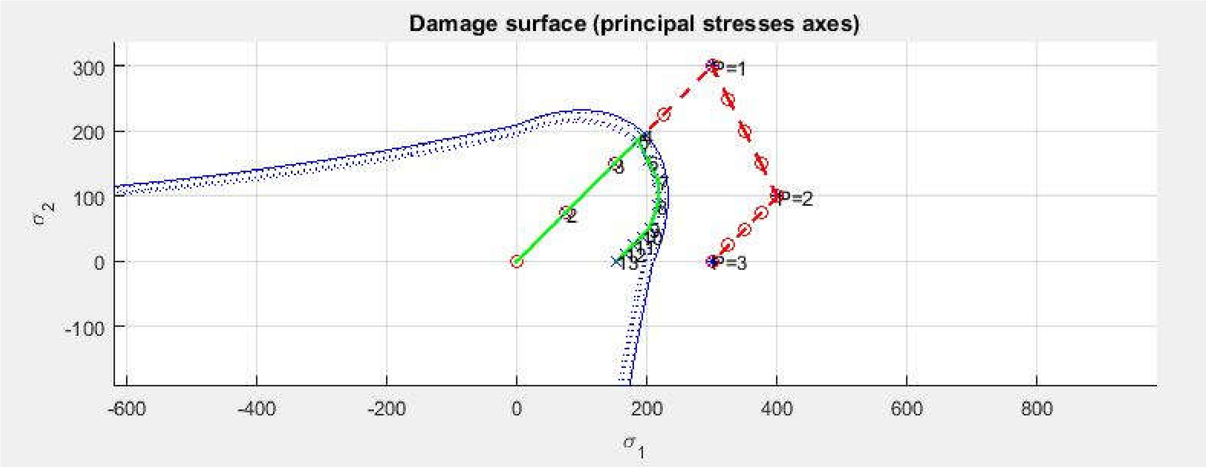

Figure B4: Linear Softening in Tension only model

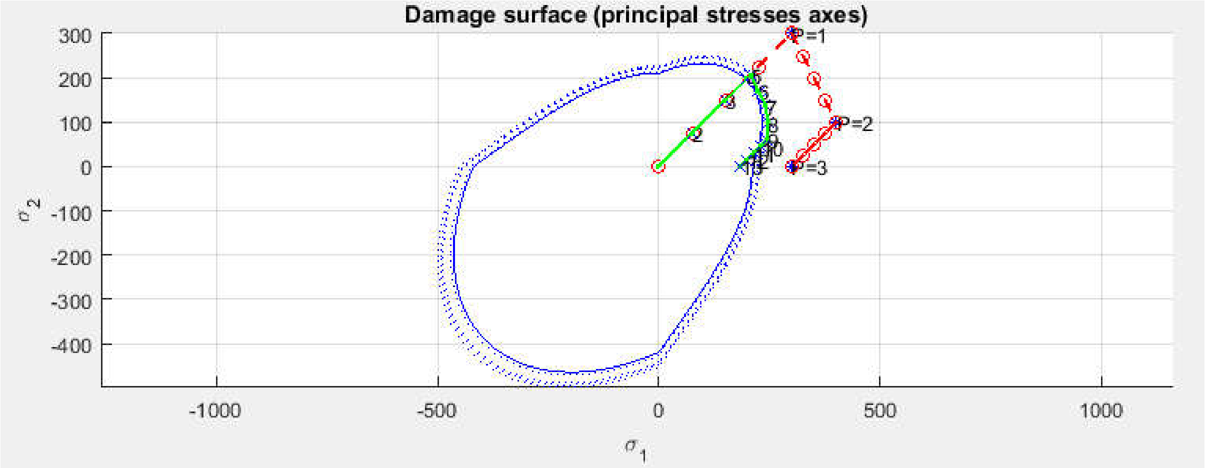

Figure B5: Linear Hardening in Non symmetric tension compression model

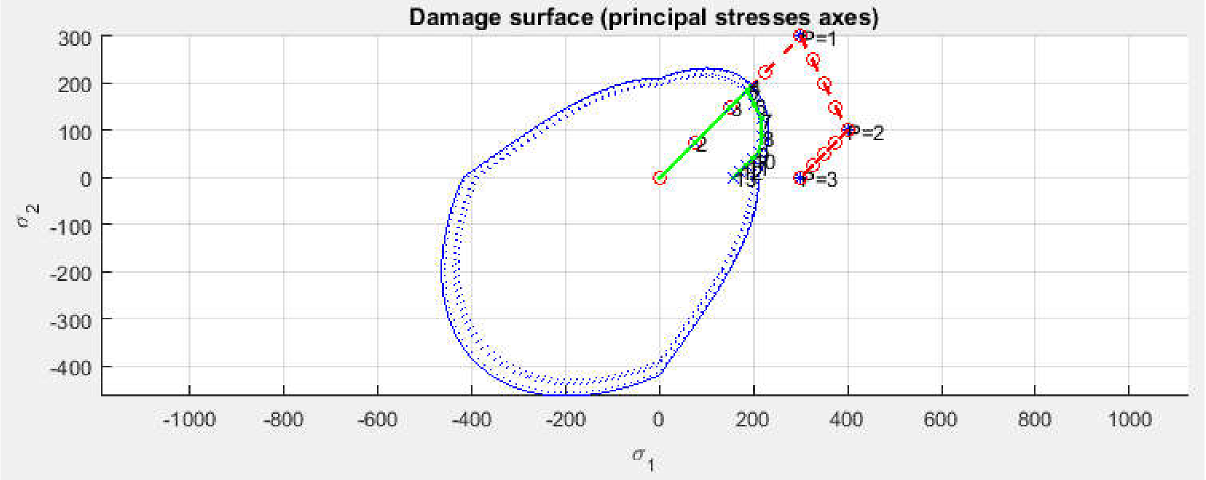

Figure B6: Linear Softening in Non symmetric tension compression model

The plots of damage surfaces for Exponential Hardening/Softening of three models are shown in following figures. An arbitrary loading path is considered. The Hardness Modulus considered is, H=0.1 for hardening and H=-0.1 for softening. Exponential hardening behaves similar to that of linear case.

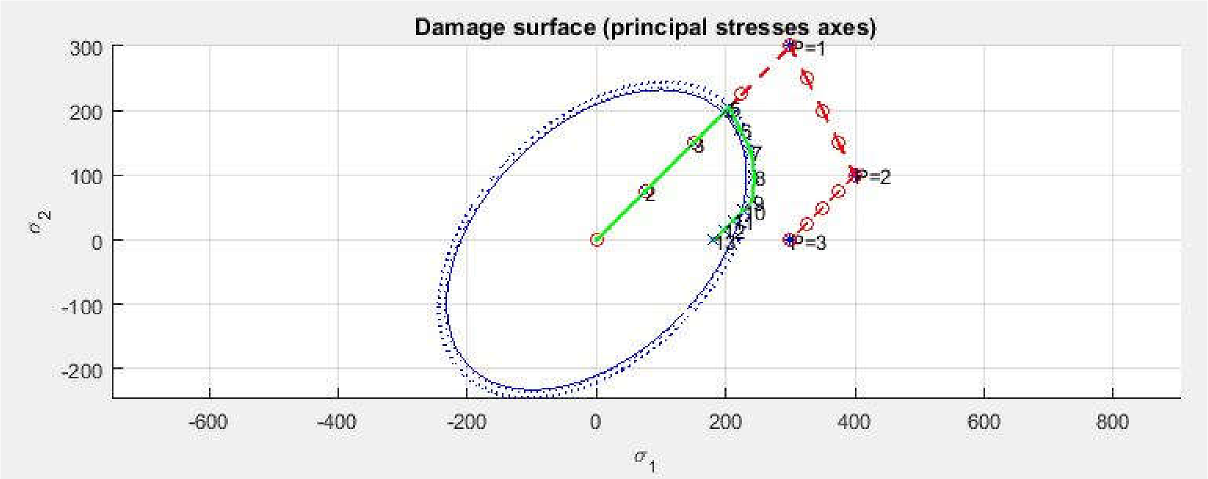

Figure B7: Exponential Hardening in Symmetric tension compression model

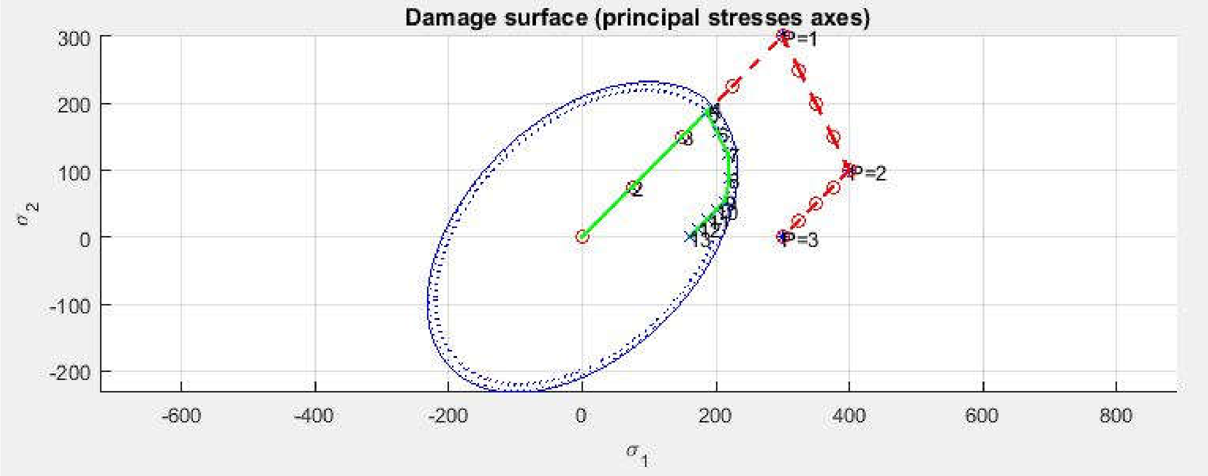

Figure B8: Exponential softening in Symmetric tension compression model

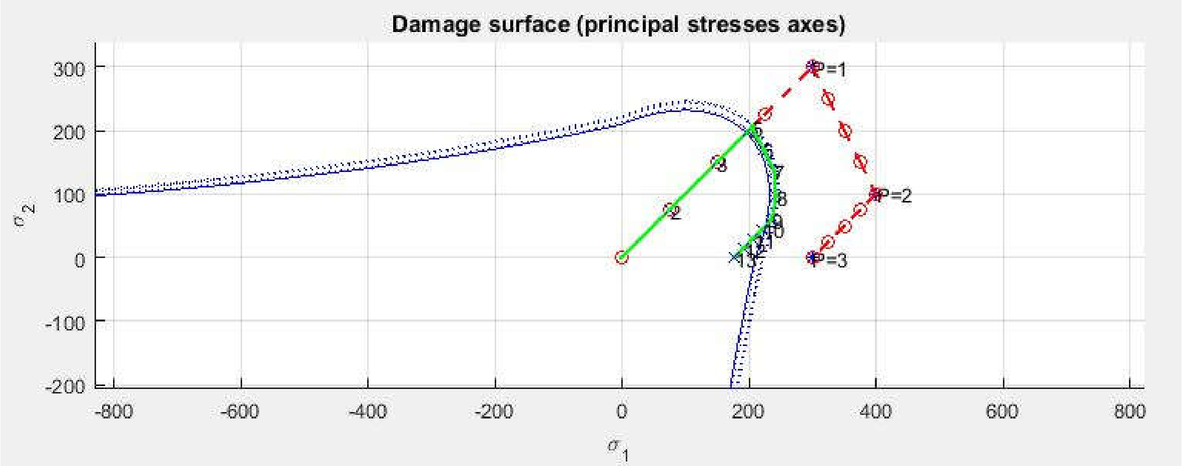

Figure B9: Exponential Hardening in tension only model

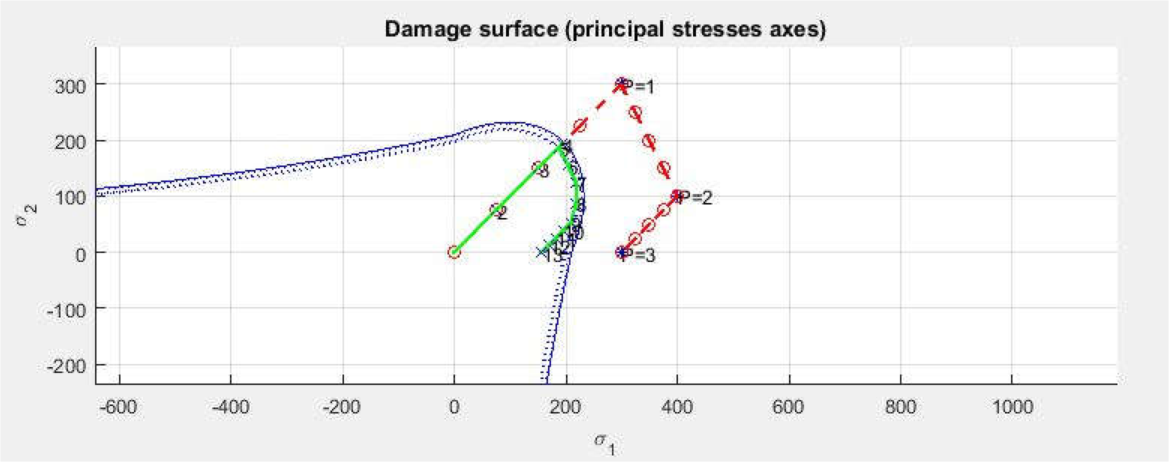

Figure B10: Exponential softening in tension only model

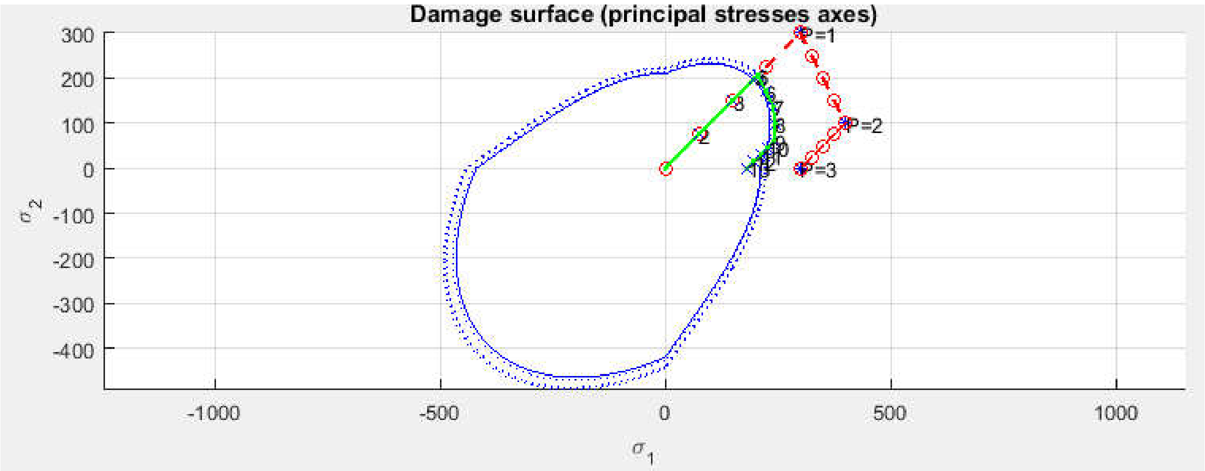

Figure B11: Exponential Hardening in Non symmetric tension compression model

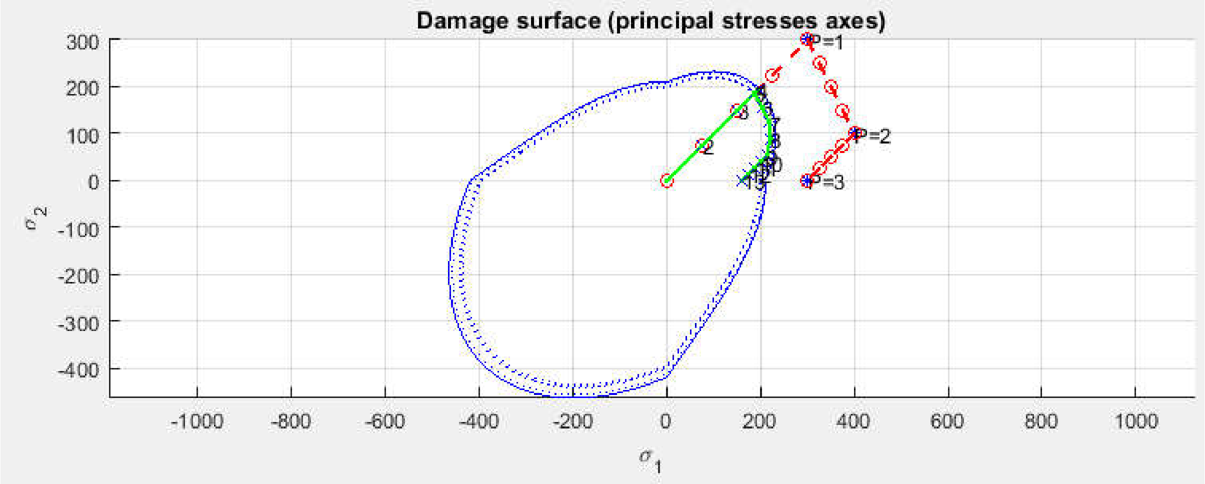

Figure B12: Exponential softening in Non symmetric tension compression model

c) (Assessment of correctness of the implementation)

The models in section a), Non symmetric tension compression model and Tension only model are loaded along appropriate paths as per given increments. To assess the correctness of implementation the path at the stress space and the stress-strain curve are analysed. The given loading paths are as follows.

1. $\Delta\sigma_1^{(1)} = \alpha$; $\Delta\sigma_2^{(1)} = 0$ (uniaxial tensile loading)
   $\Delta\sigma_1^{(2)} = \beta$; $\Delta\sigma_2^{(2)} = 0$ (uniaxial tensile unloading/compressive loading)
   $\Delta\sigma_1^{(3)} = \gamma$; $\Delta\sigma_2^{(3)} = 0$ (uniaxial compressive unloading/ tensile loading)

2. $\Delta\sigma_1^{(1)} = \alpha$; $\Delta\sigma_2^{(1)} = 0$ (uniaxial tensile loading)
   $\Delta\sigma_1^{(2)} = -\beta$; $\Delta\sigma_2^{(2)} = -\beta$ (biaxial tensile unloading/compressive loading)
   $\Delta\sigma_1^{(3)} = \gamma$; $\Delta\sigma_2^{(3)} = \gamma$ (biaxial compressive unloading/ tensile loading)

3. $\Delta\sigma_1^{(1)} = \alpha$; $\Delta\sigma_2^{(1)} = \alpha$ (biaxial tensile loading)
   $\Delta\sigma_1^{(2)} = -\beta$; $\Delta\sigma_2^{(2)} = -\beta$ (biaxial tensile unloading/compressive loading)
   $\Delta\sigma_1^{(3)} = \gamma$; $\Delta\sigma_2^{(3)} = \gamma$ (biaxial compressive unloading/ tensile loading)

Consider, α=200, β= 100, γ=200.

The following plots (Figure C1-C6) are obtained for Tensile only model. For easier visualization of plots, only two time steps are considered for each linear stage of loading path. Linear Hardening is considered with H=0.1. Arbitrary material properties are considered and they are kept constant. Only stress loading is being changed according given three loading paths.

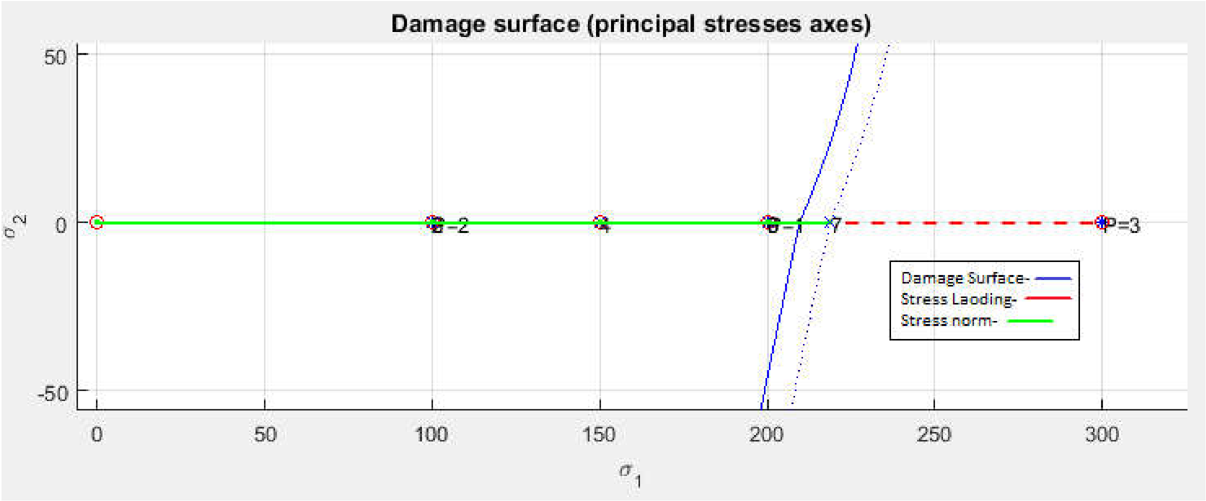

Figure C1: Stress norm path in given loading path 1

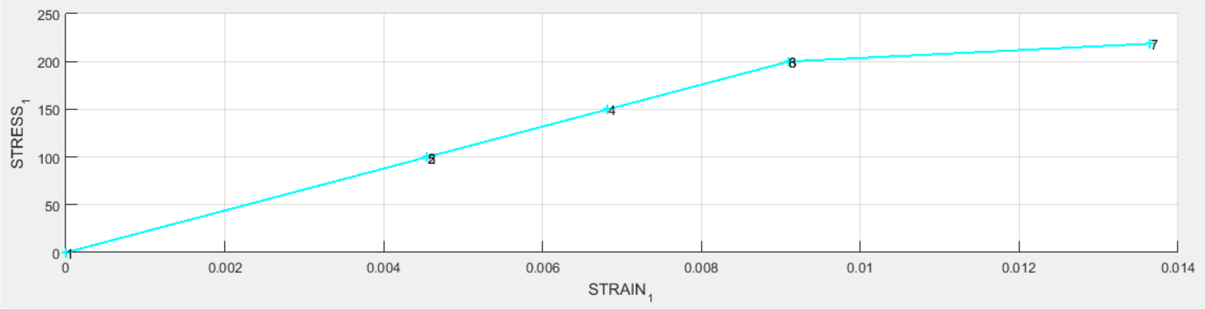

Figure C2: Plot of Stress vs. Strain for loading path 1

In figure C1, stress loading is done in three stages. In first linear stage uniaxial tensile load of 200 units is applied. The stress loaded just reaches the damage surface. Hence, there is no evolution of damage surface. In second stage uniaxial compressive load of 100 units is applied. Now, the stress moves inside the elastic domain. Hence, there is no evolution of elastic domain. In third stage uniaxial tensile load of 200 units is applied. Now the stress goes out of the elastic domain for one time step. Hence there is evolution of damage surface to cover that stress norm. The tensile model correctly satisfies the load path 1. This can also be verified in stress strain curve in figure C2. For first stage the stress is linear and elastic. For second stage the stress is elastically decreasing. Where as in third stage the stresses are non linear, indicates that the stress is outside elastic domain.

Similarly, tensile model satisfies other two given loading conditions (paths). Hence we can conclude that the tension only model is correctly implemented.

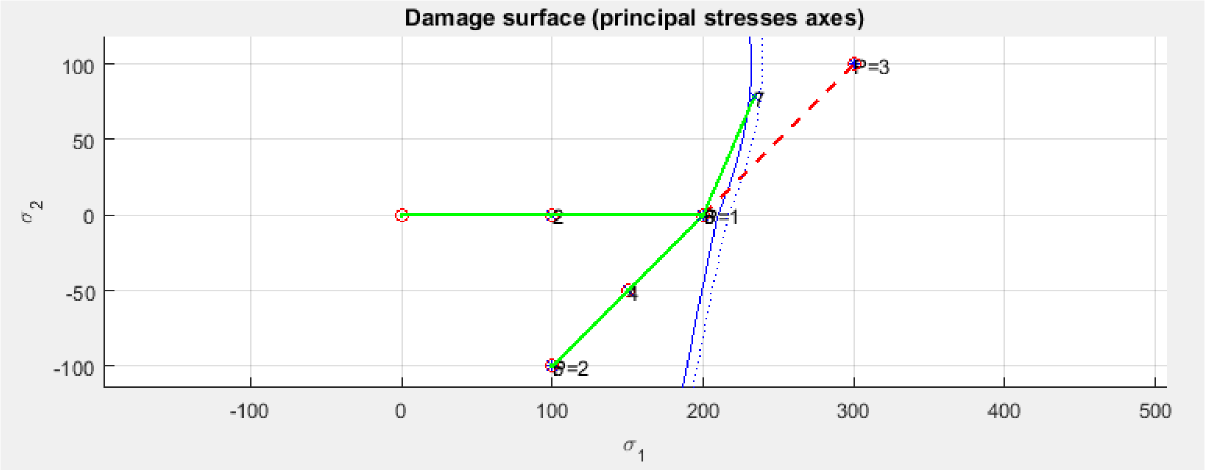

Figure C3: Stress norm path in loading path 2

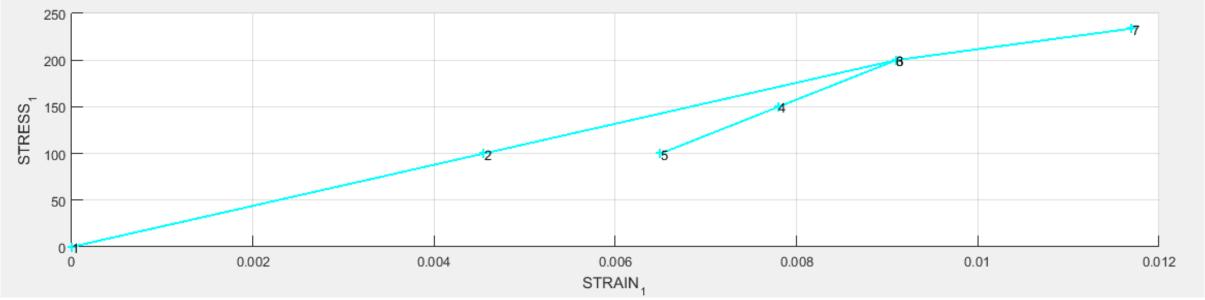

Figure C4: Plot of Stress vs. Strain for loading path 2

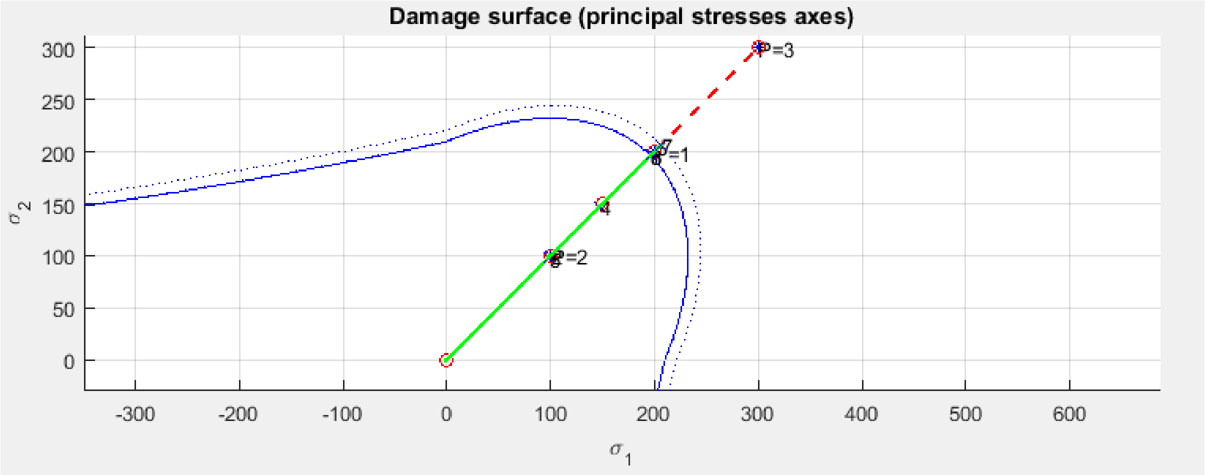

Figure C5: Stress norm path in loading path 3.

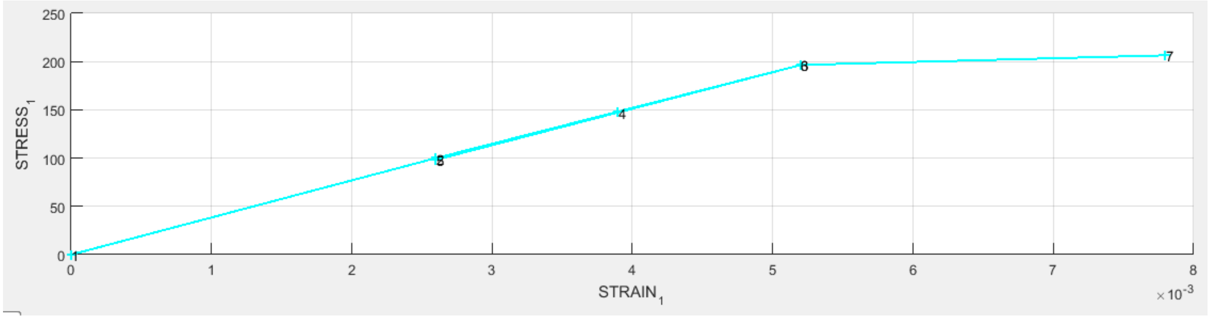

Figure C6: Stress -Strain curve for loading path 3

Now, for same values of α, β and γ, the corresponding graphs (Figures C7 to 12) are plotted for Non symmetric Tension Compression model. For easier visualization of plots only two time steps are considered for each line of loading path.

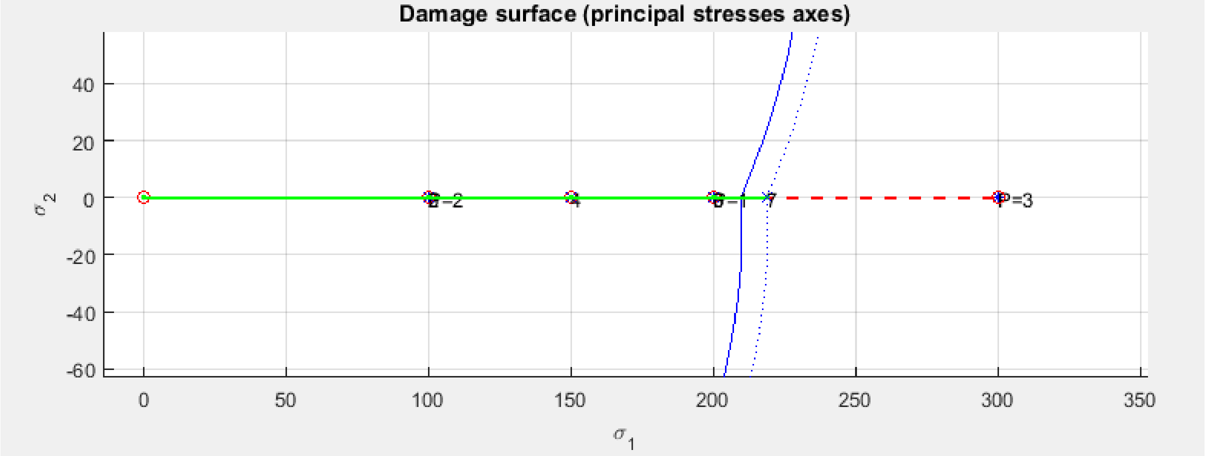

Figure C7: Stress norm path in loading path 1

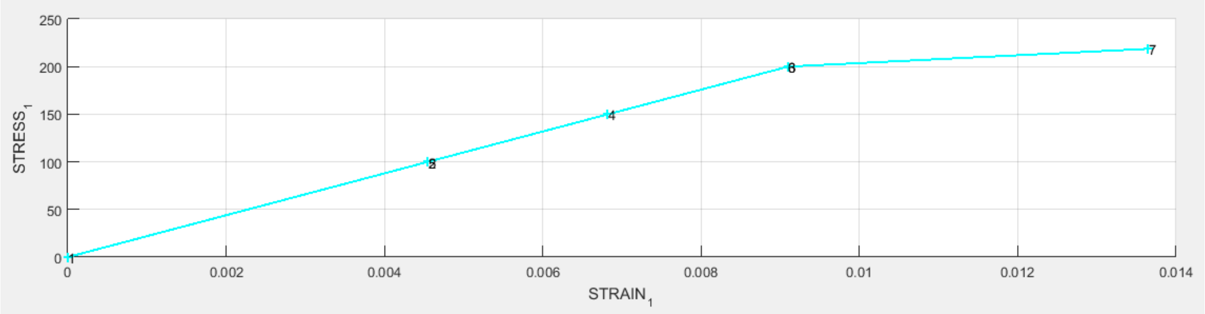

Figure C8: Plot of Stress vs. Strain   for loading path 1

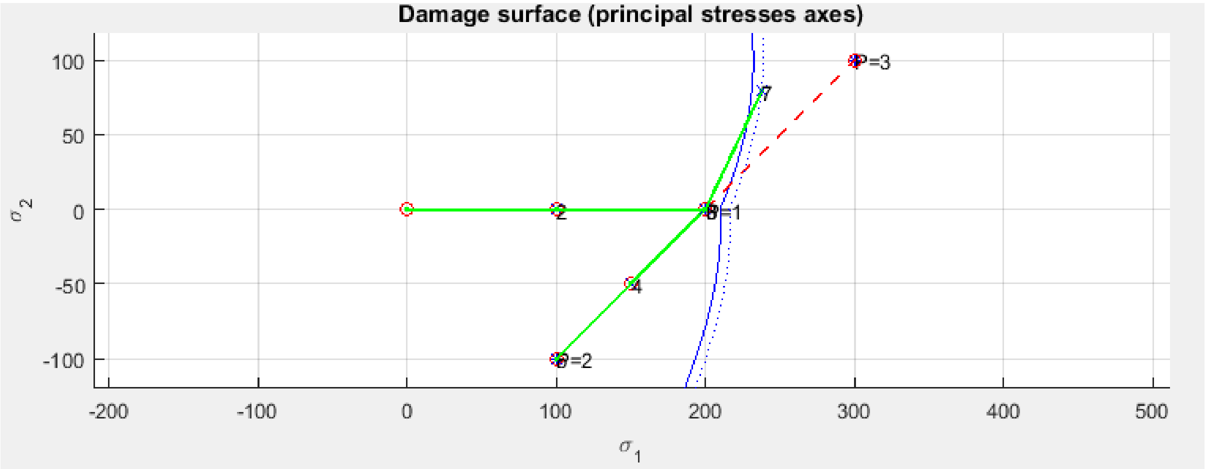

Figure C9: Stress norm path in loading path 2

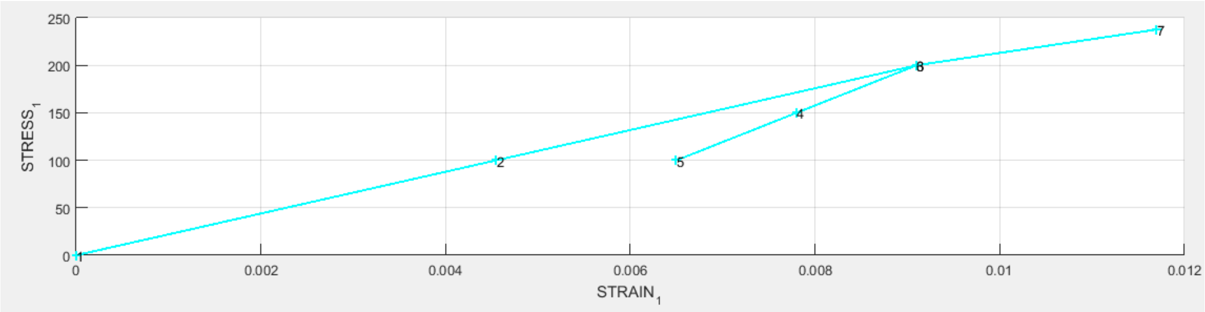

Figure C10: Stress vs. Strain   for loading path 2

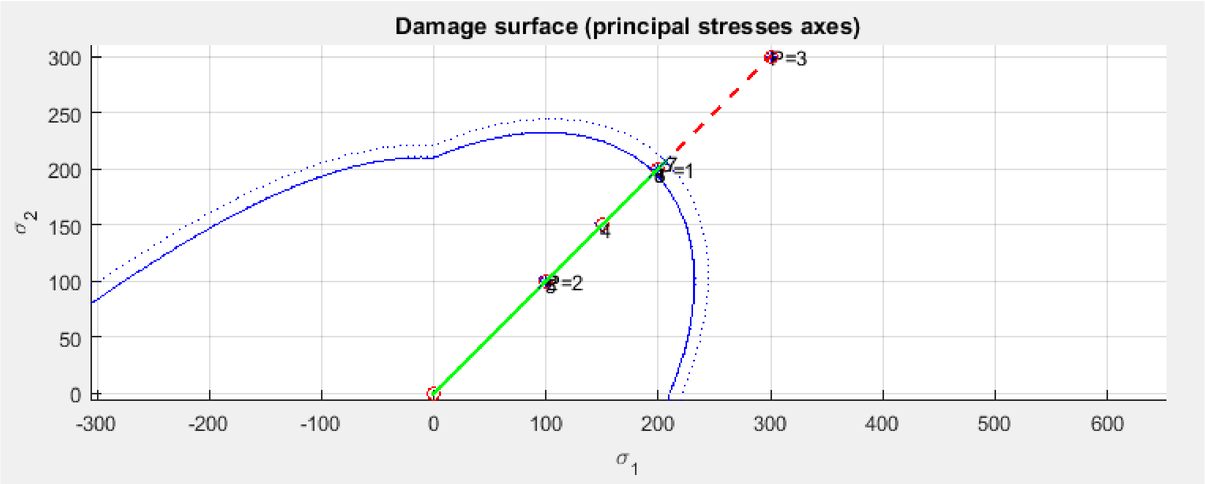

Figure C11: Stress norm path in loading path 3

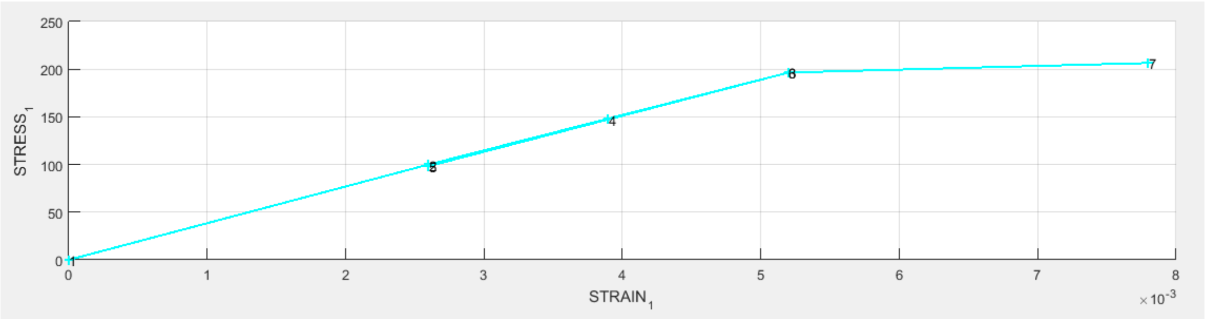

Figure C11: Stress vs. Strain   for loading path 3

Similar to ‘tensile only model’, the ‘non symmetric tension compression model’ satisfies the theory of continuum damage model in the given load paths. Hence correct implementation of non symmetric model is assessed.

PART II (rate dependent models)

d) **(Visco-damage model)**

MATLAB code is implemented for the integration algorithm (plane strain case) of the continuum isotropic visco-damage “symmetric tension compression “ model . The implemented code is attached in Annexure 3.

For an arbitrary loading path and material properties, viscosity parameter, η =0.3, the following plots are obtained (linear hardening with H=0.1 is considered). Damage surface is shown in figure D1 (for better visualisation, only four time steps are considered). The damage surface is expanding during pure loading. The corresponding Stress norm vs. strain norm plot is shown in figure D2. The viscous effect can be seen in non linear region.

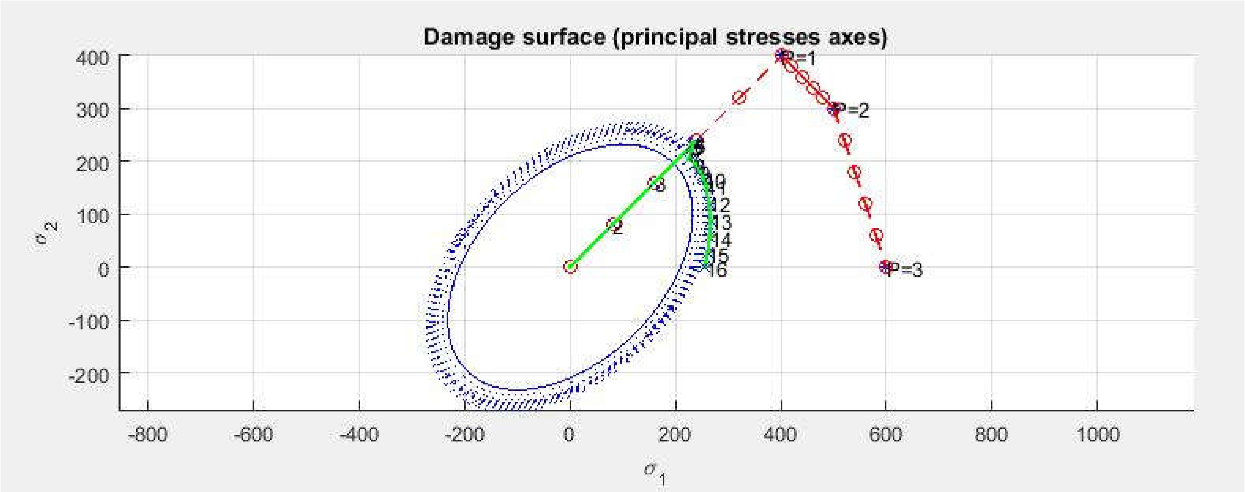

Figure D1: Damage surface for Viscous-Symmetric tension compression model

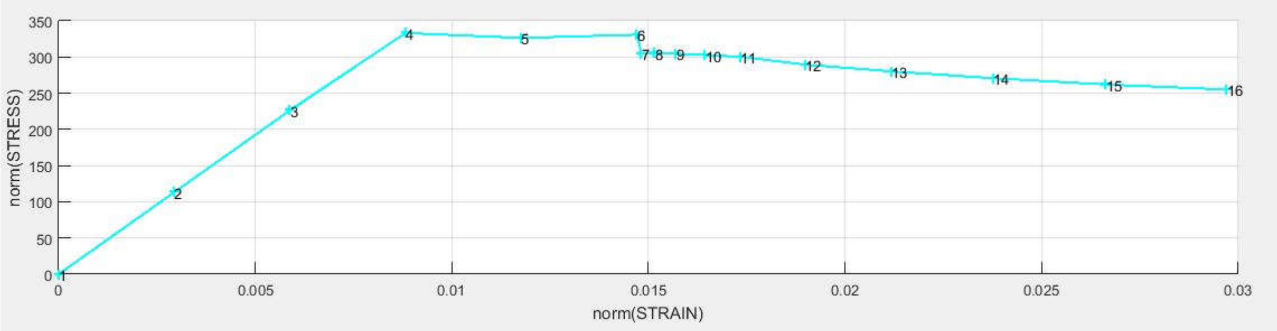

Figure D2: Stress vs. strain graph for Viscous-Symmetric model

e) (Assessing the correctness of the implementation)

1) The strain rate and the viscosity, show similar effects on the stress-strain curve. This is illustrated in figure E1 and figure E2 (for a fixed Poisson ratio of 0.3 and linear hardening with H=0.1). Viscosity rate is changed by varying the total time. The stress in non linear region of stress-strain curves increase with increase in strain rate or the viscosity parameter ‘η’.

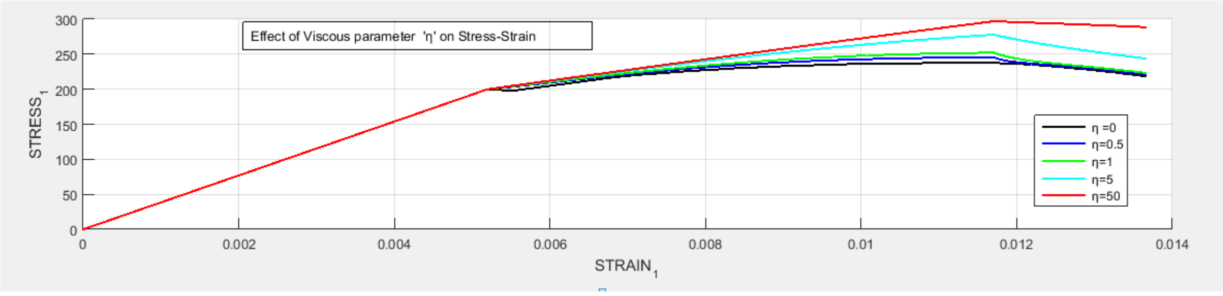

Figure E1: Effect of viscosity parameter (η)

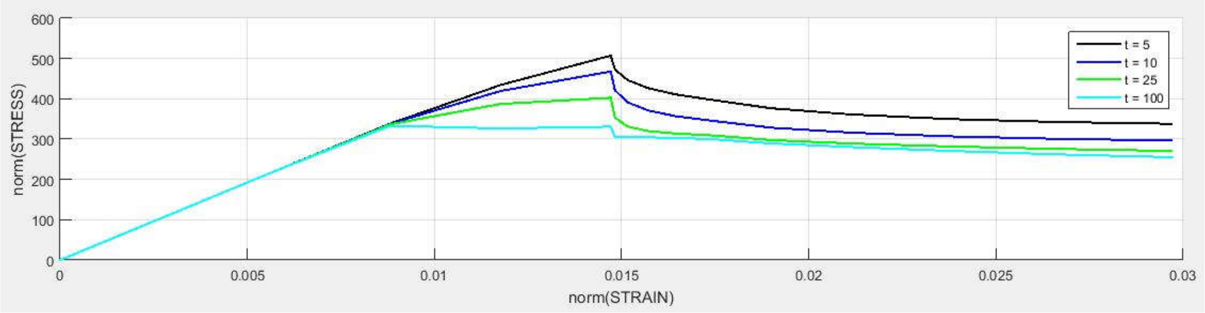

Figure E2: Effect of strain rate on Stress-strain curve

The effect of values of ‘α’ on stress-strain plots is displayed in following figure E3. The implicit methods give better results. The stress-strain curve is smoother in Backward Euler (Implicit, α=1) method. The Crank-Nicholson (midpoint rule, α=0.5) scheme gives the best result as it’s a second order method. The Forward Euler (explicit, α=0) gives highest fluctuations. It is conditionally stable.

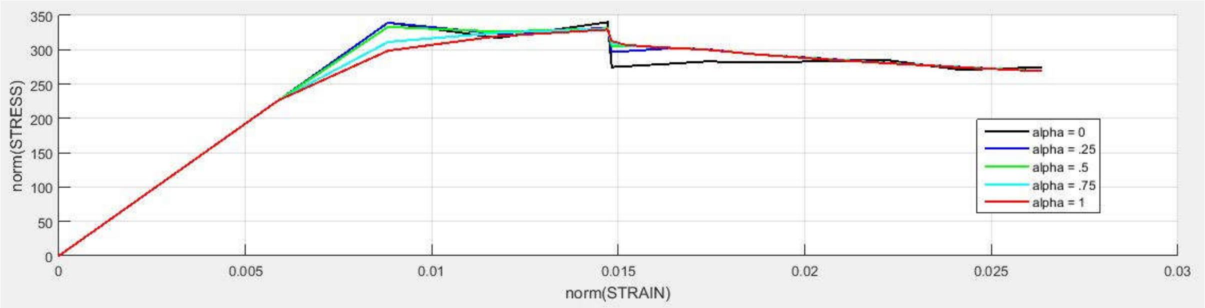

Figure E3: Effect of α (Time integration method) on stress-strain curves

From analysis of effect of strain rate, viscosity parameter (η) and α on stress-strain curves, confirms the correctness of the implementation of code for continuum isotropic visco-damage “symmetric tension compression” model.

2) MATLAB code is implemented to study the effect of 'α' values on the on the evolution along time of the C11 component of the tangent and algorithmic constitutive operators. The code is attached in Annexure 4.

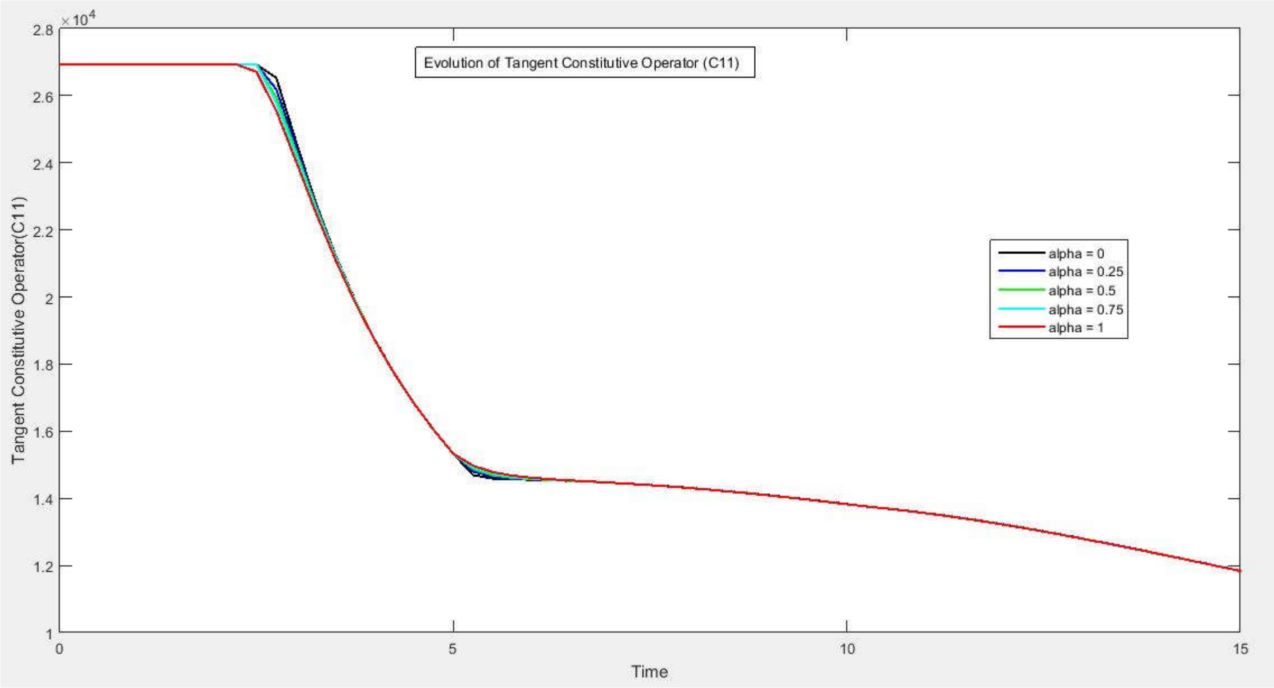

Figure E4: Evolution of Tangent Constitutive operator (C^{11}_{tang}) for different α

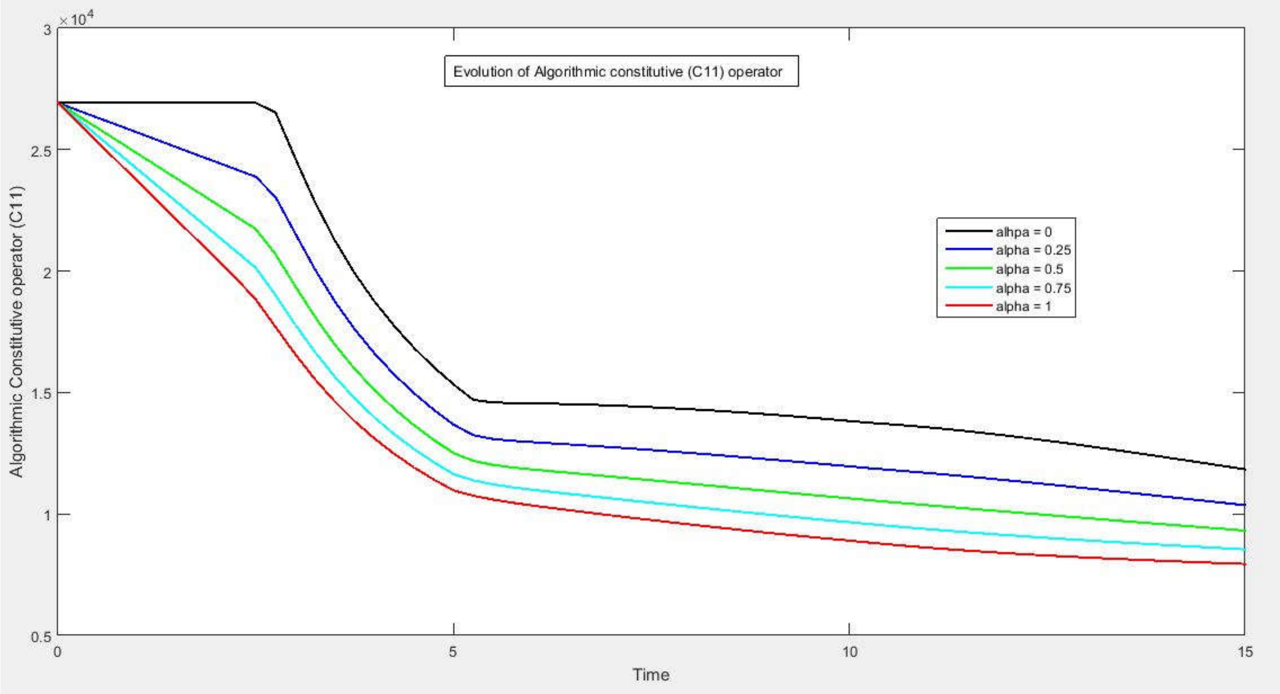

Figure E4: Evolution of Tangent Constitutive operator (C^{11}_{alg}) different α

Tangent Constitutive operator remains constant for different values of α as shown in figure E4, whereas algorithmic Constitutive operator decreases as α increases (Figure E5). From the above plots, we can make out that for α=0, $C^{11}_{tang\,=}$ $C^{11}_{alg.}$ We can conclude that $C^{vd}_{tang\,=}$ C^{vd}_{alg} for α=0.

Annexure 1

Main part of MATLAB codes for implementation of Non symmetric tension compression model and tension only model (rate independent case) are listed in this annexure.

Predictor step: Evaluation of $\tau_\varrho = \tau(\varepsilon_{n+1})$ (subroutine *Modelos_de_dano1.m*) is coded as follows for non symmetric case.

```
%**********************************start***********************************
elseif (MDtype==3) %*Non-symmetric

sigma_n = ce*eps_n1';
theeta1= (sigma_n(1).*(sigma_n(1)>0) + sigma_n(2).*(sigma_n(2)>0))
/(abs(sigma_n(1))+abs(sigma_n(2)));%Defining Theeta
rtrial= [theeta1+(1-theeta1)/n]*sqrt(eps_n1*ce*eps_n1')   ;

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

In the function *hplot* (subroutine *dibujar_criterio_dano1.m)* the following code is added for Non symmetric model Tension Compression model.

```
%**********************************start***********************************
elseif MDtype==3 %*Non-symmetric

    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
            theeta(i)=((m1(i).*(m1(i)>0) + m2(i).*(m2(i)>0))/(abs(m1(i))
            +abs(m2(i)))); % Defining Theeta
           radio(i)= q/(theeta(i)+(1-theeta(i))/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
    hplot =plot(s1,s2,tipo_linea);
%**********************************end*************************************
```

Predictor step: Evaluation of $\tau_\varrho = \tau(\varepsilon_{n+1})$ (subroutine *Modelos_de_dano1.m*) is coded as follows for tension only case.

```
%**********************************start***********************************
elseif (MDtype==2)  %*tension only model
```

```
rtrial= sqrt(eps_n1.*(eps_n1>0)*ce*eps_n1')   ;

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

In the function *hplot* (subroutine *dibujar_criterio_dano1.m)* the following code is added for Tension only model.

```
%**********************************start************************************
elseif MDtype==2  %*tension only model

    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).*(m1(i)>0) m2(i).*(m2(i)>0) 0
            nu*(m1(i)+m2(i)).*(nu*(m1(i)+m2(i))>0)]*ce_inv*[m1(i) m2(i) 0
            ...
            nu*(m1(i)+m2(i))]'); % Considered positive stresses

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

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

Annexure-2

The added code for Exponential Hardening/ Softening (subroutine *rmap_dano1.m*) is as follows.

```
%**********************************start************************************
else

        % Exponential Hardenig
        A=abs(H);
        if H>0
        q_inf=r0+(r0-zero_q);
        q_n1= q_inf-(q_inf-q_n)*exp(A*(1-r_n1/r_n));
        elseif H<0
        q_inf=zero_q;
        q_n1= q_inf-(q_inf-q_n)*exp(A*(1-r_n1/r_n));
        end
     end
%**********************************end**************************************
```

Annexure -3

Main part of added MATLAB code for implementation of integration algorithm (plane strain case) for the continuum isotropic visco-damage “symmetric tension compression” model is mentioned in this annexure.

Predictor step: Evaluation of $\tau_{n+\alpha} = (1- \alpha)*\tau(\varepsilon_n) + \alpha *\tau(\varepsilon_{n+1})$ (subroutine *Modelos_de_dano1.m*) is coded as follows for symmetric tension compression case.

```
%***********************************start*********************************
if (MDtype==1)       %* Symmetric
rtrial= sqrt(eps_n1*ce*eps_n1')   ;
te_n = sqrt(eps_n*ce*eps_n');    % strain (tau epslilon) at time step n
te_n_alpha = ((1-ALPHA_COEFF).*te_n) + (ALPHA_COEFF.*rtrial) ; % strain
(tau epslilon n plus aplha) at time ,n plus aplha
%***********************************end*********************************
```

The important part of added code is as follows (in the file *rmap_dano1.m).*

```
%***********************************start*********************************
 elseif vis == 1 %viscous model
    if (te_n_alpha > r_n)    %*   Loading
    fload=1;
      r_n1= ((eta-delta_t*(1-ALPHA_COEFF))*r_n + delta_t*te_n_alpha)
      /(eta+ALPHA_COEFF*delta_t);
      delta_r=r_n1-r_n;

    if hard_type == 0
        %  Linear
        q_n1= q_n+ H*delta_r;
    else
        % Exponential
        A=abs(H);
        if H>0
        q_inf=r0+(r0-zero_q);
        q_n1= q_inf-(q_inf-q_n)*exp(A*(1-r_n1/r_n));
        elseif H<0
        q_inf=zero_q;
        q_n1= q_inf-(q_inf-q_n)*exp(A*(1-r_n1/r_n));
        end
    End
%***********************************end*********************************
```

Annexure 4

The code for computation of tangent and algorithmic constitutive operators is as follows.

```
%***********************************start*********************************
%  Computing tangent operators
%  ****************
sigmab = ce*eps_n1'; % stress
C_tang_n1 =(1-dano_n1)*ce ; % tanget constitutive operator
C_tang_n1_11 = C_tang_n1(1,1);
% algorithmic constitutive operator
C_alg_n1 = C_tang_n1+(ALPHA_COEFF*delta_t*(H*r_n1-q_n1)
/(eta+(ALPHA_COEFF*delta_t))/rtrial/(r_n1^2)*(sigmab*sigmab'));
C_alg_n1_11 = C_alg_n1(1,1);
%***********************************end*********************************
```