UNIVERSITAT POLITÈCNICA
DE CATALUNYA
BARCELONATECH

Computational Solid Mechanics

Master's degree: Numerical Methods in Engineering

ASSIGNMENT N°1 : Damage Report

Professors: X. Oliver , A. Huespe

Student: Bonetti Calogero (Erasmus)

Contents

Introduction

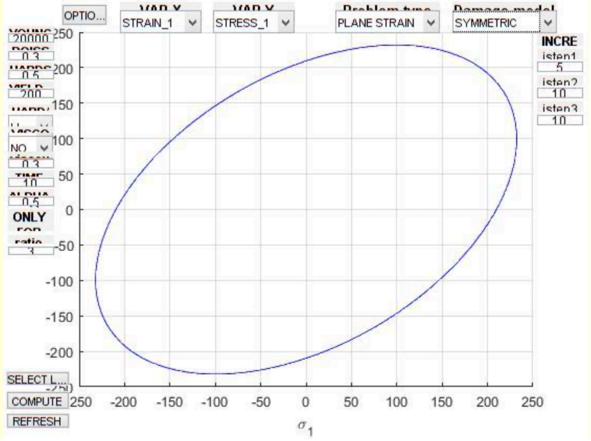
The following report shows the implementation of MatLab algorithm for the inviscid and viscous damage constitutive model for Solid Mechanics. The main aim of this report is to show the different models, changing parameters and making different loading paths, and discussing the results obtained. All the models implemented are using the planestrain hypothesis. The models shown in this topic use a Young modulus of 2000, a Yield stress of 200 and a Poisson ratio equal to 0.3.

Part I- Rate Independent Model

1.a) The supplied MATLAB code has been modified to include the Continuum- Isotropic damage "non-symmetric tension-compression damage" and the "tension only" damage models. The plots have been obtained for the following material properties-

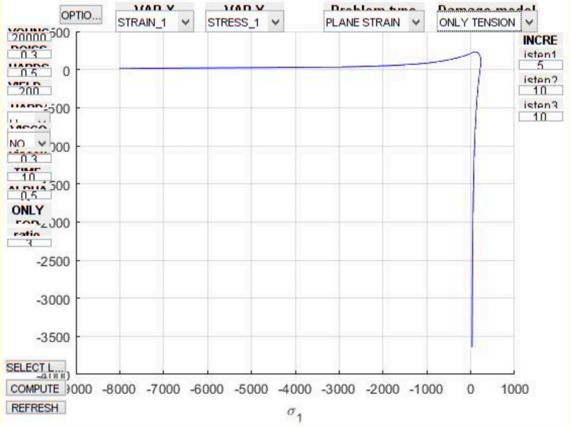
Young Modulus, E= 20000 MPa Yield Stress, σy= 200 MPa Poisson Ratio, ν= 0.3
Hardness, H= 0.5

Ratio of compression strength to tension strength, n=3
The following plots were obtained using the above properties-



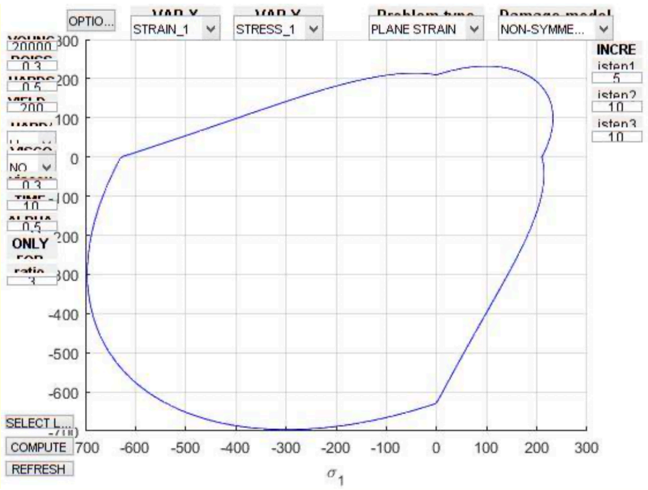
VAR X: STRAIN_1
VAR Y: STRESS_1
Problem type: PLANE STRAIN
Damage model: SYMMETRIC
20000
0.3
0.5
200
NO
0.3
10
0.5
ratio: 3
istep1: 5
istep2: 10
istep3: 10

Fig-1a: Symmetric model



VAR X: STRAIN_1
VAR Y: STRESS_1
Problem type: PLANE STRAIN
Damage model: ONLY TENSION
20000
0.3
0.5
200
NO
0.3
10
0.5
ratio: 3
istep1: 5
istep2: 10
istep3: 10

Fig-1b: Tension only model



OPTIO...

VAR X: STRAIN_1

VAR Y: STRESS_1

Problem type: PLANE STRAIN

Damage model: NON-SYMME...

20000

0.3

0.5

200

NO

0.3

10

0.5

ONLY

3

INCRE

isten1: 5

isten2: 10

isten3: 10

SELECT L...

COMPUTE

REFRESH

Fig-1c: Non-symmetric Tension-Compression Damage model

1-b) Linear and exponential models have been implemented for different cases considering hardening/softening (H>0 and H<0). The codes have been modified accordingly and the plots have been studied in the next section for various models.

1-c) The following cases have been considered for the analysis of different models

Case-1

$\Delta\sigma^{1}_{(1)}$=100MPa ; $\Delta\sigma^{2}_{(1)}$=0

$\Delta\sigma^{1}_{(2)}$=−500MPa ; $\Delta\sigma^{2}_{(2)}$=0

$\Delta\sigma^{1}_{(3)}$=900MPa ; $\Delta\sigma^{2}_{(3)}$=0

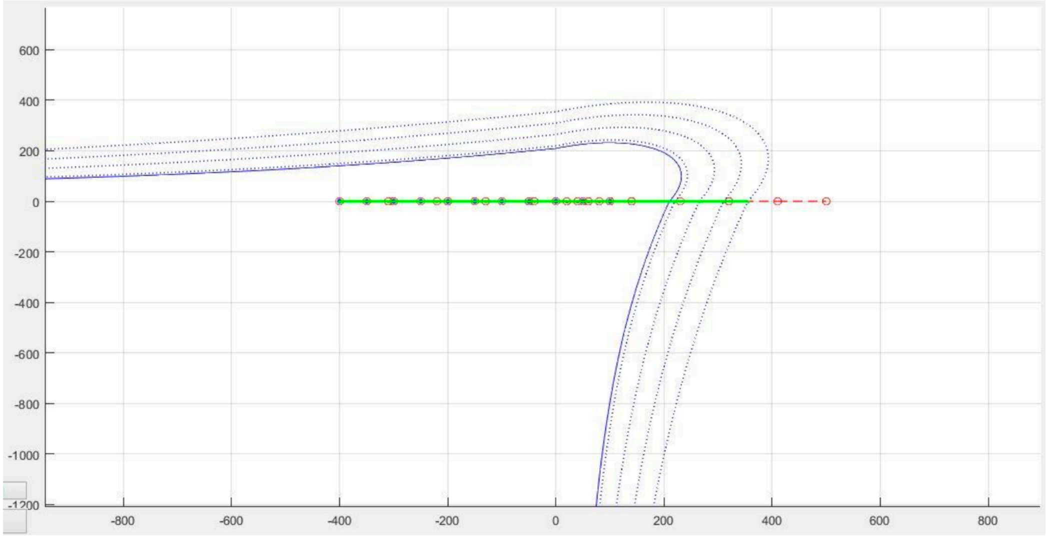


Fig-2a: σ_1 vs σ_2 for Tension only model (H=0.5)

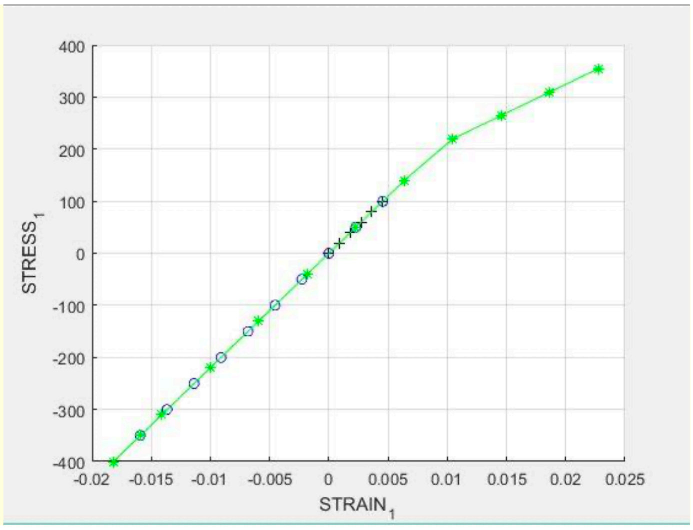


Fig- 2b: Linear Hardening for Tension only model

Black-crossed line shows uniaxial tensile loading, blue-circled line represents uniaxial tensile unloading/ compressive loading and the green-starred line represents uniaxial compressive unloading/tensile loading.

As can be seen from the plot, the loading stresses do not go beyond the yield stress, so the loading falls in the elastic regime which is reflected in the plot as all the points lie in a straight line with slope E. The tension only model does not take into account the yield stresses in the compression region so tensile unloading/ compressive loading is always elastic in this region (third quadrant of Fig- 2a), which also means that the material can only fail in tension.

We also note that during the next stage of uniaxial loading, as the stresses cross the yield stress the material undergoes hardening (H=0.5) and the stresses continue to increase with increasing strain. In case of linear hardening the curve increases in a linear fashion.

Next we take the case of linear softening for non-symmetric tension-compression damage model for the same values as the earlier model.

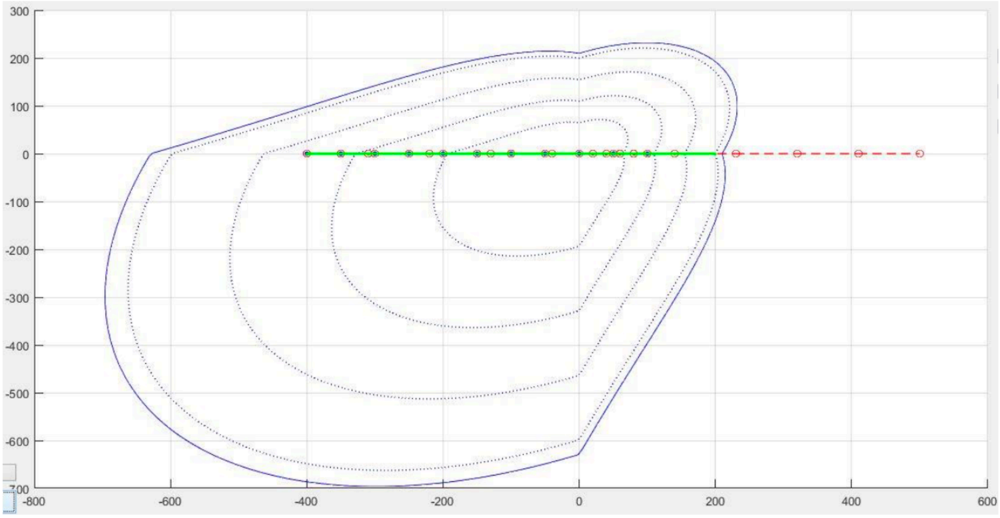


Fig-2c: σ_1 vs σ_2 for Non-symmetric tension-compression damage model (H=-0.5)

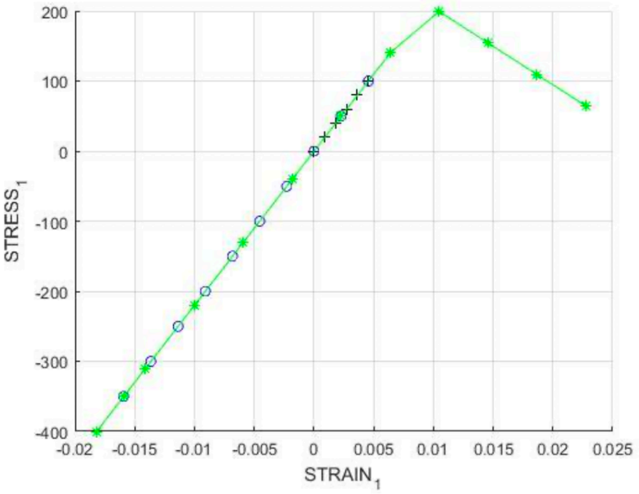


Fig- 2d: Linear Softening for Non-symmetric tension-compression model

As in the previous case, the plot is consistent for the first stage of elastic loading and then tensile unloading/ compressive loading. Next during the subsequent loading we observe that as the stresses cross the material yield stress, with increasing strain the stresses decrease which is the result of softening (H<0) and this behavior is as expected.

Case-2

$\Delta\sigma^1{}_{(1)}$=100MPa ; $\Delta\sigma^2{}_{(1)}$=0

$\Delta\sigma^1{}_{(2)} = -300$ MPa ; $\Delta\sigma^2{}_{(2)}$ = -300 MPa

$\Delta\sigma^1{}_{(3)}$=600MPa ; $\Delta\sigma^2{}_{(3)}$=600MPa

Linear hardening and softening have been analyzed for this case for various models.

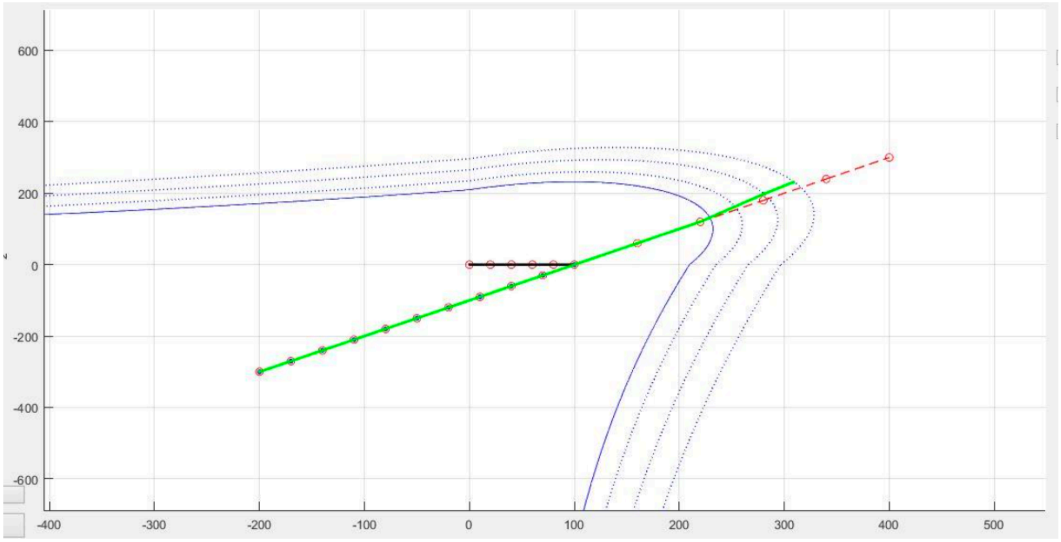


Fig-3a: σ_1 vs σ_2 for Tension only model (H=0.5)

Just as in the previous case we observe in the above plot that the initial uniaxial tensile loading gives an elastic response as the stresses do not cross the yield stress. Also in the case of biaxial tensile unloading/ compressive loading, the response is still elastic irrespective of the value of the loads as the tension only model do not account for the yield stresses in the compressive regime. Next during biaxial loading the domain expands as the stresses cross the yield stress boundary.

This is evident from figure 3b, that as the stresses cross the yield boundary during the last loading step, the stresses increase with the strain causing a hardening effect on the material and the internal variable (q) evolves accordingly.

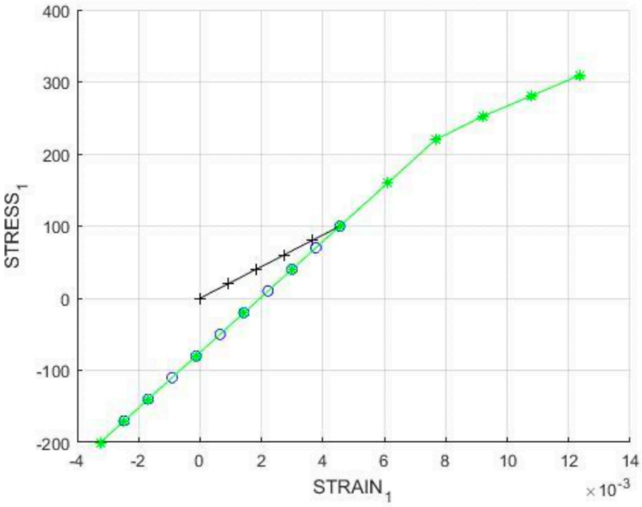


Fig- 3b: Linear Hardening for Tension only model

Now we consider the case of non-symmetric tension-compression model to represent the case of linear softening.

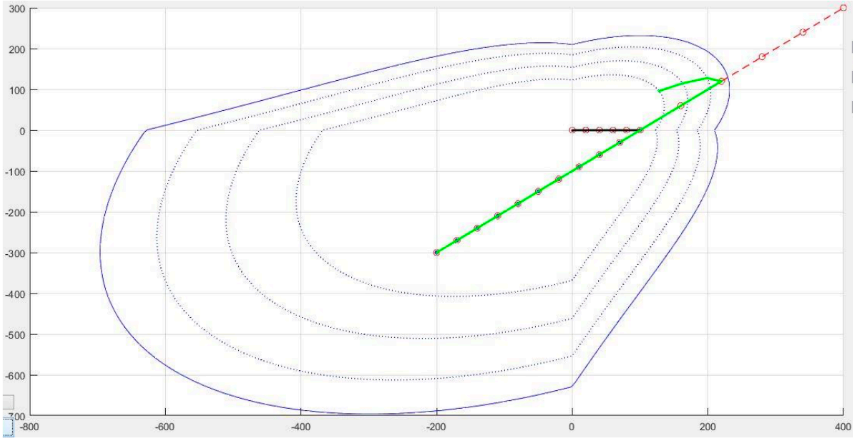


Fig-3c: σ_1 vs σ_2 for Non-symmetric tension-compression damage model (H=-0.5)

For this case also we observe that the initial loading and unloading fall within the elastic limits and hence, we get purely elastic response. However as the stresses exceed the yield limit, the stresses begin to drop with the evolution of internal variable as the strain increases, as can be seen from the following stress strain curve highlighting the linear softening for H=-0.5.

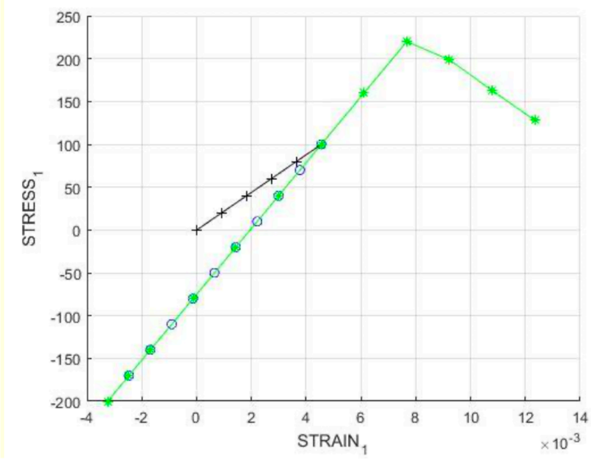


Fig- 3d: Linear Softening for Non-symmetric tension-compression model

Case-3

$\Delta\sigma^{1}{}_{(1)}$=100MPa ; $\Delta\sigma^{2}{}_{(1)}$=100MPa

$\Delta\sigma^{1}{}_{(2)}$ = −300 MPa ; $\Delta\sigma^{2}{}_{(2)}$ = -300 MPa

$\Delta\sigma^{1}{}_{(3)}$=600MPa ; $\Delta\sigma^{2}{}_{(3)}$=600MPa

Exponential hardening and softening have been analyzed for this case for various models.

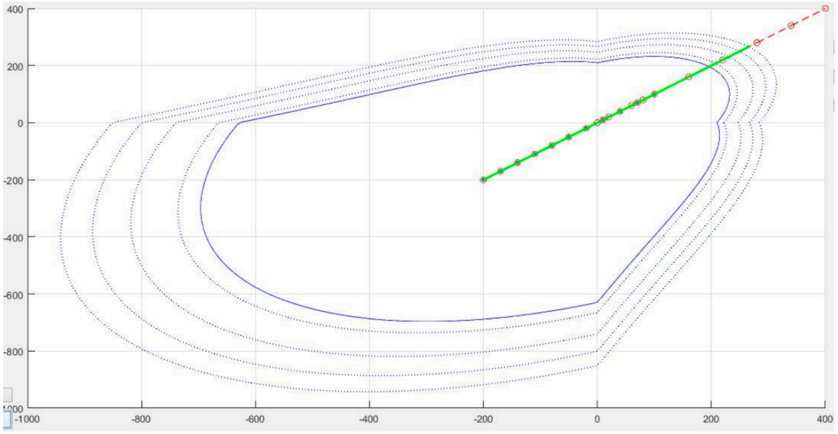


Fig-4a: σ_1 vs σ_2 for Non-symmetric tension-compression damage model (H=0.5)

Here we analyze exponential hardening for non-symmetric tension-compression model. We can see from the plot that the biaxial tensile loading falls within the elastic region and therefore exhibits elastic response, straight line with slope E. The biaxial tensile unloading/ compressive loading also does not exceed the elastic threshold and therefore shows elastic behavior.

However, during the subsequent stage of biaxial compressive unloading/ tensile loading, hardening effect is observed as the stresses begin to grow with the increase in strains. This physical significance of this behavior can be interpreted as the material trying to increase it's yield stress limit to try and account for this hardening effect.

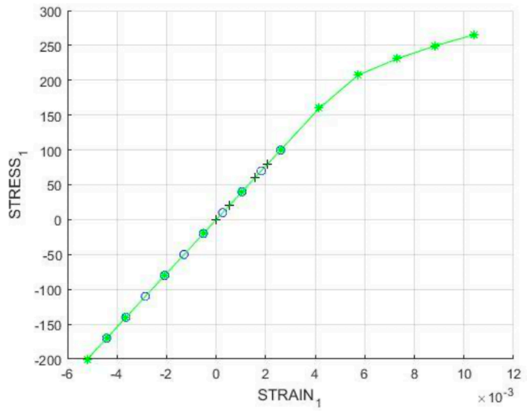


Fig- 4b: Exponential Softening for Non-symmetric tension-compression model

Now we observe the effect of exponential softening for Tension only model. It behaves in a similar fashion for the initial biaxial tensile loading and biaxial tensile unloading/compressive loading. However during the stage of biaxial compressive unloading/tensile loading we find that the stresses decrease with the increasing strain which results due to the softening property of the material as can be seen from the figure 4d. However the stresses outside the yield limit follows an exponential profile because of considering exponential nature of softening.

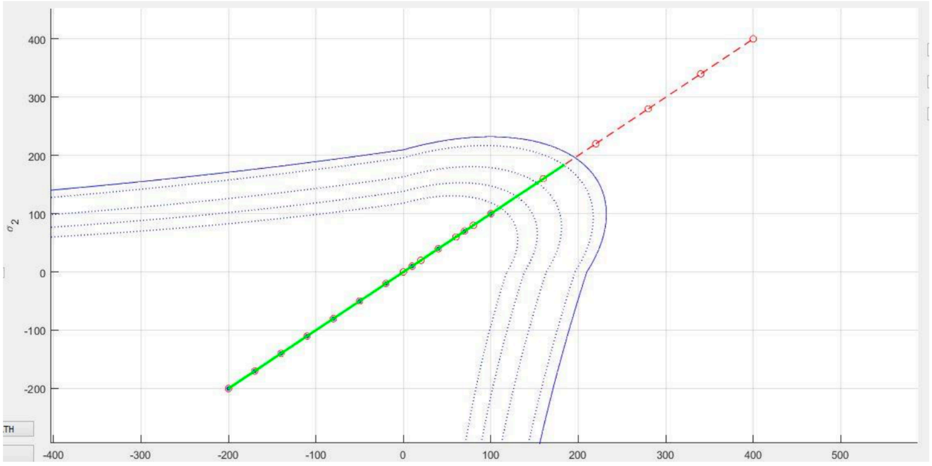


Fig-4c: σ_1 vs σ_2 for Tension only model (H=-0.5)

The exponential nature of the softening is even more evident from the stress-strain plot given below.

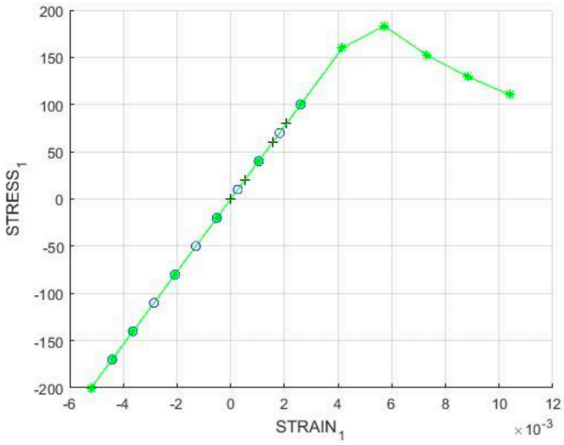


Fig- 4d: Exponential softening for Tension only model

Part II- Rate Dependent Model

In this section we consider time as an independent variable which means that the stress tensor can still change even when the strain tensor remains constant because of its dependence on time.

In the subsequent sections we have implemented the MATLAB code to account for Rate- dependent model for plane strain case of isotropic visco-damage "symmetric tension-compression" model.

2-a) Here we check the correctness of the implementation by considering dependency on various parameters. We consider an uniaxial loading/unloading and the following parameters for the subsequent simulations.

$\Delta\sigma^1_{(1)}$=100MPa ; $\Delta\sigma^2_{(1)}$=0

$\Delta\sigma^1_{(2)}$=−200MPa ; $\Delta\sigma^2_{(2)}$=0

$\Delta\sigma^1_{(3)}$=600MPa ; $\Delta\sigma^2_{(3)}$=0

Young Modulus, E= 20000 MPa

Yield Stress, σy= 200 MPa

Poisson Ratio, ν= 0.3

Hardness, H= 0.5
Ratio of compression strength to tension strength, n=3

Case-1: Different viscosity parameters η

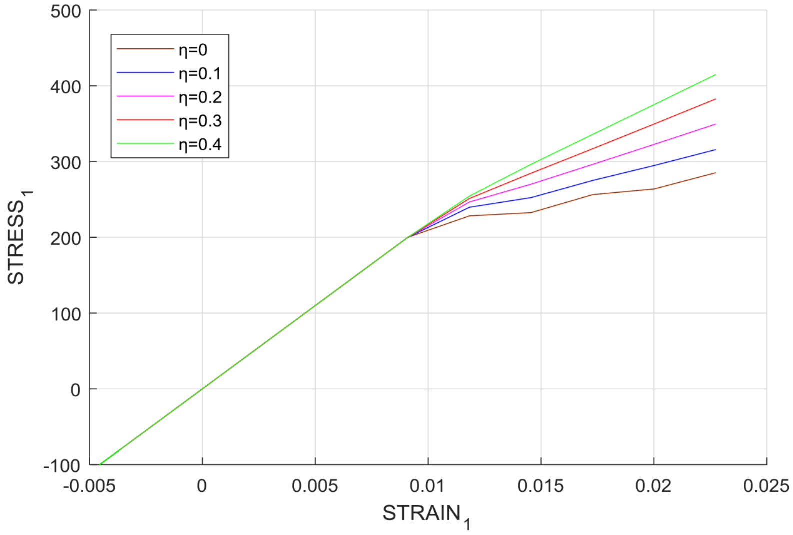


Fig-5a: Dependency of stress-strain curve on the variation of η

For uniaxial loading /unloading cycle, the above plot is obtained. It is evident from the stress- strain curve that the stresses outside the elastic region increase with the increase in viscosity which is consistent with the damper model which requires a high value of force for higher values of viscosity. And in accordance with the theory we find the elastic region remains unaffected by the effects of viscosity.

Case-2: Different strain rate values

For this case we chose to vary the strain rate by assigning different values of total time to carry out the simulation and then analyzing the effect on the resulting stress with the variation of time. It is clear from the figure obtained in 5b, that there is no variation in stress within the elastic limit as was the observation in the earlier case of variation of viscosity. However we observe that once the stresses cross the yield stress for the material, it starts deforming non-elastically and the stresses show variation with strain rates. Higher values of strain rates yield higher values of stresses.

This could probably be attributed to the fact that a slow strain rate or in other words allowing the strains to slowly increase over time produces a softening effect on the material and as a result the stresses begin to decrease.

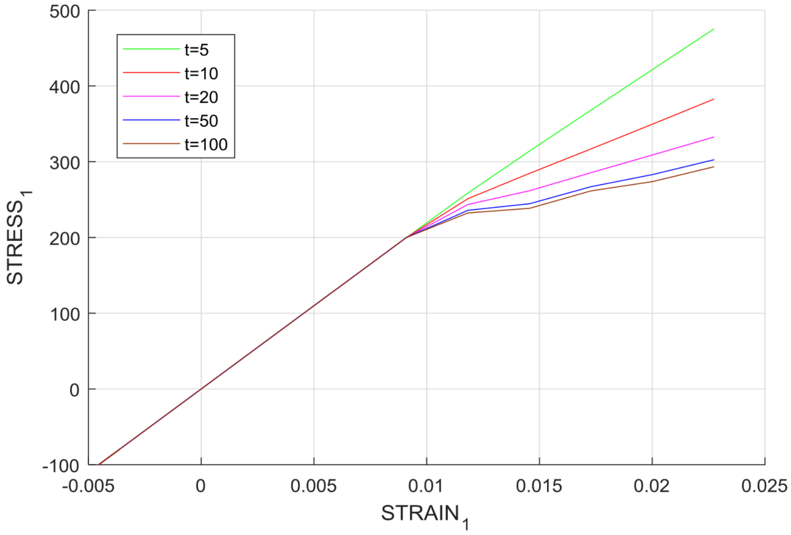


Fig-5b: Dependency of stress-strain curve on the variation of total time

Case-3: Different α values

The α time integration method gives different numerical methods for different values of α. The following figure-5c illustrates how the stress-strain relationship varies for different numerical methods or for different values of α.

It is evident from the plot that stresses obtained are different for different values of α. α=0 α=0.25, α=0.5, α=0.75 and α=1. These deviations result from the different discretization schemes employed by these methods. Although explicit schemes have low computational costs but they are not unconditionally stable which can give rise to unreliable solutions. While all the other methods here are first order accurate, Crank-Nicholson (α=0.5) has second order accuracy and is unconditionally stable. So this method for α=0.5 is expected to give most reliable results in most situations compared to the other methods.

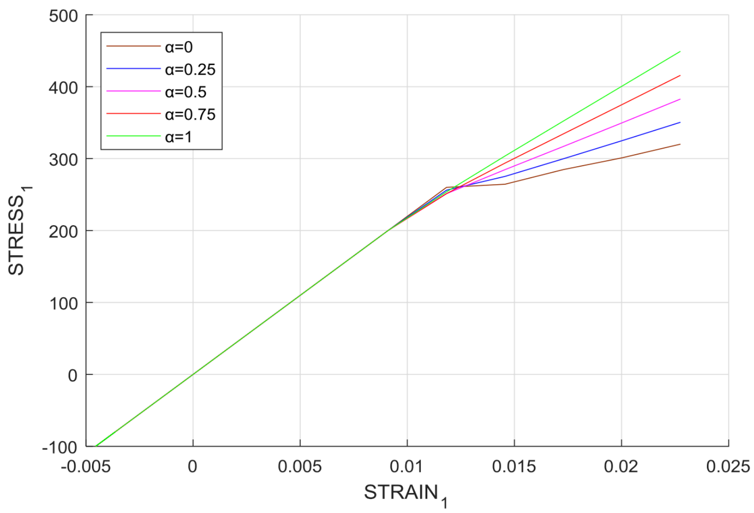


Fig-5c: Dependency of stress-strain curve on the variation of total time

2-b) Effects of variation of α on the evolution of C_{11} component of the tangent and algorithmic constitutive operators

In this section we analyze how the C_{11} components vary with the variation of α. The following values were considered in this analysis-

$\Delta\sigma^1$ (1) =150MPa ; $\Delta\sigma^2$ (1) =0

$\Delta\sigma^1$(2) =150MPa ; $\Delta\sigma^2$ (2) =0

$\Delta\sigma^1$ (3) =150MPa ; $\Delta\sigma^2$ (3)=0

Young Modulus, E= 20000 MPa

Yield Stress, σy= 200 MPa

Poisson Ratio, ν= 0.8

Hardness, H=- 0.5

Figure 6a shows the variation of C_{11} tangent component with alpha for the time of simulation. We can see that it varies with different methods. The plot also highlights the presence of instabilities for α<0.5 which is to be expected as these methods are explicit and are conditionally stable leading to fluctuations in the result which is not the case for the implicit methods. We also find that the values of C_{11} tangent component is gradually falling with time which can be attributed to the fact that C_{tan}=(1-d)C. So with the increase in damage, which increases with time, the value of Tangent operator falls.

Similarly, the C_{11} component of algorithmic operator also follows a gradual fall in its values with the increase in time just as the case with Tangent operator. This can be clearly seen from figure 6b. We also find huge oscillations in the result which is to be expected for α<0.5 as the methods are explicit and conditionally stable. The solutions are however stable for methods with α>0.5.

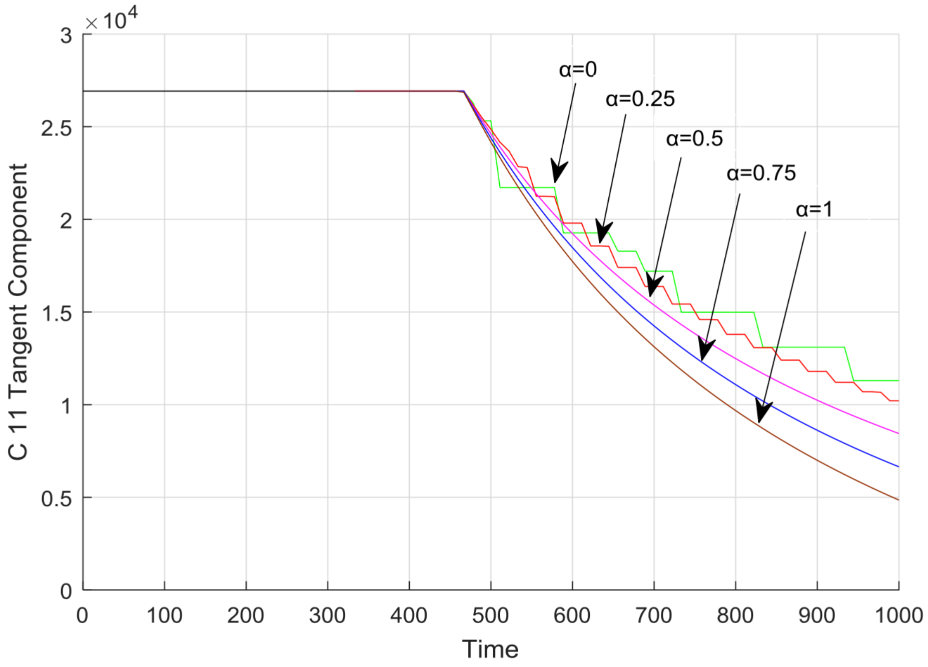


Fig-6a: Variation of C11 tangent Component with α

An interesting aspect to note is that both the curves for tangent operator and algorithmic operator coincide for α=0 which is in accordance with the theory. This can also be viewed as a verification for the correctness of the code implemented.

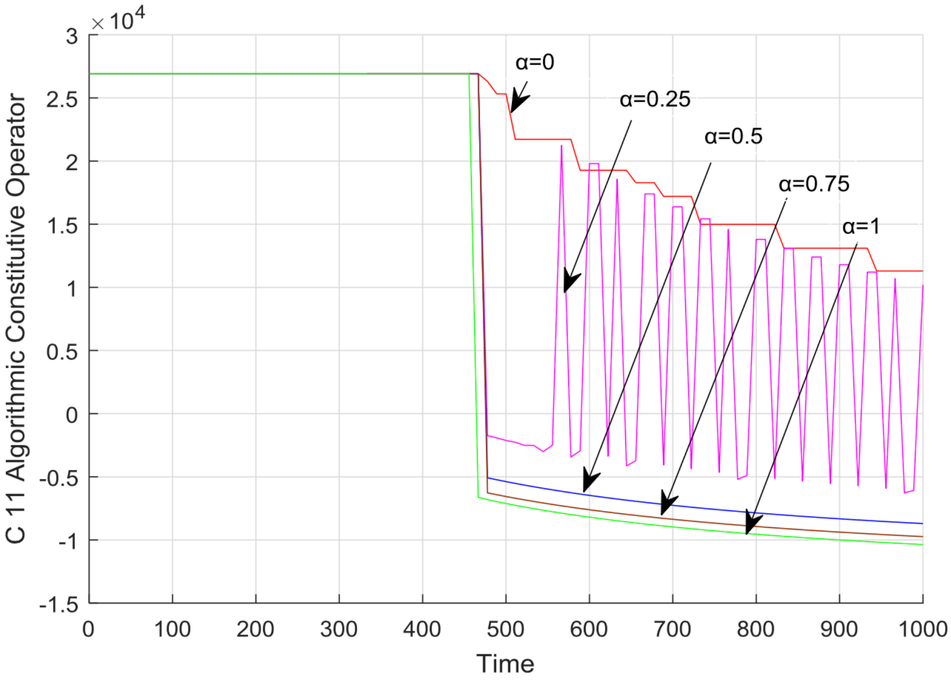


Fig-6b: Variation of C11 Algorithmic Constitutive Operator with α

3 – Annex – MatLab Modified Routines

```
clc
clear all
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% 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; %FOR EXPONETIAL CASE H is q'(r0)= Ho = H
% 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 = 1;
% Ratio compression strength / tension strength
% ---------------------------------------------
n = 2 ;
% SOFTENING/HARDENING TYPE
% ------------------------
HARDTYPE = 'LINEAR' ; %{LINEAR,EXPONENTIAL}
% VISCOUS/INVISCID
% ------------------------
VISCOUS = 'YES' ;
% Viscous coefficient ----
% ------------------------
eta = 3;
% TimeTotal (initial = 0) ----
% ------------------------
TimeTotal = 12;
% Integration coefficient ALPHA
% ------------------------
ALPHA_COEFF = 1;
% Points ---------------------------
% ----------------------------------
nloadstates = 3 ;
SIGMAP = zeros(nloadstates,2) ;
SIGMAP(1,:) =[500 0];
SIGMAP(2,:) =[1000 0];
SIGMAP(3,:) =[1500 0];
% Number of time increments for each load state
% --------------------------------------------------------
istep = 10*ones(1,nloadstates) ;
% VARIABLES TO PLOT
vpx = 'STRAIN_1' ; % AVAILABLE OPTIONS: 'STRAIN_1', 'STRAIN_2'
%                     '|STRAIN_1|', '|STRAIN_2|'
% 'norm(STRAIN)', 'TIME'
vpy = 'STRESS_1';            % 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,C11t,C11a]=damage_main(Eprop,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))
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) ;
```

```
function
[sigma_v,vartoplot,LABELPLOT,TIMEVECTOR,C11t,C11a]=damage_main(Eprop,ntype,istep,strain,MDty
pe,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);
alpha = Eprop(8);

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
%SETTING VISCOSITY TO ZERO FOR RATE INDEPENDENT ANALISYS
if viscpr ~= 1
    Eprop(7) = 0;
else
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_n  = zeros(mstrain,1); %STRAIN STATE AT t_n
eps_n1  = zeros(mstrain,1);%STRAIN STATE AT t_n+1
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)
%INITIALIZATION C11 COMPONENT OF TANGENT AND ALGORITHIM CONSTITUTIVE TENSOR
C11a =  ce(1,1);
C11t = ce(1,1);

for  iload = 1:length(istep)
    % Load states
    for iloc = 1:istep(iload)
        i = i + 1 ;
        dt = delta_t(iload);
        TIMEVECTOR(i) = TIMEVECTOR(i-1)+ dt;
        % Total strain at step "i"
        % ---------------------------------
        eps_n = strain(i-1,:);                %STRAIN STATE at t_n
        eps_n1 = strain(i,:);                 %STRAIN STATE at t_n+1
        %************************************************************************
        %*      DAMAGE MODEL
        % %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        [sigma_n1,hvar_n,aux_var,Ct,Ca]                                                     =
rmap_dano1(eps_n,eps_n1,hvar_n,Eprop,ce,MDtype,n,dt);
        % 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 ;
        %C11 COMPONENT OF TANGENT AND ALGORITHIM CONSTITUTIVE TENSOR
        C11a(i) = Ca(1,1);
        C11t(i) = Ct(1,1);
        % 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
```

```
functio
 [sigma_n1,hvar_n1,aux_var,Ct,Ca] =rmap_dano1(eps_n,eps_n1,hvar_n,Eprop,ce,MDtype,n,dt)

%*************************************************************************************
%*                                         *
%*           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);
eta = Eprop(7);
alpha = Eprop(8);
%************************************************************************************
%************************************************************************************
%*       initializing                                                     %*
 r0 = sigma_u/sqrt(E);
 zero_q= 1.d-6*r0;
 %SUPERIOR LIMIT FOR q infinity
 inf_q = 1.d6;
 %EXPONETIAL MODEL CONSTANT EXPOENT
 A = 1;
% if(r_n<=0.d0)
%     r_n=r0;
%     q_n=r0;
% end
%*********************************************************************
%*********************************************************************
%*       Damage surface
[rtrialn rtrialn1] = Modelos_de_dano1 (MDtype,ce,eps_n,eps_n1,n);
rtrial = alpha*rtrialn1+(1-alpha)*rtrialn; %STRAIN RADIUS AT t_n+alpha
%*********************************************************************
%*********************************************************************
%*   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;
    %THETA INTEGRATION METHOD
    r_n1 = (r_n*(eta-dt*(1-alpha))+dt*rtrial)/(eta+alpha*dt);
    delta_r = r_n1-r_n;
    if hard_type == 0 %LINEAR SOFTENING/HARDENING LAW
        q_n1= q_n+ H*delta_r;
        if(q_n1<zero_q) %SOFTENING
        q_n1=zero_q;
        elseif (q_n1 > inf_q)%HARDENING
        q_n1=inf_q;
        end
        Hn1 = H; %H PARAMETER at t_n+1
    else %EXPONETIAL SOFTENING/HARDENING LAW
       Ho = H; %H  PARAMETER at t_0
       inf_q = (Ho/A+1)*r0; %
       q_n1 = q_n +(inf_q-r0)*(exp(A*(1-r_n/r0))-exp(A*(1-r_n1/r0)));
       Hn1 = A*((inf_q-r0)/r0)*exp(A*(1-rn1/r0)); %H PARAMETER at t_n+1
    end
    dano_n1   = 1.d0-(q_n1/r_n1);
    %CONSTITUTIVE TENSORS
    Ca = (1.d0-dano_n1)*ce + ((alpha*dt)/(eta+alpha*dt))*(1/rtrialn1)*...
        ((Hn1*r_n1-q_n1)/(r_n1^2))*((ce*eps_n1')*(ce*eps_n1')');
    Ct = (1.d0-dano_n1)*ce;
else
    %*     Elastic load/unload
    fload=0;
    r_n1= r_n  ;
    q_n1= q_n  ;
    dano_n1   = 1.d0-(q_n1/r_n1);
    %CONSTITUTIVE TENSORS
    Ca = (1.d0-dano_n1)*ce;
    Ct = (1.d0-dano_n1)*ce;
end
%  Computing stress
%  ****************
sigma_n1  = (1.d0-dano_n1)*ce*eps_n1';
%TANGENT CONSTITUTIVE TENSOR
%*************************************************************************
%*************************************************************************
%* 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;
%*************************************************************************
```

```
function [rtrialn rtrialn1] = Modelos_de_dano1 (MDtype,ce,eps_n,eps_n1,n)
%**************************************************************************************
%*          Defining damage criterion surface                                        %*
%*                                                                                   %*
%*
%*                          MDtype=  1      : SYMMETRIC                              %*
%*                          MDtype=  2      : ONLY TENSION                           %*
%*                          MDtype=  3      : NON-SYMMETRIC                          %*
%*                                                                                   %*
%*                                                                                   %*
%* OUTPUT:                                                                           %*
%*                          rtrial                                                   %*
%**************************************************************************************
%*************************************************************************************
if (MDtype==1)      %SYMMETRIC
rtrialn = sqrt(eps_n*ce*eps_n');     %STRAIN RADIUS at t_n
rtrialn1 = sqrt(eps_n1*ce*eps_n1'); %STRAIN RADIUS at t_n+1

elseif (MDtype==2)  %ONLY TENSION
% ELASTIC STRESS VECOTOR
sigma_n =ce*eps_n';
sigma_n1 =ce*eps_n1';
% MACAULLEY BRACKET
sigma_n = sigma_n.*(sigma_n > 0);
sigma_n1 = sigma_n1.*(sigma_n1 > 0);
%DAMAGE SURFACE (TRIAL STATES)
rtrialn = sqrt(sigma_n'*eps_n');     %STRAIN RADIUS at t_n
rtrialn1 = sqrt(sigma_n1'*eps_n1'); %STRAIN RADIUS at t_n+1

elseif (MDtype==3)  %NON-SYMMETRIC
% ELASTIC STRESS VECTOR
sigma_n =ce*eps_n';
sigma_n1 =ce*eps_n1';
%THETA DEFINITION
%macaulley bracket
Mn = sigma_n.*(sigma_n > 0);
Mn1 = sigma_n1.*(sigma_n1 >0);
Smn = sum(Mn);
Smn1 = sum(Mn1);
Sn = sum(abs(sigma_n));
Sn1 = sum(abs(sigma_n1));
Thetan = Smn/Sn;
Thetan1 = Smn1/Sn1;
%DAMAGE SURFACE (TRIAL STATES)
Kn = Thetan+(1-Thetan)/n;
Kn1 = Thetan1+(1-Thetan1)/n;
rtrialn = Kn*sqrt(eps_n*ce*eps_n');     %STRAIN RADIUS at t_n
rtrialn1 = Kn1*sqrt(eps_n1*ce*eps_n1');%STRAIN RADIUS at t_n+1
end
%**************************************************************************************
return
```

```
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  % SYMMETRIC 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) 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  % COMPRESSION ONLY (TENSION) MODEL
    tetha=[-0.5*pi/2:0.01:0.75*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
        S = [m1(i) m2(i) 0 nu*(m1(i)+m2(i))];
        %MACAULLEY BRACKET
        S = S.*(S>0);
        radio(i)= q/sqrt(S*ce_inv*S');
        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];
    %***************************************************************************************
    %* 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
        S = [m1(i) m2(i) 0 nu*(m1(i)+m2(i))];
        %MACAULLEY BRACKET
        M = S.*(S>0);
        %SUMMATTION NUMERATOR
        Sm = sum(M);
        %SUMMATTION DENOMINTOR
        Smm = sum(abs(S));
        %THETA
        Theta = Sm/Smm;
        K = Theta+(1-Theta)/n;
        radio(i)= q/(K*sqrt(S*ce_inv*S'));
        s1(i)=radio(i)*m1(i);
        s2(i)=radio(i)*m2(i);
    end
    hplot =plot(s1,s2,tipo_linea);
end
%***************************************************************************************
%***************************************************************************************
return
```