UNIVERSITAT POLITÈCNICA DE CATALUNYA
MSc IN NUMERICAL METHODS IN ENGINEERING
COMPUTATIONAL SOLID MECHANICS

Assignment 1

Gabriel Valdés Alonzo

April 7th, 2017

Problem statement

The objective of this assignment is to implement a code for computing the basic plots for the Continuum Damage Theory for Elastoplastic materials. The code, implemented by professor Hernández, is partially supplied as part data in the homework, while the idea is to complete the code in order to obtain full functionality of the plain strain case.
The code initially is implemented for the symmetric model with a linear Hardening/Softening Law for strain-rate independent models. In this assignment, it is asked to develop the necessary changes to include the Tension-only model and the Non-symmetric Tension-Compression model, as well as an exponential Hardening/Softening Law. In a second instance, it is asked to provide functionality to implement the Rate-dependent models and viscosity that should work together with the previous models and Hardening/Softening Laws.

Rate independent models

Case 1

The cases are analyzed with the following constants, emulating loosely the values corresponding to steel:

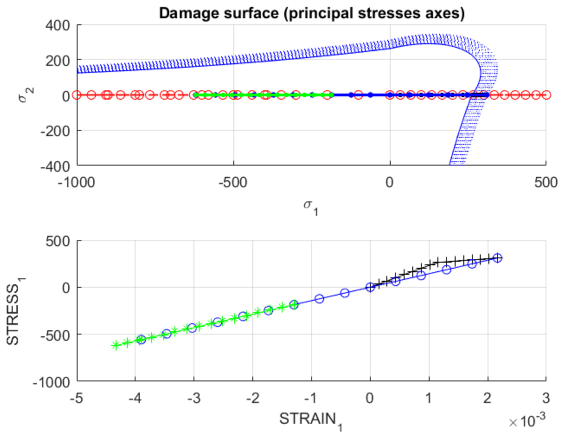
$$E = 210\ GPa$$
$$\nu = 0.3$$
$$H = A = \pm 0.2 \text{ (linear/exponential constants)}$$
$$\sigma_y = 250\ MPa$$
$$n = 3 \text{ (non-symmetric case)}$$

The three proposed cases are analyzed with both hardening/softening laws for the two models implemented in the code and compared against the symmetric one. Different values of increments are chosen in order to display better the properties of the different models.
The first case to analyze is a uniaxial stress state, proposed as

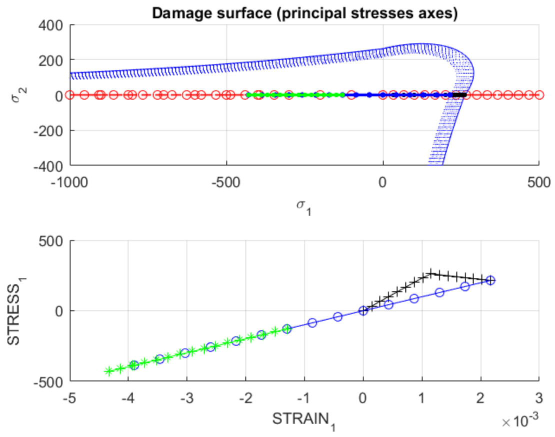
$$\Delta\bar{\sigma}_1^{(1)} = \alpha; \quad \Delta\bar{\sigma}_2^{(1)} = 0$$
$$\Delta\bar{\sigma}_1^{(2)} = -\beta; \Delta\bar{\sigma}_2^{(2)} = 0$$
$$\Delta\bar{\sigma}_1^{(3)} = \gamma; \quad \Delta\bar{\sigma}_2^{(3)} = 0$$

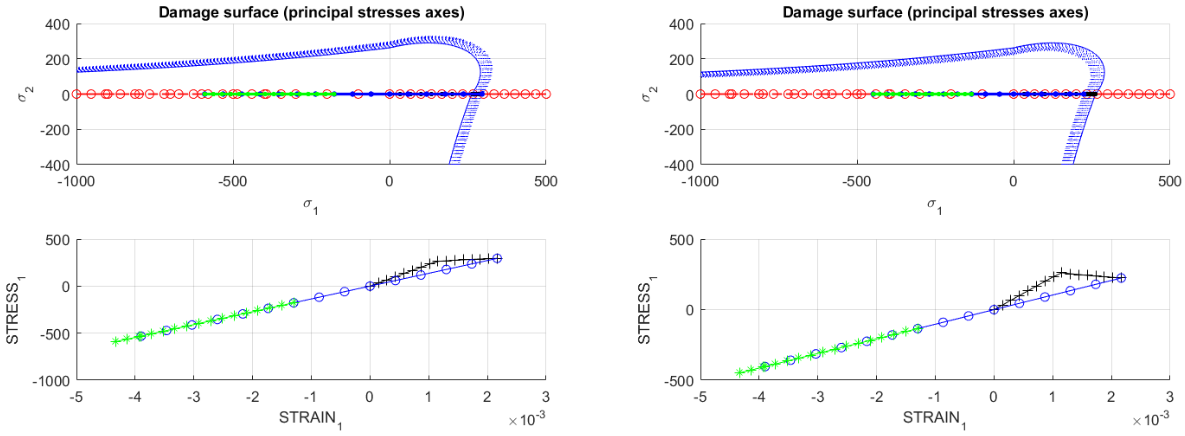
For this case the values are chosen as α=500 MPa, β=-1500 MPa and γ=700 MPa. It is worth noting that these values are in general too big for any steel piece, but are chosen in order to have a better illustration of the stress-strain curves that are displayed. Considering 15 steps per increment, the results are shown below for only-tension and non-symmetric cases:

a) Linear Hardening



b) Linear Softening

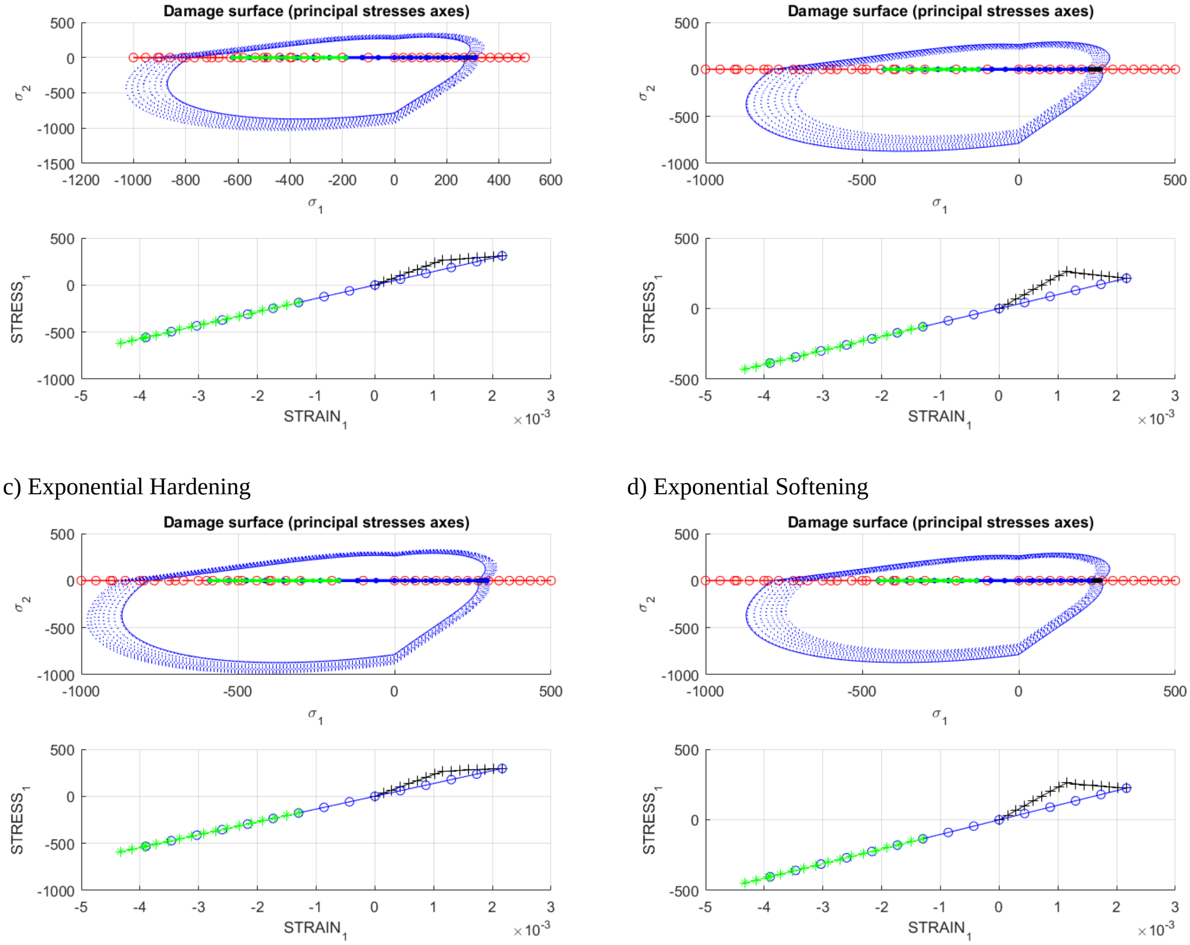




c) Exponential Hardening

d) Exponential Softening

Figure 1: Tension-only model, case 1



a) Linear Hardening

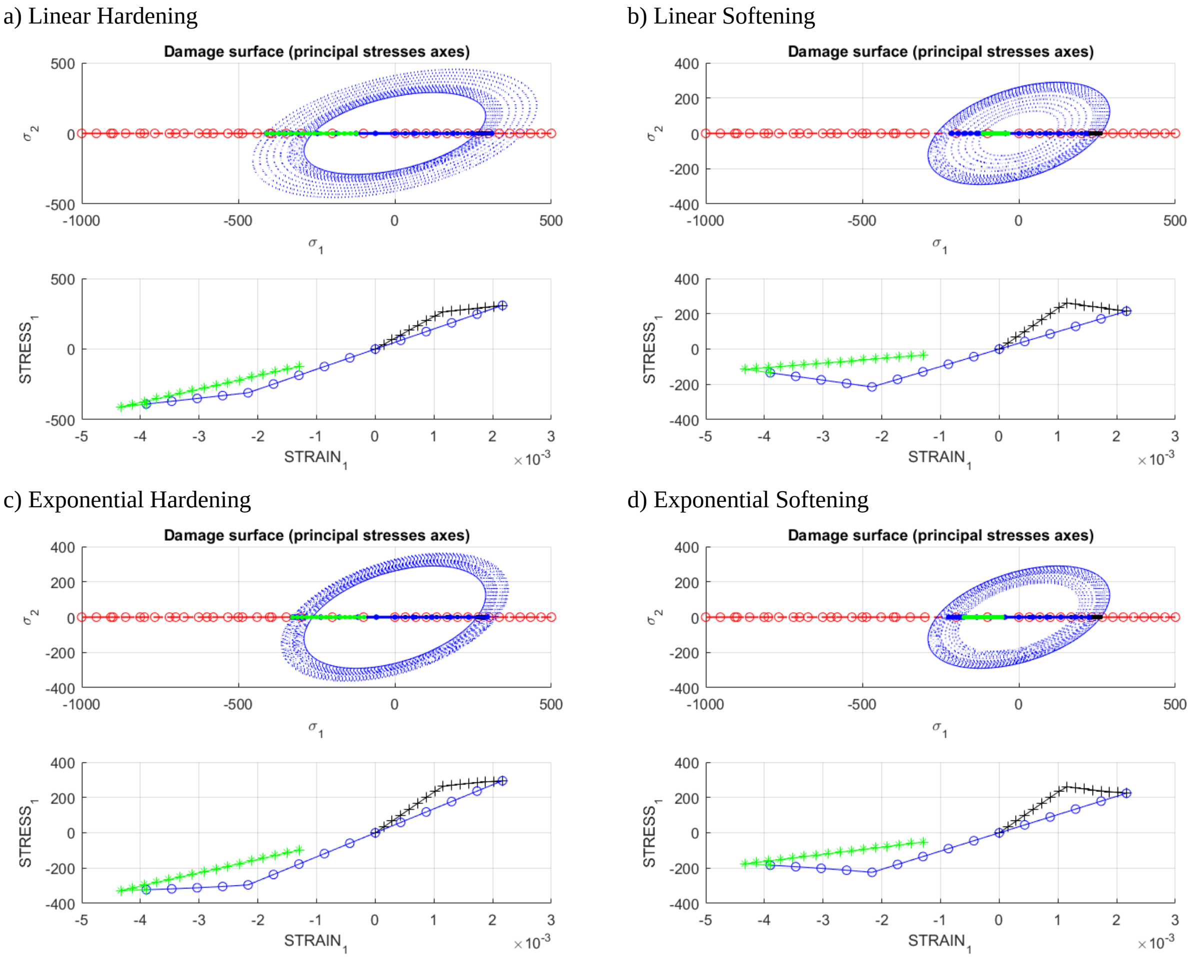
b) Linear Softening

c) Exponential Hardening

d) Exponential Softening

Figure 2: Non-symmetric model, case 1

The symmetric case is shown below:



a) Linear Hardening

b) Linear Softening

c) Exponential Hardening

d) Exponential Softening

Figure 3: Symmetric model, case 1

Figures 1 and 2 show that the model is working as planned. Stress paths were chosen specifically to address inelastic effects in combination with elastic one, in order to see if the curves of stress-strain display the values that should. For both Tension-only and Non-symmetric cases the first load path goes into the inelastic zone, activating the hardening or softening law as required. For compression, Tension-only model will remain always in elastic zone, which can be seen from the blue and green lines keeping the same discharge and reload slope. In the case of Non-symmetric, it is seen that the value of n provided gives a big enough range of elastic domain, enough for this model to remain in the elastic part for both discharge and compression loading. All this can be inferred by the fact that the stress-strain curves are exactly the same for both cases.

For the sake of analysis, the symmetric case is shown, which displays the behavior expected: for both tensile load and compression load the stresses go into inelastic zone, which can be seen in the stress-strain curve as a bilinear effect. Unloading also show the interesting effect that unloading always go back and pass through the origin, which is due to the fact that we have a uniaxial effect. Regarding the hardening/softening laws, it can be seen that the difference between them is not too much relevant, since for both linear and exponential laws the results are similar in value. However, when the same case is considered between hardening and softening the effect can be noticed both in the stress space plot as well as in the stress strain curve, and this is better appreciated in the symmetric case. Softening implies a loss of strength in the inelastic domain, which shifts the slope downwards and thus provokes a change in the discharging slope, which ultimately affects how the reload is done. This can be seen for example in figure 3.c and figure 3.d, where for softening there is a “widening” of the curve, as opposed to hardening where the slope of load and reload are more similar than the softening case.

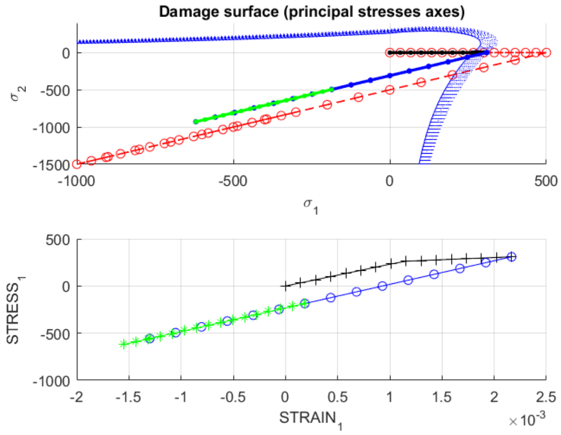
Case 2

The second case to analyze is a uniaxial load with biaxial discharge and reload, proposed as

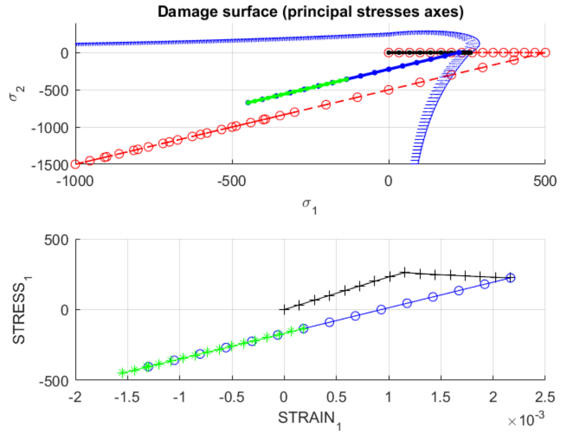
$$\begin{aligned}
&\Delta\bar{\sigma}_1^{(1)} = \alpha; \quad \Delta\bar{\sigma}_2^{(1)} = 0\\
&\Delta\bar{\sigma}_1^{(2)} = -\beta; \Delta\bar{\sigma}_2^{(2)} = -\beta\\
&\Delta\bar{\sigma}_1^{(3)} = \gamma; \quad \Delta\bar{\sigma}_2^{(3)} = \gamma
\end{aligned}$$

For this case the values are chosen as α=500 MPa, β=-1500 MPa and γ=700 MPa. Again considering 15 steps per increments, the results are shown below:

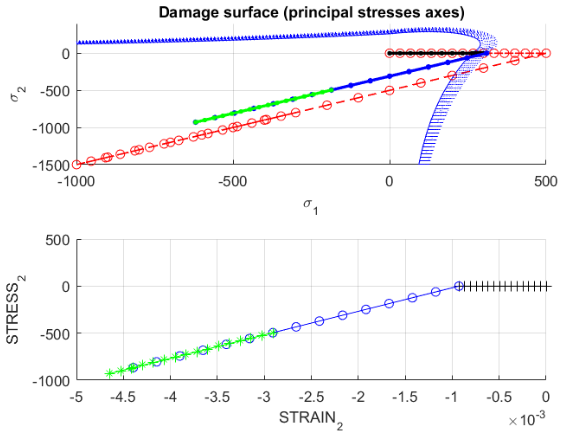
a) Linear Hardening, direction 1



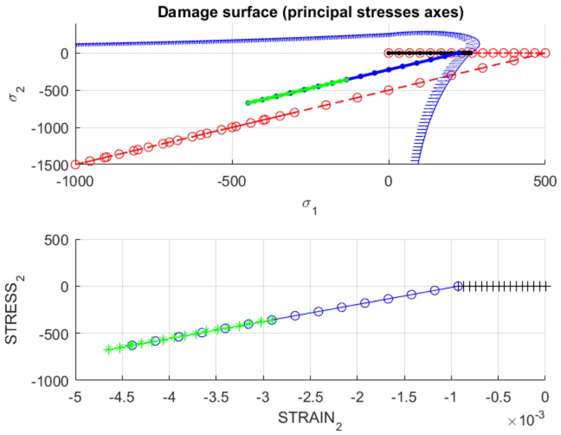
b) Exponential Softening, direction 1

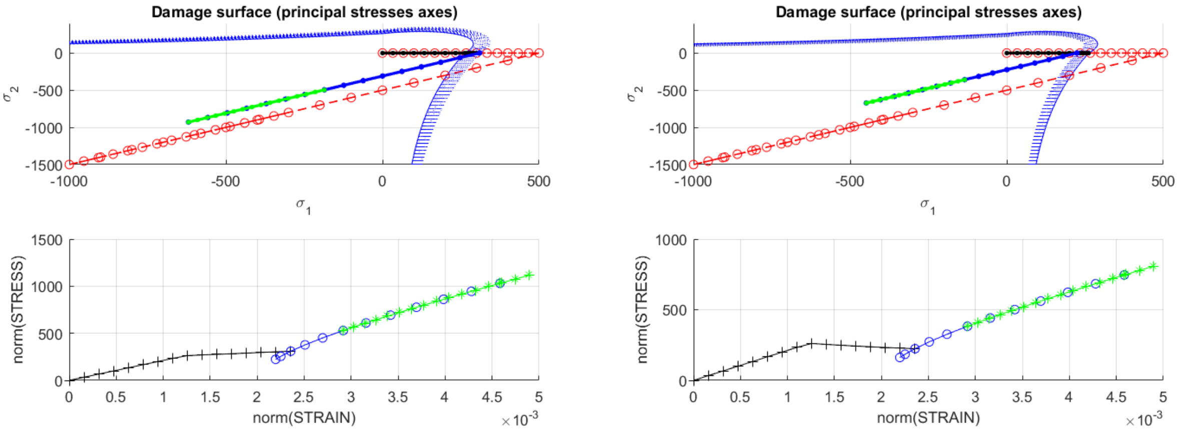


c) Linear Hardening, direction 2



d) Exponential Softening, direction 2

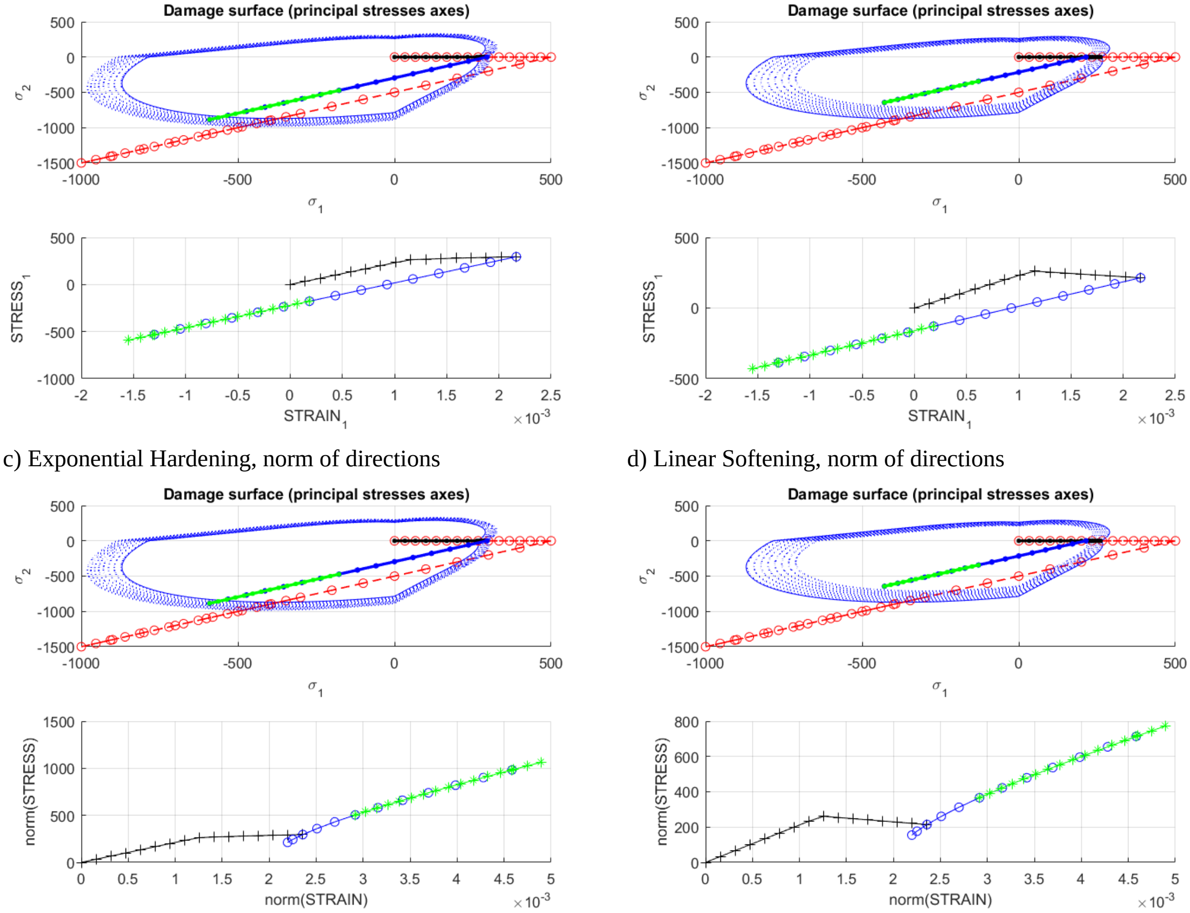




e) Linear Hardening, norm of directions

f) Exponential Softening, norm of directions

Figure 4: Tension-only model, case 2



a) Linear Hardening, direction 1

b) Linear Softening, direction 1

c) Exponential Hardening, norm of directions

d) Linear Softening, norm of directions

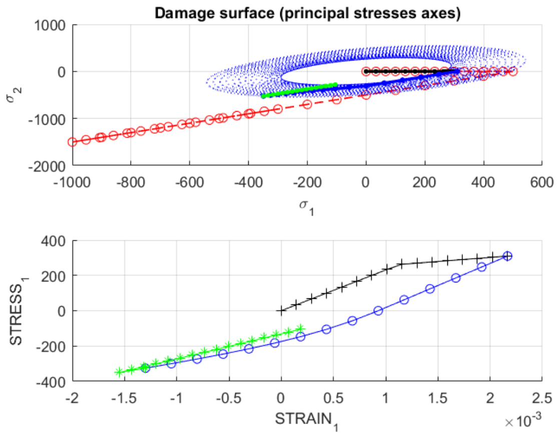
e) Exponential Hardening, norm of directions

f) Exponential Softening, norm of directions

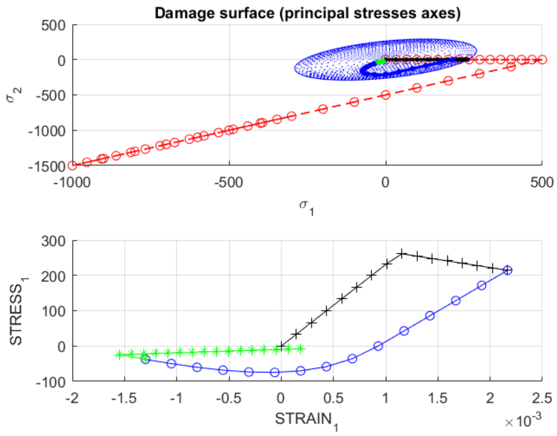
Figure 5: Non-Symmetric model, case 2

The symmetric case is as follows:

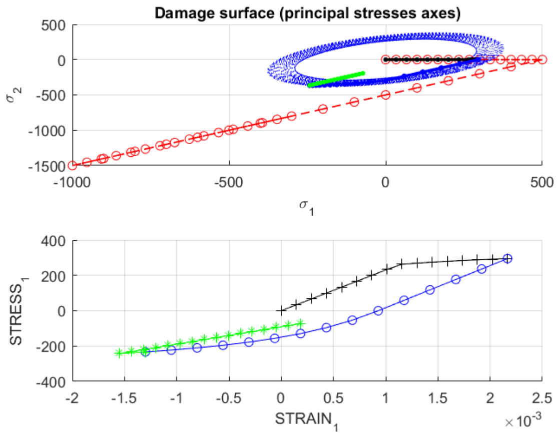
a) Linear Hardening, direction 1



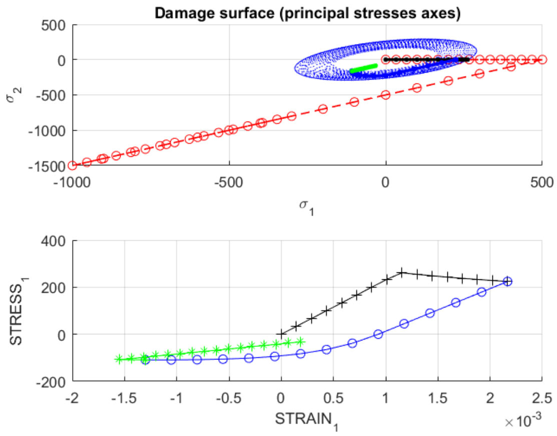
b) Linear Softening, direction 1



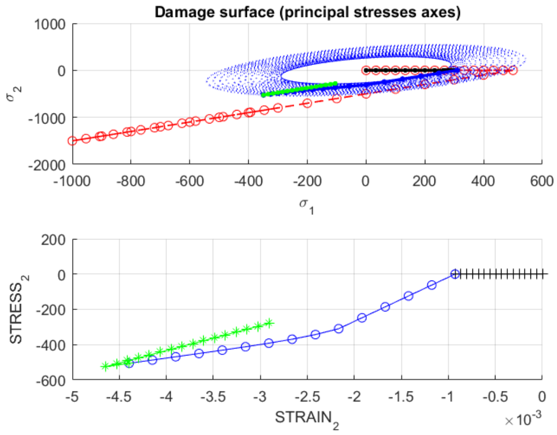
c) Exponential Hardening, direction 1



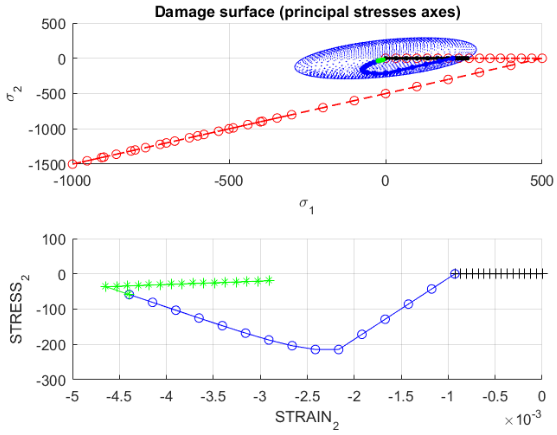
d) Exponential Softening, direction 1

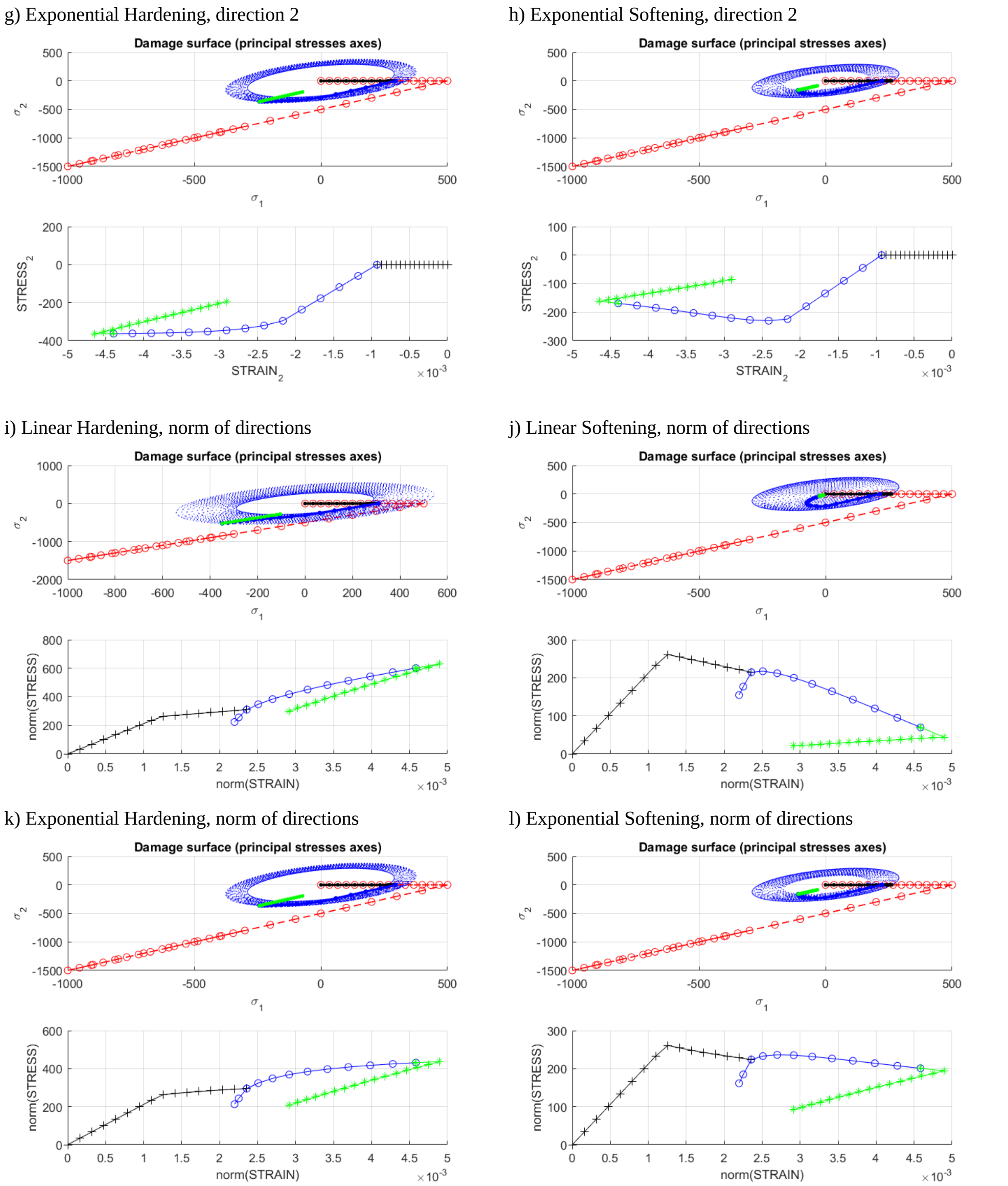


e) Linear Hardening, direction 2



f) Linear Softening, direction 2





g) Exponential Hardening, direction 2

h) Exponential Softening, direction 2

i) Linear Hardening, norm of directions

j) Linear Softening, norm of directions

k) Exponential Hardening, norm of directions

l) Exponential Softening, norm of directions

Figure 6: Symmetric model, case 2

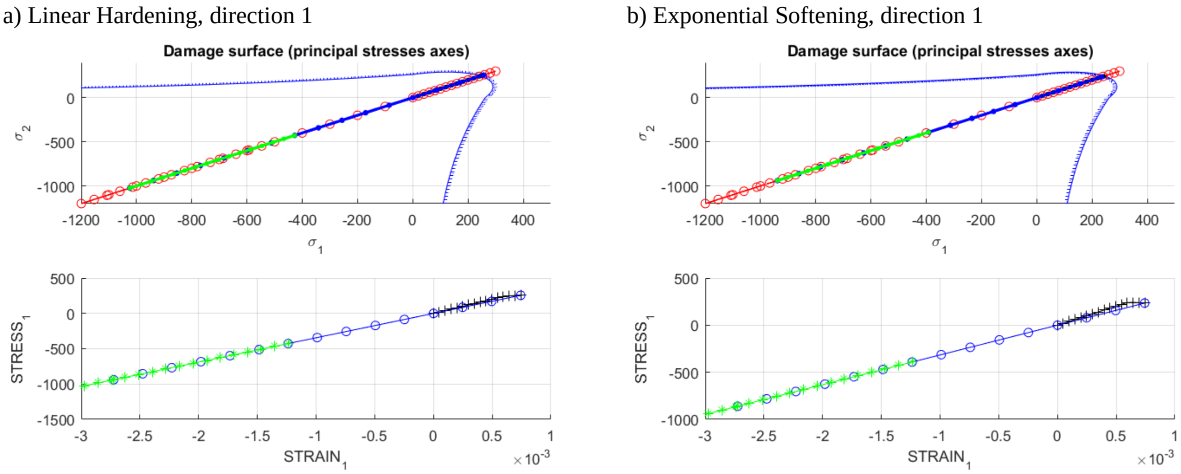
For this case, plotting of the two dimensions is necessary to display all the effects on the material, but since the Non-symmetric and the Tension-only models are practically the same for direction 2, in Non-symmetric this plots are omitted. Also, exponential and linear effects are shown alternated due to the similarities of both cases, except in the symmetric case where differences are noticeable. It can be seen from the images that, when the stress is only applied in dimension 1, there is still some strain in direction 2, which is directly linked to the Poisson ratio, which affects the components of the C tensor, particularly C_{22} and C_{12}. For both Tension-only and Non-symmetric the stress in the unload/compressive load remains in the elastic zone due to the hardening and softening according to the case, so the plots only show the linear effects similar as the ones seen in both case 1 and 3. For this two models, when analyzing the norm of the stress-strain curve it is noticed that there is a sort of a linear increase after the first path, which is the representation of the bigger compressive stresses that are applied, remaining on the elastic domain. The rest of the effects is similar to what has been seen in the first case, with the hardening and softening expanding or contracting the slopes of the path that come after the one analyzed. In this case, however, is worth noticing the effects of the symmetric model. In both hardening and softening, the biaxial unload/compressive load occurs outside the elastic zone, expanding or contracting the damage zone in every step and showing different effects, which is the reason why the stress-strain is curved, for example. In this case, due to the biaxial nature of the stresses, unloading the material does not take the curve to the origin as seen on the previous case, which is explained by the fact that, although unloading the 1 dimension, the 2 dimension is in compression from the second increment, which takes the curve outside of the common path seen before. This contrast both with cases 1 and 3, since in those cases there is pure axial or biaxial stress, different to the mixed behavior shown in this particular case. Norm of the strain-stress curve on the symmetric case show that there is no pattern to find in the curve, since there is an elastoplastic behavior on the first path that transition into a second curved path, which can only be explained due to this dichotomy of stresses applied. This curves, however, display the effect of the exponential/linear laws, which shape the paths: for exponential the inelastic behaviors are curved, while for linear case there is a bilinear effect in the transition of elastic and plastic behavior. Also, for hardening the slope remains positive, while for softening the slope becomes negative at the inelastic zone.

Case 3

Finally, the third case is a biaxial load and unload, defined as

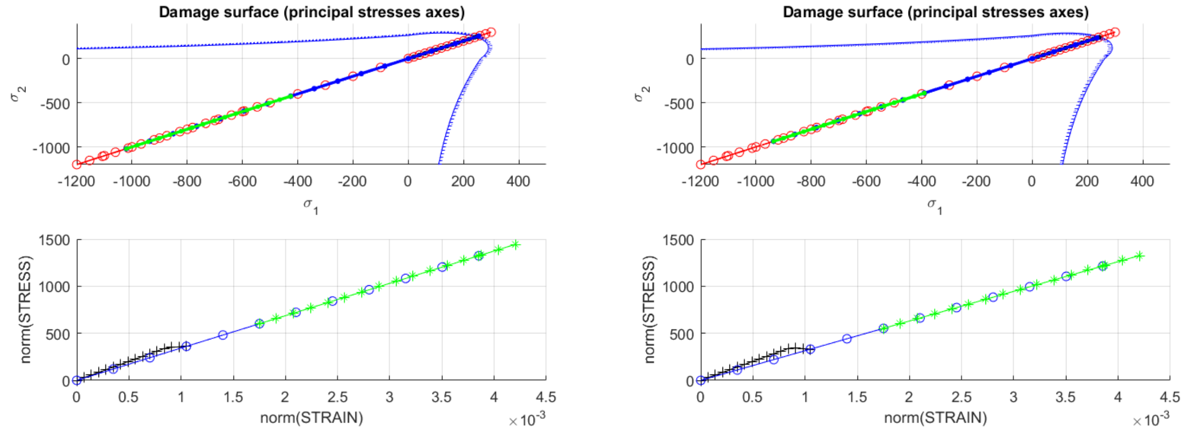
$$\begin{aligned}
&\Delta\bar{\sigma}_1^{(1)} = \alpha; \quad \Delta\bar{\sigma}_2^{(1)} = \alpha \\
&\Delta\bar{\sigma}_1^{(2)} = -\beta; \Delta\bar{\sigma}_2^{(2)} = -\beta \\
&\Delta\bar{\sigma}_1^{(3)} = \gamma; \quad \Delta\bar{\sigma}_2^{(3)} = \gamma
\end{aligned}$$

For this case the values are chosen as α=300 MPa, β=-1500 MPa and γ=700 MPa. Considering 15 steps per increments, the results are shown below:



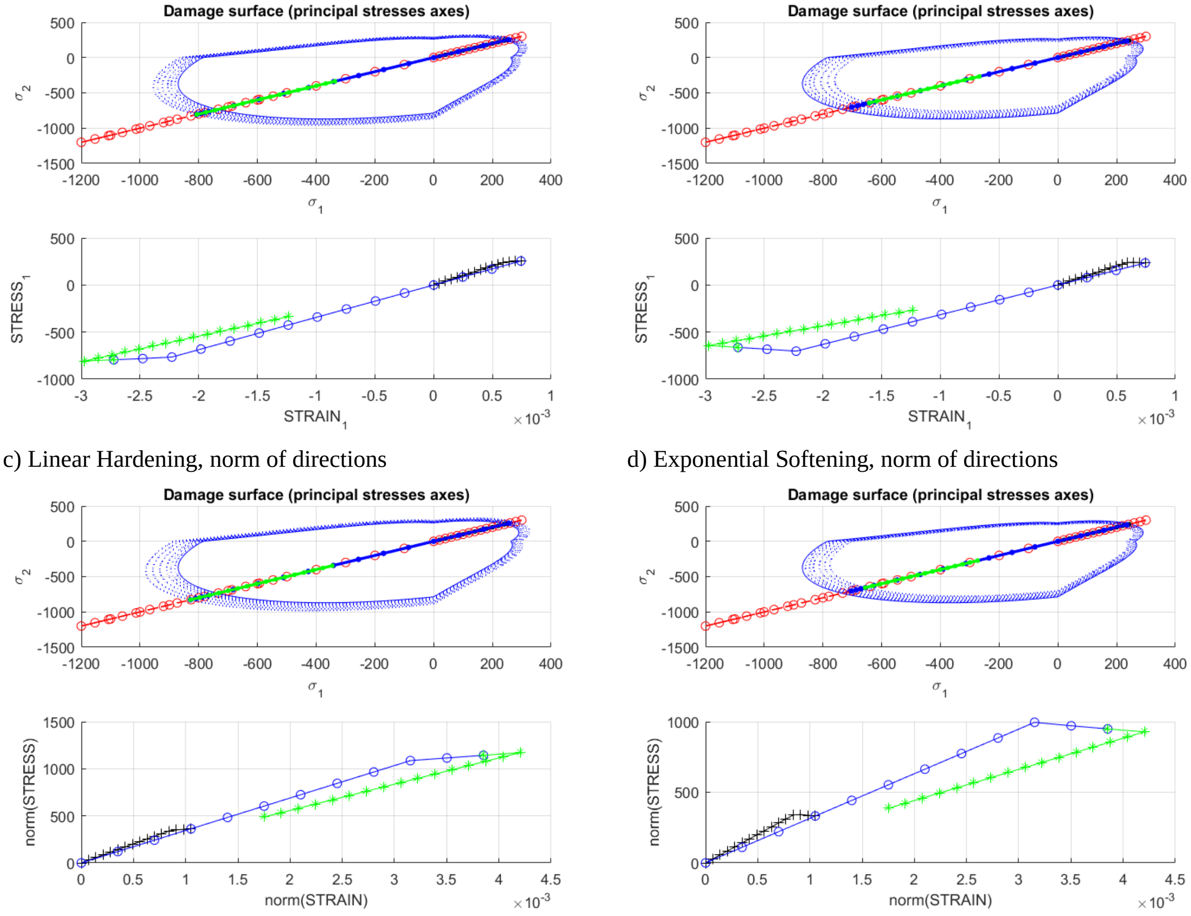
a) Linear Hardening, direction 1

b) Exponential Softening, direction 1


c) Exponential Hardening, norm of directions

f) Linear Softening, norm of directions

Figure 7: Tension-only model, case 3


a) Exponential Hardening, direction 1

b) Linear Softening, direction 1

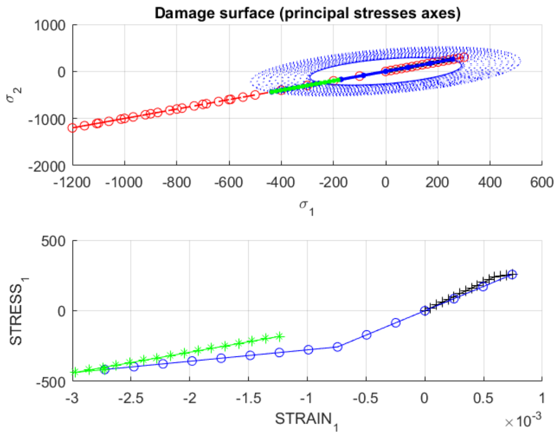
c) Linear Hardening, norm of directions

d) Exponential Softening, norm of directions

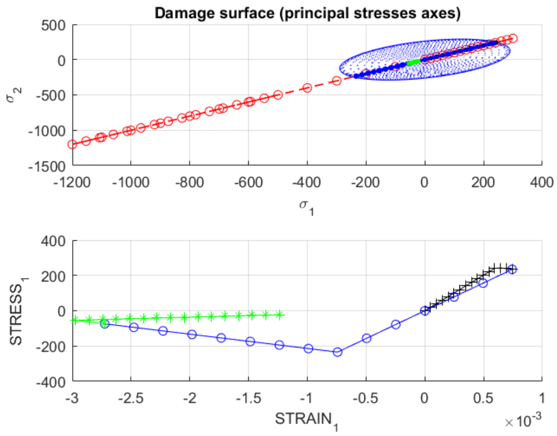
Figure 8: Non-symmetric model, case 3

Comparing against the symmetric case:

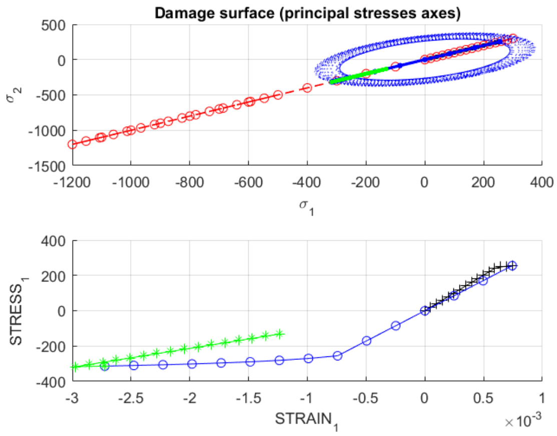
a) Linear Hardening, direction 1



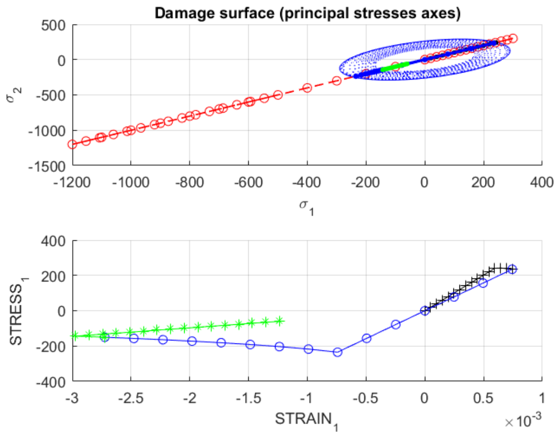
b) Linear Softening, direction 1



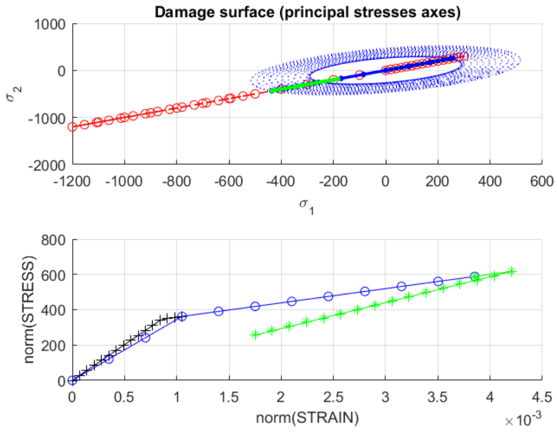
c) Exponential Hardening, direction 1



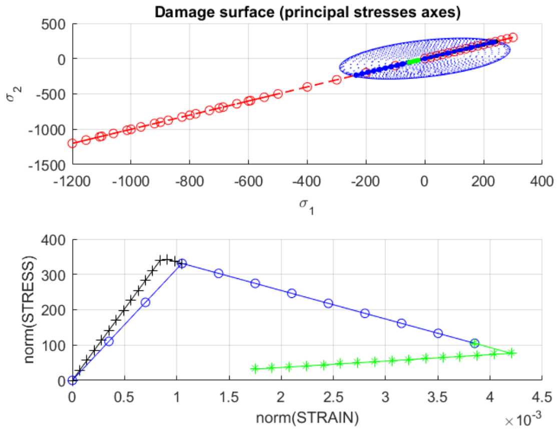
d) Exponential Softening, direction 1



e) Linear Hardening, norm of directions



f) Linear Softening, norm of directions



g) Exponential Hardening, norm of directions

h) Exponential Softening, norm of directions

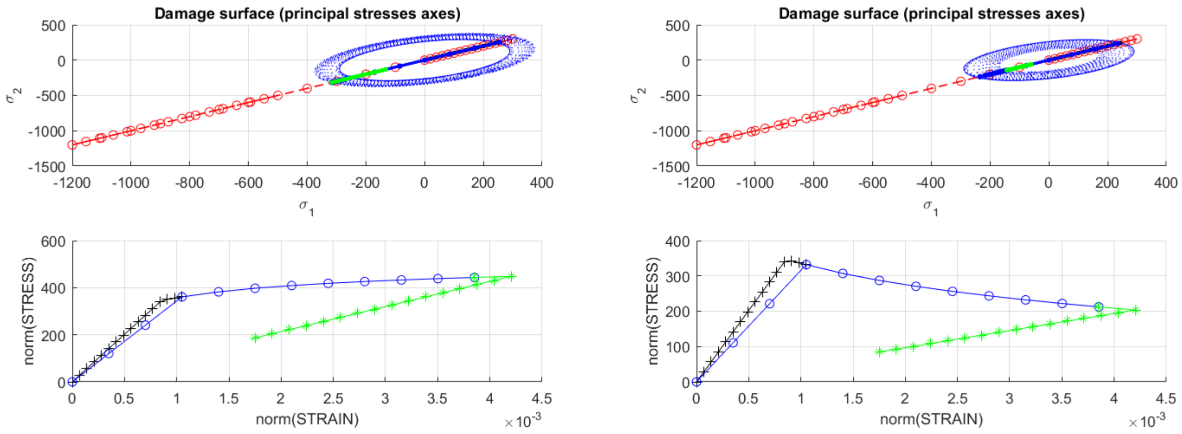
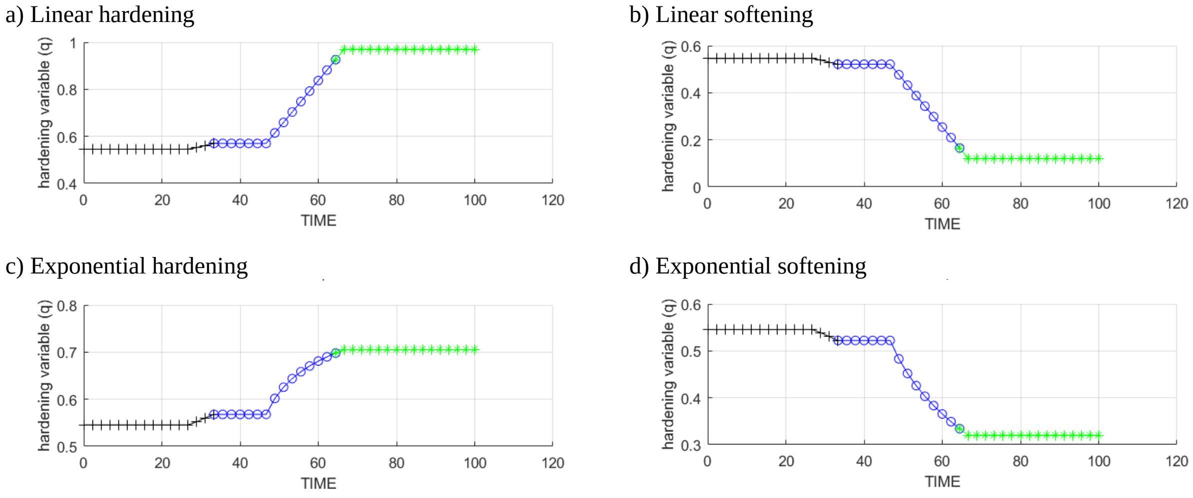


Figure 9: Symmetric model, case 3

It can be seen that this case looks very similar to the first one, since we are considering practically the same, only in both dimensions. Since the curves lie in the xy line, stress-strain curves in dimension 1 and 2 exactly the same, so one of them is omitted from the plots. The tension-only exhibit the same behavior as case 1, with the first path having an inelastic behavior and the rest lying on the same slope. Non-symmetric case it's more interesting since both load cases (tensile and compressive) delve into the inelastic zone, creating a bilinear effect on both sides. When displaying the norm of stress-strain curves the different load slopes are noticeable. For Non-symmetric case we see that the black path (tension) has a limit of around 400 MPa, while the blue curve (discharge/compression) rises up to 1200 MPa, showing the effects of the election of the n value. This can be contrasted in the symmetric case, where it is noticeable that the inelastic slope is always the same, so the even for different elastic loads, the inelastic load behaves in the same way, with the elastic discharge again going to the origin of the strain-stress plot. In the symmetric case also it is noticeable the differences between the exponential and linear laws, since the inelastic part displays the same behavior as the hardening variable shown in the next section. In the symmetric model it is also seen the effect of this different laws in the slopes of inelastic loads, since linear tends to give bigger values that exponential ones.

Hardening variable

To show that exponential and linear Hardening/Softening laws are working and to display their differences, case 3 is plotted in the symmetric model for the same four case studied before:



a) Linear hardening

b) Linear softening

c) Exponential hardening

d) Exponential softening

Figure 10: hardening variable (q) for hardening and softening.

Here the effects of both laws can be explained better, since it is seen how the variation occurs in the hardening variable. In the exponential case the curve tends to a value in a curved and slower way than in linear, which means that at the moment it reaches path 3, that in this case is elastic, the value obtained for q is closer to the original q than the linear one, which means that the curves of the damage zone in the stress domain plot are smaller. This can difference can also be noted in the shape of the damage curves in previous figures. For linear this are equally spaced because the rate of increase is the same, while for exponential one the curves are more separate at the beginning and then more closer for bigger values of stress, which shows the convergence to the asymptotic value of an exponential curve.

Visco-damage model

For analyzing the visco-damage model the parameters η, $\dot{\varepsilon}$, and α are studied. The methodology to show how this parameters affect the model is as follows: for any given α, different values of η and $\dot{\varepsilon}$ are plotted together in the symmetric model case, to show the dependency between the integration constant and the viscosity and strain rate parameters. The control value will be for $\alpha = 1$ and $\eta = 0$, which correspond to the inviscid case treated in the previous section. To show how the integration scheme affects the values, plots are drawn with a constant η and varying values of α with a constant strain rate. Material values are kept the same as the previous case, with the difference that now there is only one path instead of three, since there's no interest in varying stresses as the focus is kept on the influence of the viscous parameters.

The case is analyzed with a linear hardening law, considering the constant parameters:

$$E = 210\ GPa$$
$$\nu = 0.3$$
$$H = 0.2$$
$$\sigma_y = 150\ MPa$$

Viscosity effect

The results for varying viscosities at different integration schemes are shown in the figures below. For this case, it was a considered a time step $\Delta t = 0.5263\ s$.

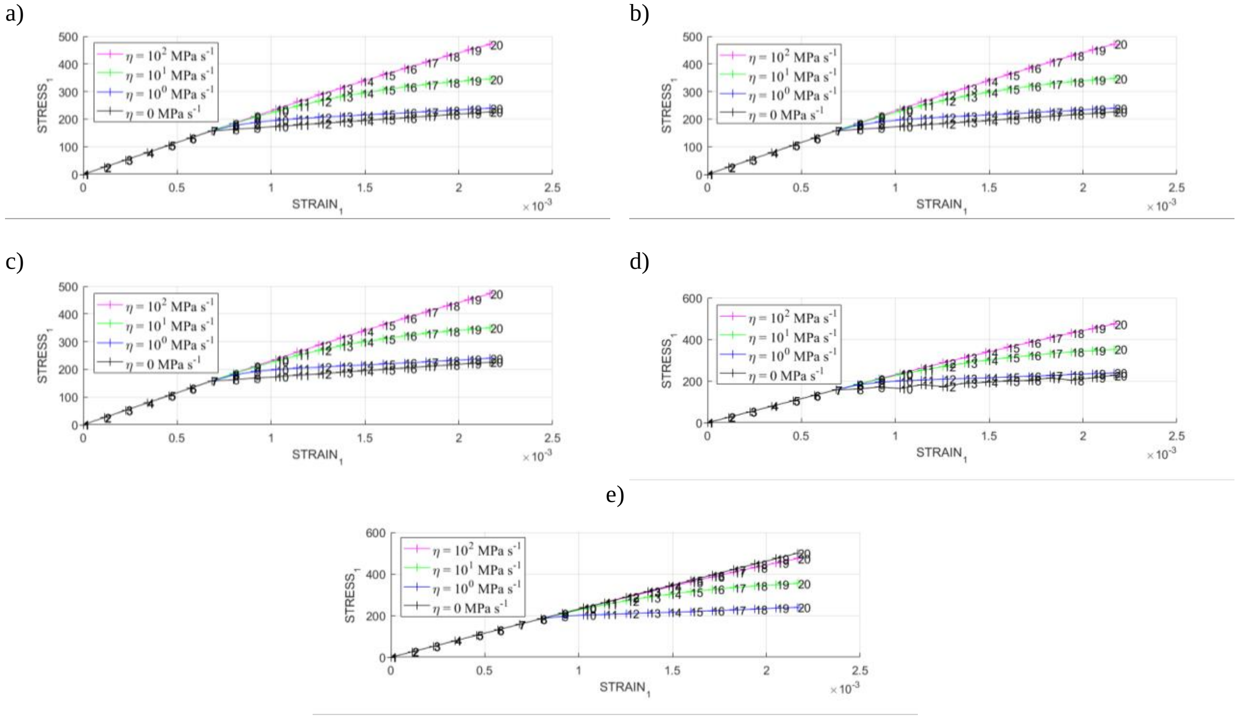


Figure 11: varying viscosities for different α values: a) α=1.00; b) α=0.75; c) α=0.50; d) α=0.25; e) α=0.00.

The effect of viscosities proves that the model is working as it should, since it recreates what theory says about it. For every integration scheme we have the same tendency: higher viscosities account for higher stresses achieved for the same values of strain. It can be seen from the images that when viscosity is high (tending to infinity) the stress strain curve tends to a linear curve, which is the limit to what a material can tend, and this is explained because bigger values of viscosities help to reduce the value of the hardening variable, while maintaining the same value for the strain and stress, thus helping exceed the limitations of the damage curve in the non-viscous model.

Strain rate effects

The analysis was also done on stress rates, where the viscosity was kept fixed at η=10 MPa s^{-1}. The results are shown in the figures below.

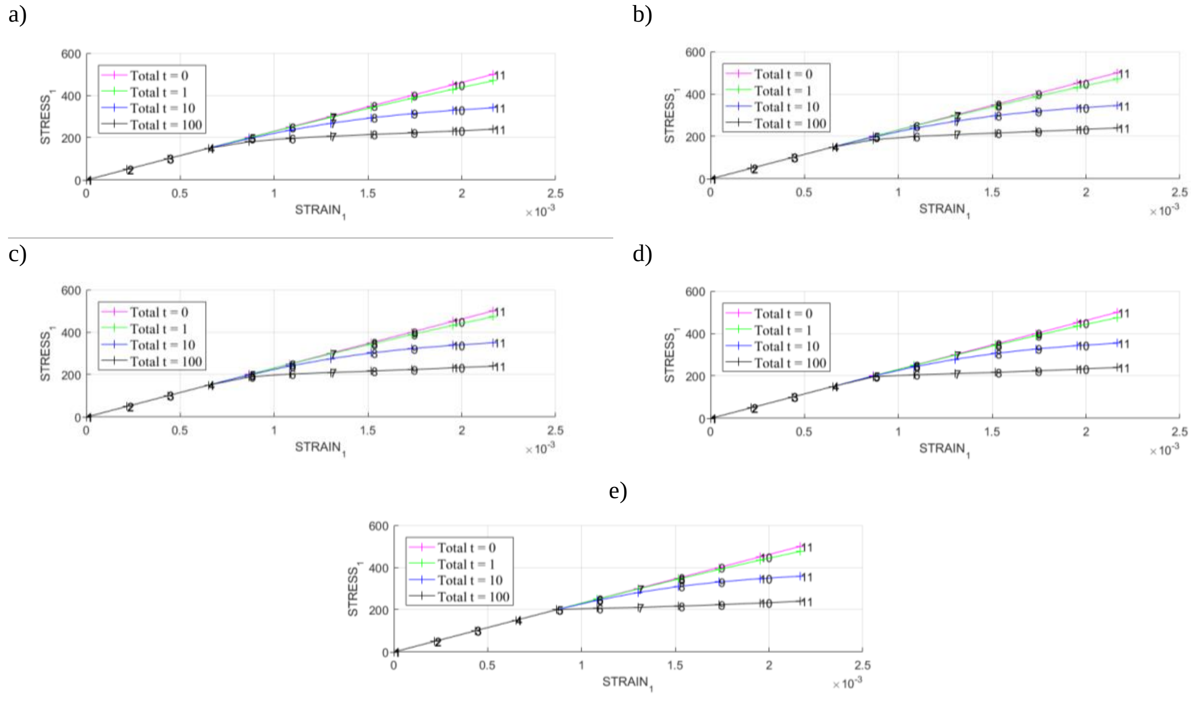


Figure 12: varying strain rates for different α values: a) α=1.00; b) α=0.75; c) α=0.50; d) α=0.25; e) α=0.00.

The effects of strain rate also behave as expected. As can be seen from the images, for smaller total time (which means bigger strain rates) the curves go up, equating the effects of viscosities seen in the previous section. The case for total t = 0, which corresponds to a strain rate tending to infinity, displays the same linear effect seen for big viscosities, which is the maximum value at which the material can tend.

Integration scheme effects

To see how the integration scheme affects the results, varying α values were plotted against different viscosities at a fixed strain rate (Δt=0.5263 s), and the results obtained were the following:

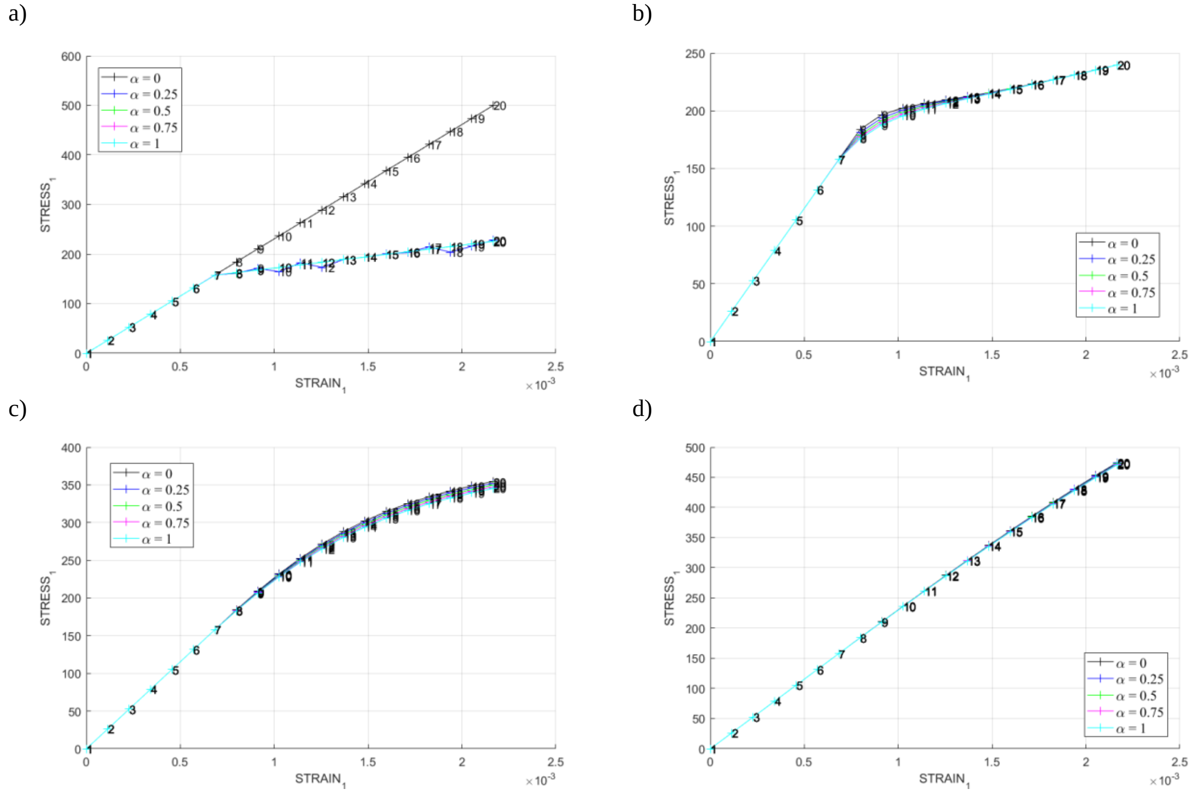


Figure 13: varying α values for different viscosities: a) η=0 MPa s^{-1}; b) η=10^0 MPa s^{-1}; c) η=10^1 MPa s^{-1}; d) η=10^2 MPa s^{-1}.

The first thing that can be noticed is that when there is no viscosity in an explicit scheme (α=0) is that the system does not converge, so we actually get no solution, and this is expressed as an completely elastic material, so this case should be neglected from the analysis. Analyzing the rest of the cases it can be deduced that the integration scheme does not have too much effect in the result, but this little effect is related to the viscosity of the system. For bigger viscosities the effect is negligible, while in null viscosity is noticeable the instability of the implicit scheme as well has the low α schemes. For intermediate values of viscosities there is some spreading of the values: for small values of viscosity the integration scheme affects the change in slopes, while for a bigger viscosity the result varies on the maximum stress value. In any case, there isn't any noticeable pattern that can be extracted from the results, so the conclusion that is extracted is that, except from very particular set of values or conditions, as well as some instabilities of the explicit method, there is no effect from integrations schemes in the strain-stress relation. Instabilities can be tackled modifying the timestep, since explicit methods are really dependent of this parameter. For every method with an α bigger or equal than 0.5 is unconditionally stable, as this is known fact in PDE solving methods.

C_{11} dependance on α

Lastly, a plot of the 11 values of the tangential and algorithmic constitutive matrix against time is done to show how this is affected by the integration scheme. Constant value of η=10 MPa s^{-1} and Δt=0.5263 s are considered for the analysis, which are shown in the figures below:

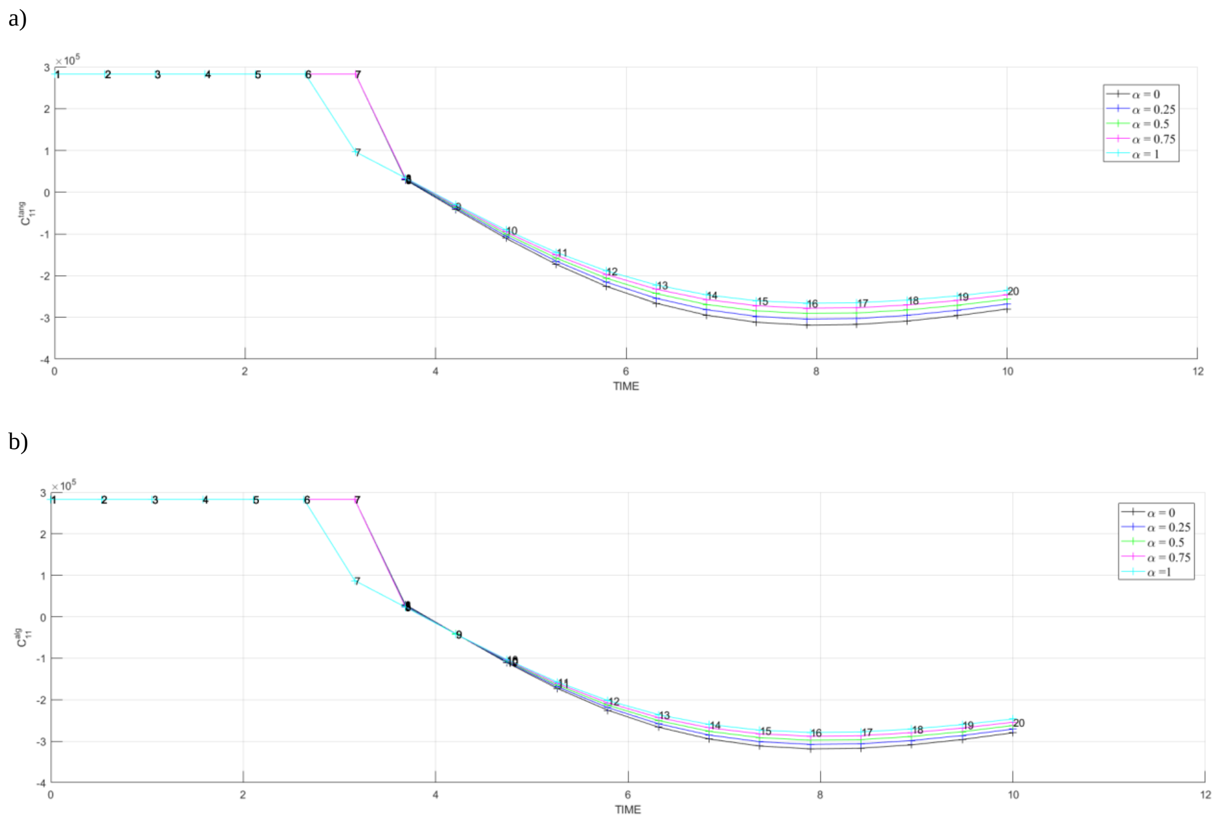


Figure 14: varying α values for: a) C_{11}^{tang}; b) C_{11}^{alg}.

It can be seen that α does have an inherence on the values of C_{11}, clearly noticeably on the spread of values near the end of the path, which at the same time correlates with figure 13.c, since is the same case analyzed. For other viscosities results should correlate the other images from figure 13. As can be seen, the implicit method in both tensors starts the inelastic region before the others, which is related to the fact that an implicit scheme neglects completely the before step, as opposed to the others that in a certain percentage account for the step that has passed. The difference in values is big, due to the units that the C tensor occupies, but translated to stresses the differences are around 10 MPa, which is small for the range of values considered in the material, so this differences can be relatively neglected.

Conclusion

In this report is covered the results obtained from the implementation of the code provided by professor Hermández. It was shown, proving all its capabilities, that the code is working as it should through a series of computation of different cases, contrasting it against: 1) the symmetric case already implemented and working; and 2) the theory, which has been reviewed in class. It remains for future improvements to adapt the code to the plane stress, as well as to the 3D theory, which can be expected for future assignments.

Annex

Modified routines:

main_nointeractive.m

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

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

% YOUNG's MODULUS
% ---------------
YOUNG_M = 210000;
% Poisson's coefficient
% -----------------------
POISSON = 0.3;
% Hardening/softening modulus
% ---------------------------
HARDSOFT_MOD = 0.2;
% Yield stress
% ------------
YIELD_STRESS = 150;
% Problem type  TP = {'PLANE STRESS','PLANE STRAIN','3D'}
% ------------------------ = 1            =2         =3
% ------------
ntype = 2;
% Model    PTC = {'SYMMETRIC','TENSION','NON-SYMMETRIC'} ;
%                     = 1         = 2         = 3
% ---------------------------------------------------
MDtype = 3;
% Ratio compression strength / tension strength
% ---------------------------------------------
n = 3;
% SOFTENING/HARDENING TYPE
% ------------------------
HARDTYPE = 'LINEAR' ; %{LINEAR,EXPONENTIAL}
% HARDTYPE = 'EXPONENTIAL' ; %{LINEAR,EXPONENTIAL}
% EXPONENTIAL HARDENING CONSTANT
% ------------------------
% VISCOUS = 'NO';
VISCOUS = 'YES';
% Viscous coefficient ----
% ------------------------
eta = 10;
% TimeTotal (initial = 0) ----
% ------------------------
TimeTotal = 10;
% Integration coefficient ALPHA
% ------------------------
ALPHA_COEFF = 0.5;
% Points ---------------------------
% ----------------------------------
nloadstates = 1;
SIGMAP = zeros(nloadstates,2);
SIGMAP(1,:) =[ 500 0];
% SIGMAP(2,:) =[ 200 -200];
% SIGMAP(3,:) =[ 500 100];

% Number of time increments for each load state
% ---------------------------------------
istep = 19*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)','C^{tang}_{11}','C^{alg}_{11}'};

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

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

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

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

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

% DAMAGE MODEL
% ------------
[sigma_v,vartoplot,LABELPLOT_out,TIMEVECTOR]=damage_main(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 = zeros(length(sigma_v)-1,1);

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(i-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) ;
```

damage_main.m

```
function [sigma_v,vartoplot,LABELPLOT,TIMEVECTOR]=damage_main(Eprop,ntype,istep,strain,MDtype,n,TimeTotal)
global hplotSURF

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% CONTINUUM DAMAGE MODEL
% ----------------------
% Given the almansi strain evolution ("strain(totalstep,mstrain)") and a set of
% parameters and properties, it returns the evolution of the cauchy stress and other  variables
% that are listed below.
%
% INPUTS <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
% ----------------------------------------------------------------
% Eprop(1) = Young's modulus  (E)
% Eprop(2) = Poisson's coefficient (nu)
% Eprop(3) = Hardening(+)/Softening(-) modulus (H)
% Eprop(4) = Yield stress (sigma_y)
% Eprop(5) = Type of Hardening/Softening law  (hard_type)
%            0 --> LINEAR
%            1 --> Exponential
% Eprop(6) = Rate behavior (viscpr)
%            0 --> Rate-independent (inviscid)
%            1 --> Rate-dependent   (viscous)
%
% Eprop(7) = Viscosity coefficient (eta)  (dummy if inviscid)
% Eprop(8) = ALPHA coefficient (for time integration), (ALPHA)
%             0<=ALPHA<=1 , ALPHA = 1.0 --> Implicit
%                           ALPHA = 0.0 --> Explicit
%            (dummy if inviscid)
% Eprop(9) = Value of q at infinity
% Eprop(10) = Exponential constant for hardening/softening
%
% ntype    = PROBLEM TYPE
%            1 : plane stress
%            2 : plane strain
%            3 : 3D
%
% istep = steps for each load state (istep1,istep2,istep3)
%
% strain(i,j) = j-th component of the linearized strain vector at the i-th
%               step, i = 1:totalstep+1
%
% MDtype      = Damage surface criterion %
%            1 : SYMMETRIC
%            2 : ONLY-TENSION
%            3 : NON-SYMMETRIC
%
%
% n          = Ratio compression/tension strength (dummy if MDtype is different from 3)
%
% TimeTotal  = Interval length
%
%  OUTPUTS <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
%  ------------------------------------------------------------------
%  1) sigma_v{itime}(icomp,jcomp)  --> Component (icomp,jcomp) of the cauchy
%                                    stress tensor at step "itime"
%                                    REMARK: sigma_v is a type of
%                                    variable called "cell array".
%
%
%  2) vartoplot{itime}              --> Cell array containing variables one wishes to plot
%                                     --------------------------------------
%   vartoplot{itime}(1) =   Hardening variable (q)
%   vartoplot{itime}(2) =   Internal variable (r)
%
%
%  3) LABELPLOT{ivar}              --> Cell array with the label string for
%                                     variables of "varplot"
%
%          LABELPLOT{1} => 'hardening variable (q)'
%          LABELPLOT{2} => 'internal variable'
%
%
%  4) TIME VECTOR  - >
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% SET LABEL OF "vartoplot" variables  (it may be defined also outside this function)
% ----------------------------------

LABELPLOT = {'hardening variable (q)','internal variable'};

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

if ntype == 1
    menu('PLANE STRESS has not been implemented yet','STOP');
    error('OPTION NOT AVAILABLE')
elseif ntype == 3
    menu('3-DIMENSIONAL PROBLEM has not been implemented yet','STOP');
    error('OPTION NOT AVAILABLE')
else
    % mstrain = 4;
      mhist   = 6;
end

if viscpr == 1
    % Extend properties vector to pass Dt value
    Eprop = [Eprop  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_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  = cell(mhist) ;

% INITIALIZING  (i = 1)
% *************
i  = 1;
r0 = sigma_u/sqrt(E);
hvar_n{3} = ce;         % hvar structure and type modified to include C11 in analysis
hvar_n{4} = ce;
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)
vartoplot{i}(4) = hvar_n{3}(1,1);        % C_tang_11
vartoplot{i}(5) = hvar_n{4}(1,1);        % C_alg_11

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"
        % ------------------------
        Eprop(end) = TIMEVECTOR(i) - TIMEVECTOR(i-1);
        eps = strain(i-1,:) ;
        eps_n1 = strain(i,:);

        %***************************************************************************************
        %*      DAMAGE MODEL
        %***************************************************************************************
        [sigma_n1,hvar_n,aux_var] = rmap_dano1(eps,eps_n1,hvar_n,Eprop,ce,MDtype,n);

        % 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)
        vartoplot{i}(4) = hvar_n{3}(1,1);           % C_tang_11
        vartoplot{i}(5) = hvar_n{4}(1,1);           % C_alg_11
    end
end
```

rmap_dano1.m

```
function [sigma_n1,hvar_n1,aux_var] = rmap_dano1 (eps,eps_n1,hvar_n,Eprop,ce,MDtype,n)
% added eps to inputs to compute difference in time
%********************************************************************************************
%*                                                                                          *
%*           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);
H         = Eprop(3);
sigma_u   = Eprop(4);
hard_type = Eprop(5);
Aexp      = Eprop(3);

visc      = Eprop(6);
alf       = Eprop(8);
eta       = Eprop(7);
%**************************************************************************************

%**************************************************************************************
% Initializing

r0 = sigma_u/sqrt(E);
zero_q=1.d-6*r0;
q_inf=r0+(r0-zero_q);
%**************************************************************************************

%**************************************************************************************
% Damage surface

tau_bf = Modelos_de_dano1 (MDtype,ce,eps,n);
tau_af = Modelos_de_dano1 (MDtype,ce,eps_n1,n);

if visc == 0
    rtrial = tau_af;
else
    tau = (1-alf)*tau_bf+alf*tau_af;
    dt = Eprop(9);
    if tau <= r_n
        rtrial = r_n;
    else
        rtrial = r_n*(eta-dt*(1-alf))/(eta+alf*dt)+tau*(dt)/(eta+alf*dt)
    end
end
%**************************************************************************************

%**************************************************************************************
%*   Ver el Estado de Carga
%*   --------->    fload=0 : elastic unload
%*   --------->    fload=1 : damage (compute algorithmic constitutive tensor)
```

```
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;
        H_n1 = H;
    else
        % Exponential
        q_n1 = q_inf-(q_inf-q_n)*exp(Aexp*(1-rtrial/r_n));
        H_n1 = H*(q_inf-q_n)/r_n*exp(Aexp*(1-rtrial/r_n));
    end

    if(q_n1<zero_q)
        q_n1=zero_q;
        H_n1 = 0;
    elseif(q_n1>q_inf)
        q_n1=q_inf;
        H_n1 = 0;
    end

    ce_tang_n1 = (q_n1/r_n1)*ce-(q_n1-H_n1*r_n1)/(r_n1.^3)*((ce*eps_n1')*(ce*eps_n1')');
    if (visc == 1)
        ce_alg_n1  = ce_tang_n1+(alf*dt)/(eta+alf*dt)*(H_n1*r_n1-
                     q_n1)/(tau_af*r_n1.^2)*((ce*eps_n1')*(ce*eps_n1')');
    end

else
    % Elastic load/unload

    fload=0;
    r_n1= r_n  ;
    q_n1= q_n  ;
    ce_alg_n1  = (q_n1/r_n1)*ce;
    ce_tang_n1 = ce_alg_n1;
end

% Damage variable
% ---------------
dano_n1  = 1.d0-(q_n1/r_n1);

% Computing stress
% ----------------
sigma_n1 = (1.d0-dano_n1)*ce*eps_n1';
%*************************************************************************************

%*************************************************************************************
% Updating historic variables
hvar_n1{3} = ce_tang_n1;          % Modified the structure of hvar to include C11
hvar_n1{5} = r_n1;
hvar_n1{6} = q_n1;
if (visc == 1)
    hvar_n1{4} = ce_alg_n1;
end
%*************************************************************************************

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

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
sig_n1 = eps_n1*ce;
sig_n1(sig_n1 < 0) = 0;
rtrial= sqrt(sig_n1*eps_n1');

elseif (MDtype==3)  %*Non-symmetric
sig_n1 = eps_n1*ce;
sig_n1p = sig_n1;
sig_n1p(sig_n1p < 0) = 0;
th = (sig_n1p(1)+sig_n1p(2))/(abs(sig_n1(1))+abs(sig_n1(2)));
rtrial= (th+(1-th)/n)*sqrt(eps_n1*ce*eps_n1');

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