Universitat Politècnica de Catalunya

Computational Solid Mechanics

Master's Degree in Numerical Methods in Engineering

Assignment on Constitutive Damage Models

Authors:
Pau Márquez

Supervisor:
Prof. J. A. Hernández Ortega

Academic Year 2019-2020

Contents

1 Introduction and Motivation

The constitutive damage models are a very useful and rather simple tool for evaluating the behavior of some geomaterials such as concrete or rocks that present a characteristic propagation of micro-cracks under certain loads. This models are therefore suitable to study the diminishing capacity of these materials to carry stresses when certain thresholds are surpassed.

The procedure to asses the code developed will be the following:

- The characteristics of the analysis will be detailed as well as the values of the parameters used.
- The graphs will be shown in such a way that there is a balance between the amount of information present and the easiness with which information is extracted and correctly visualized.
- Conclusions regarding the similarities of the code response with respect to the theoretical explanations of the phenomena taking place will be made in a clear and concise way inside boxed text.

The motivation of this project is thus to code and asses these constitutive laws that will allow us to analyze the loading capacity of certain materials.

2 Rate independent models

In this section the capabilities of the code regarding the tension-only damage model and the non-symmetric tension-compression damage model will be explained through the use of graphics that will help understanding the behavior of the code under certain parameters.

Before referring to the specific developed models, let us consider a simple case that will allow us to better understand the behavior of the materials being analyzed under certain stress paths. Fig. 1 shows the stress strain curve of a three-uni-axial stress path using the only-tension damage model. From this Figure, important concepts will be introduced that will be later on referred to several times:

- The yield stress (σ_y), which will define the elastic domain, and more specifically, the damage surface.
- The diminished Young's Modulus, defining the loading branch once σ_0 is exceeded.
- The damage variable, d.
- The hardening/softening modulus, H.

It can clearly be seen that, when the material is loaded at a value higher than σ_y, the material diminishes its capacity to carry stresses. Then, even if the material is unloaded (red path) it does not heal and when it is loaded again, it will move along the inelastic path, characterized by a diminished Young's modulus. Eventually, the plastic behavior is reached at a lower σ_y product of the presence of irreversible damage ($d > 0$). When strains continue to be grow, a fully damaged state will be reached in which the material section is occupied by cracks and the stresses are zero no matter how large the strains. In this state, that can be observed in Fig. 1, $d = 1$.

Then, to better emphasize the effect of the H modulus, Fig. 2 shows the plastic behavior of the materials with different linear and exponential H modulus. For clarity, only one loading path has been included. In this project only we will consider the relationship between q and r to be linear or exponential. Damage will be produced once $q_0 = r_0 = \frac{\sigma_y}{\sqrt{E}}$ is exceeded, and this damage will be with hardening, softening, or perfect depending on the value of H. Fig. 2 shows the effects of inserting different values of H and is a good indication that the code gives good results when choosing between linear or exponential hardening/softening law. For most of the report, a value of $H = -0.4$ will be chosen to account for physical feasibility. On addition, the material to be analyzed will be concrete, that suits the models developed here for the reasons explained. In the case of concrete n (the ratio between compression and tension strength) is approximately equal to 10, and this is a value that could be used throughout the report to ensure that the results reflect the behavior of a real geomaterial. However, when testing this material for all the cases, it has been realized that it has a strong non-symmetric response, therefore making it difficult to observe damage for both loading and unloading since the value for compressing damage is much higher. In like manner, a value of 2 has been chosen, that allows a better visualization of the results for each loading path.

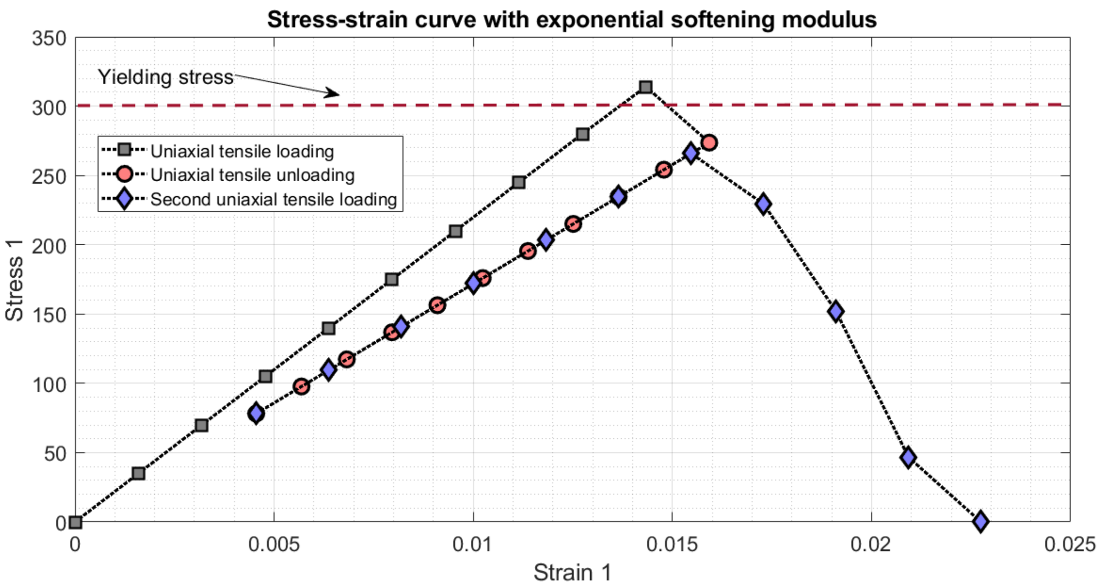

Figure 1: This stress-strain curve show the loss of stress capacity of a material and it serves as an example of the code capacities.

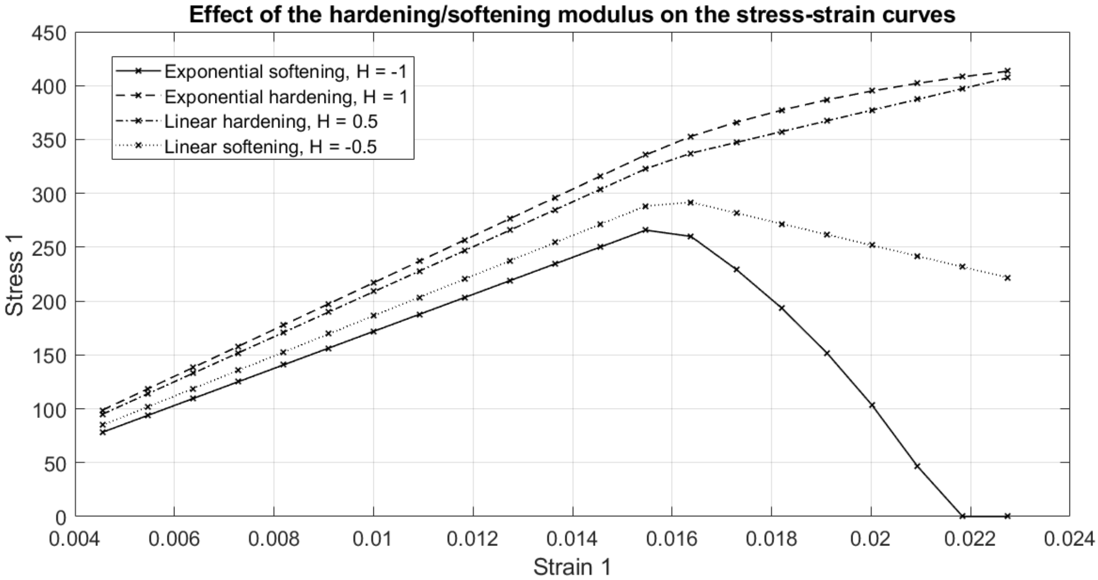

Figure 2: Exponential and linear hardening/softening modulus effect of stress-strain curves.

The values of variables chosen for this section are: $E = 20000$, $n = 2$, $H = -0.4$, $\nu = 0.3$ and $\sigma_y = 300$.

Before analyzing the different developed damage cases, it is important to study a simple symmetric case to later evaluate the differences of the other models with respect to the symmetric. Fig. 3 shows the stress-strain curve for a uni-axial tension/compression path with three effective stress points departing from a value of zero, $\overline{\sigma_1}^1 = 400, \overline{\sigma_1}^2 = -1000, \overline{\sigma_1}^3 = -200$. When observing Fig. 3, it is possible to see that the loading and unloading is done along the elastic branch without damage, but when compressed to a value higher that the yield stress, it is damaged and will diminish its properties. When loaded again, it will perform the same way as when compressed, as it is symmetric.

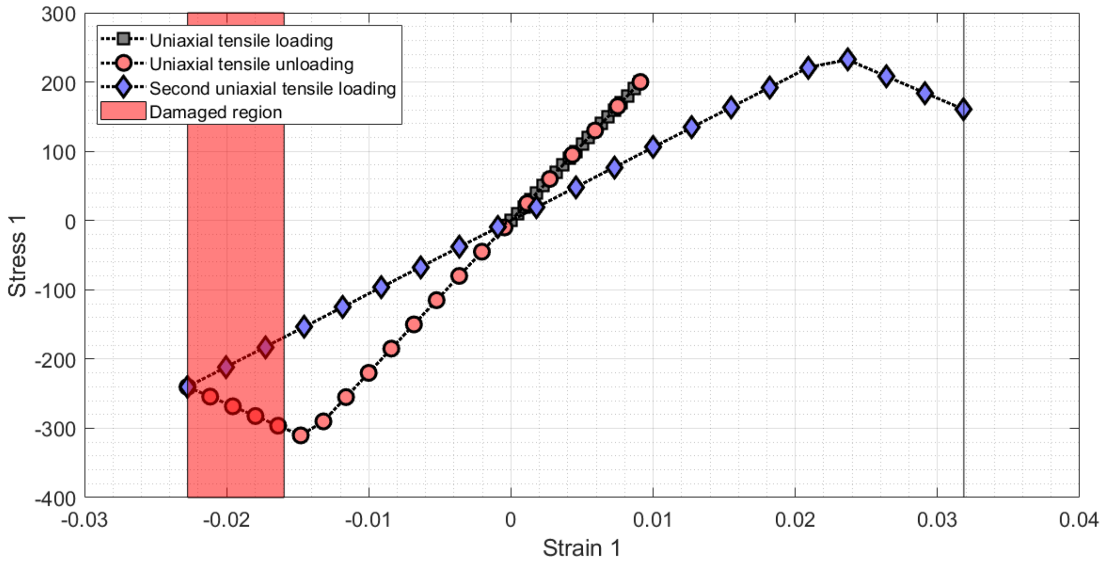

Figure 3: Explanation of the symmetric damage model.

Now, three plane strain cases will be studied for the rate independent tension-only and non-symmetric models.

2.1 Tension-only damage model

The characteristic that will be observed along this subsection is that damage will only be produced for tension loading paths, whereas no effect will be seen on the Young's Modulus of the material when it is compressed.

2.1.1 Uni-axial tensile loading and unloading

The path at the stress space is shown in Fig. 5. Though it is not clear as all the values are in axis σ_1, it is possible to see that no stress points exist beyond the boundary of the damage surface, and all points are inside. As for the stress-strain curve (Fig. 4), is it possible to see the damage generated when the strain norm surpasses the value of r_0, or when $\sigma_1 > \sigma_y$. Then, during the second and third paths, the increments on the tension do not generate further damage and E_d remains constant.

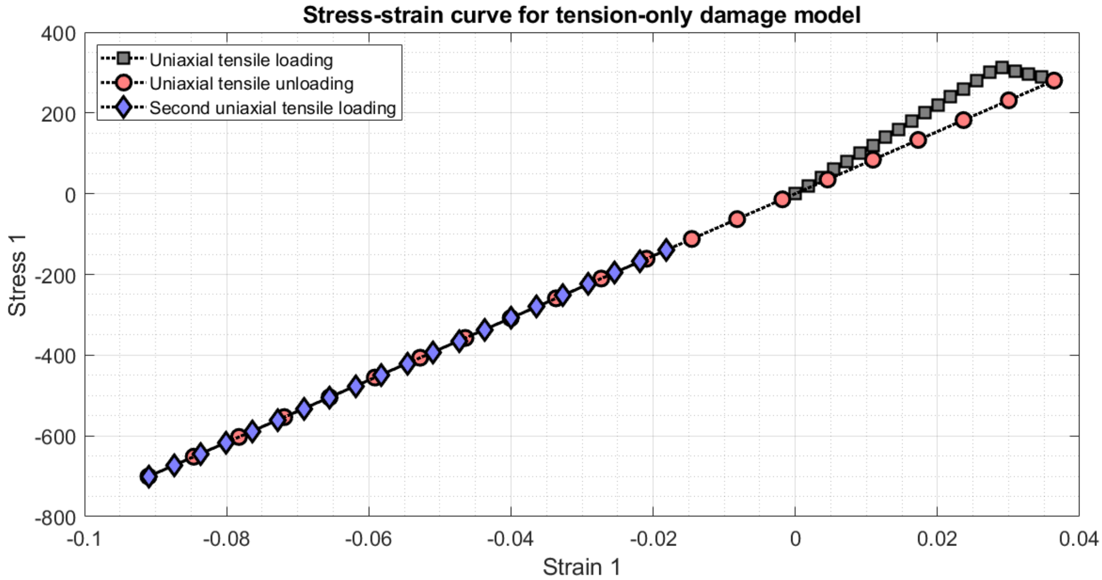

Figure 4: Stress-strain curve for the three proposed loading paths.

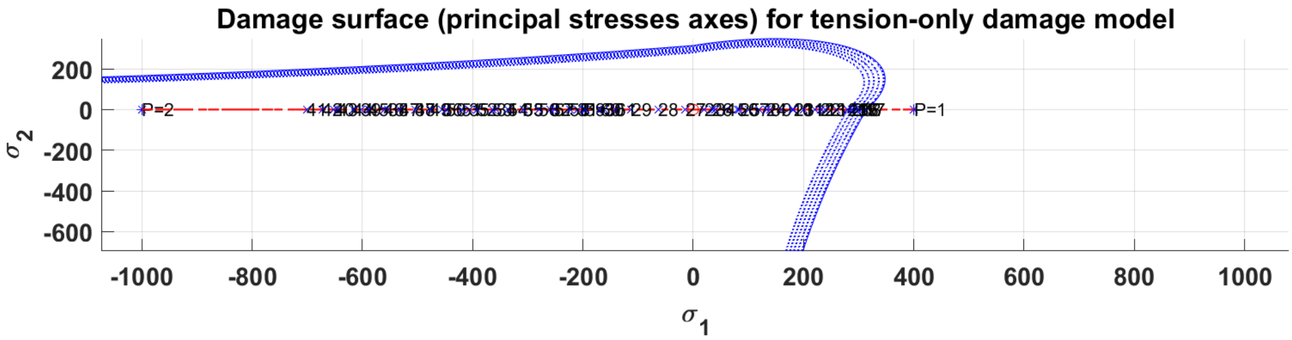

Figure 5: Stress paths and damage surfaces.

2.1.2 Uni-axial tensile loading and bi-axial tensile unloading

In this case here the loading path will be the following, based on the values previously stated. Here, $\alpha = 400, \beta = 1400, \gamma = 800$.

$$(\sigma_1^1, \sigma_2^1) = (400, 0) \longrightarrow (\sigma_1^2, \sigma_2^2) = (-1000, -1400) \longrightarrow (\sigma_1^3, \sigma_2^3) = (-200, -600)) \tag{1}$$

This path can be seen in Fig. 7 and its respective stress-strain graph in Fig. 6. Here the response is not the same as in the first case, since the bi-axial tensile unloading and posterior loading maintain approximately the same Young's Modulus even before damage has been produced (during the first loading path). This is not clealy a general rule but the parameters chosen have led to this configuration. Again the softening parameter is observed by noticing the decreasing phenomena after reaching σ_y.

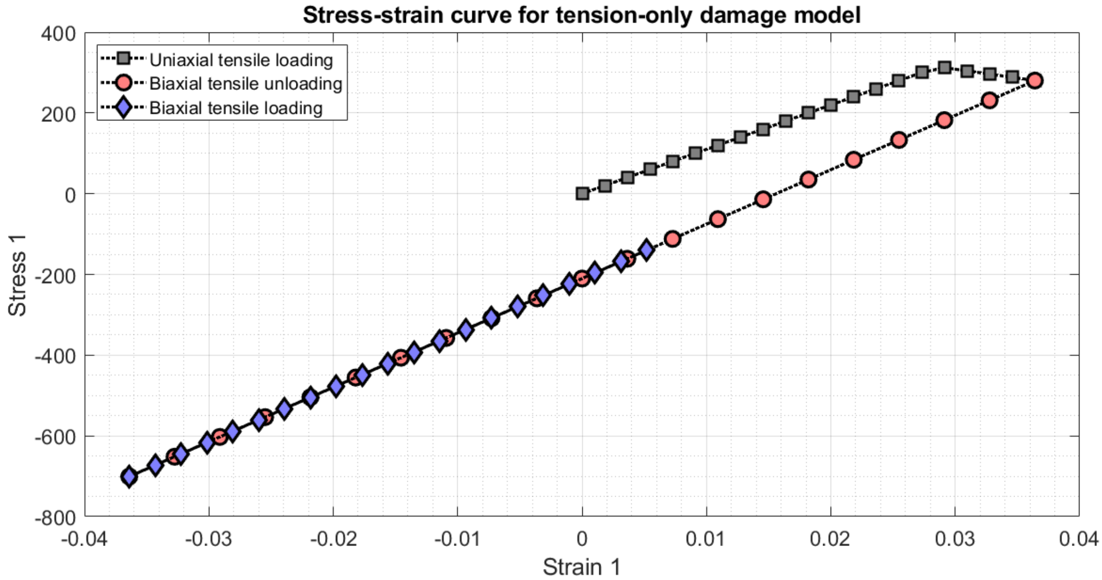

Figure 6: Stress-strain curve for the three proposed loading paths.

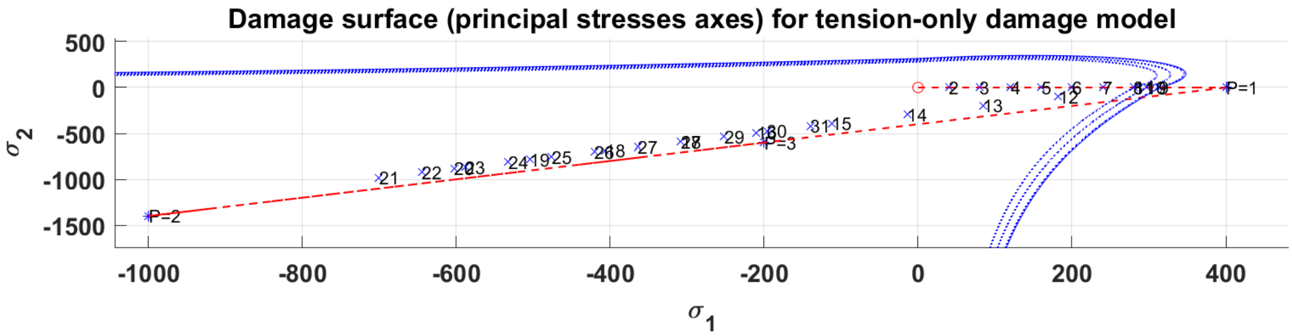

Figure 7: Stress paths and damage surfaces.

2.1.3 Bi-axial tensile loading

According to the proposed increments, the stress paths will now be

$$(\sigma_1^1, \sigma_2^1) = (400, 400) \longrightarrow (\sigma_1^2, \sigma_2^2) = (-1000, -1000) \longrightarrow (\sigma_1^3, \sigma_2^3) = (-200, -200)) \qquad (2)$$

The path is seen in Fig. 9 and the stress-strain curves in Fig. 8. Now the response has a very similar shape to that of the full uni-axial but the values are different. Now, for the same strain to be reached, higher stresses have to be applied to the principal direction. This means that E_d will be higher in this case. This conclusion is assessed by analyzing the paths of both examples. A rotation has been produced of the stress path, which before was a horizontal line and now it is inclined due to $\sigma_2 \neq 0$.

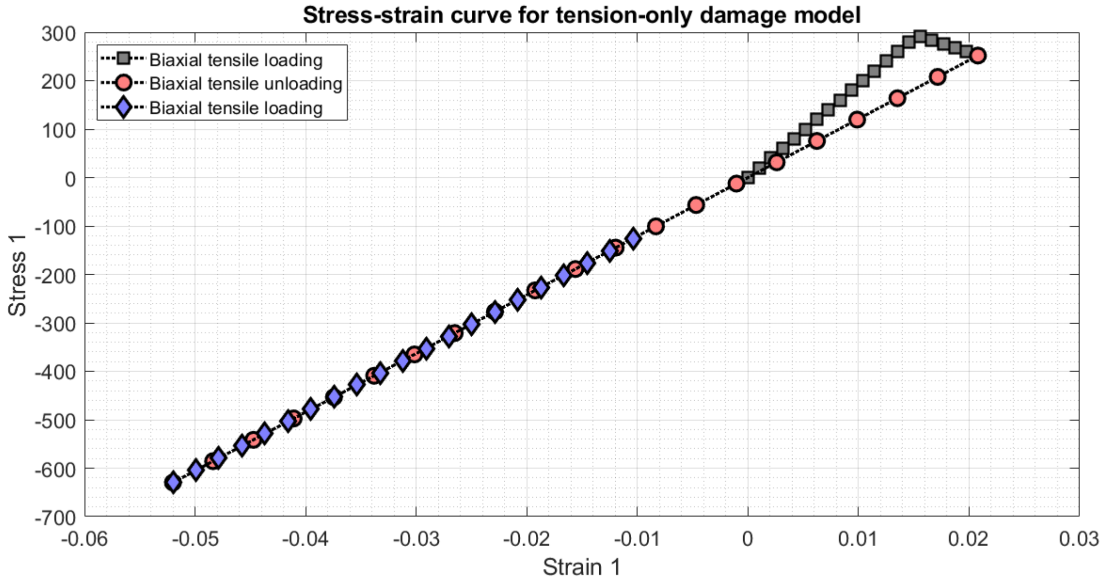

Figure 8: Stress-strain curve for the three proposed loading paths.

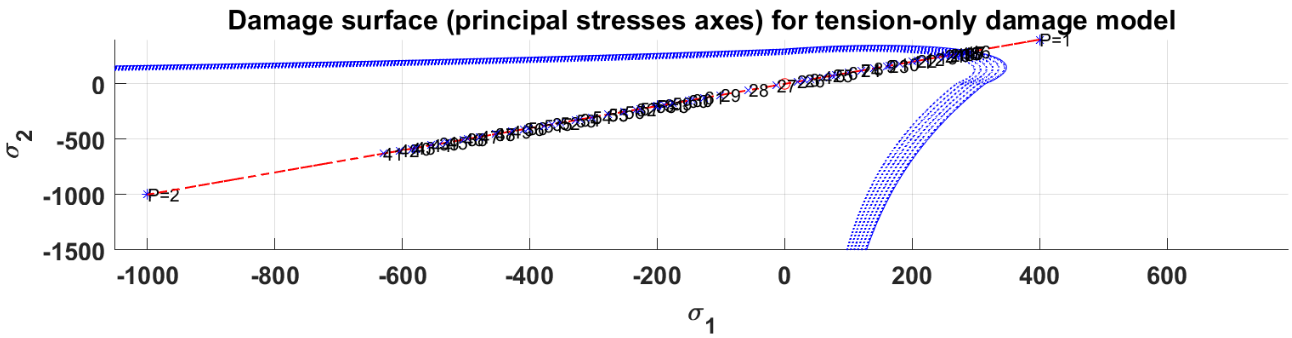

Figure 9: Stress paths and damage surfaces.

2.2 Non-symmetric tension-compression damage model

Here it will be seen again that all the stress points will behave elastically when remaining inside the domain, and all of them will remain to be inside or on the damage surface. Again this is due to the criteria of rate independent damage models. However, here materials which present different behavior when subjected to tension or compression will be analyzed, and damage by compression will be also seen. The same stress paths will be used in this section. Again, it is worth mentioning that the parameters were chosen so that the damage for compression is also seen.

2.2.1 Uni-axial tensile loading and unloading

It is not possible to appreciate much of the stress path from Fig. 11 in this case, due to the already-built graphic interface. However, it is possible to see the extreme points. From Fig. 10, the stress-strain graph shows that there is indeed damage for compression, and the Young's Modulus is decreased further as the slope is reduced even more. This is proof that the non-symmetric damage model works, at least for the introduced parameters.

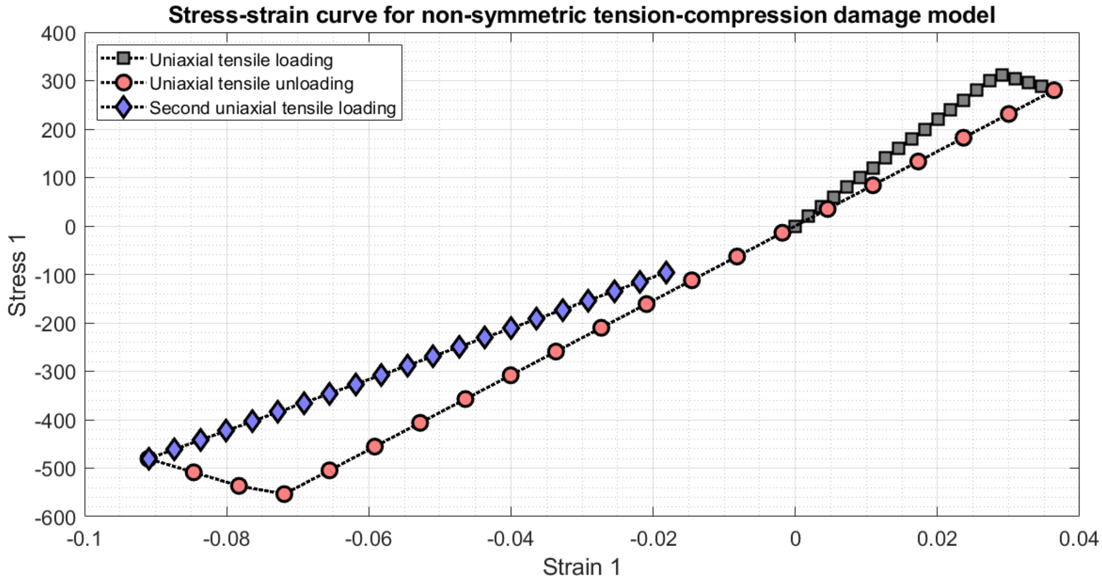

Figure 10: Stress-strain curve for the three proposed loading paths.

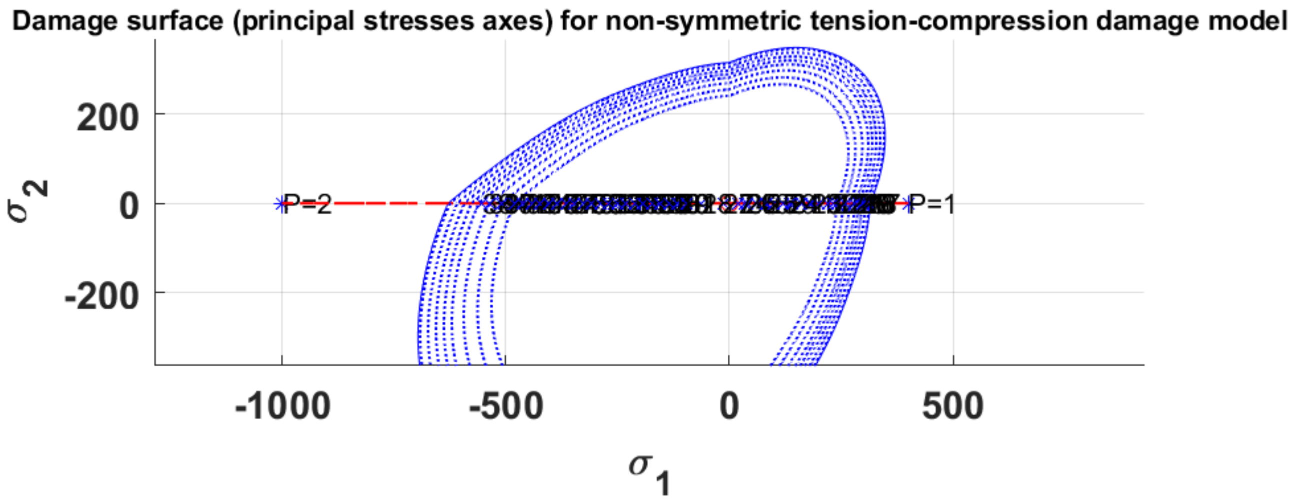

Figure 11: Stress paths and damage surfaces.

2.2.2 Uni-axial tensile loading and bi-axial tensile unloading

The most notorious information that the stress path brings here is that it can be seen that there are stress points lying outside the damage surface. The stress-strain curve (Fig. 12) is equal to that of the tension-only damage model (as could be expected) for the uni-axial and bi-axial tensile loading, but when reaching the corresponding damage surface the inelastic region is reached and the unloading path will thus be along a different line.

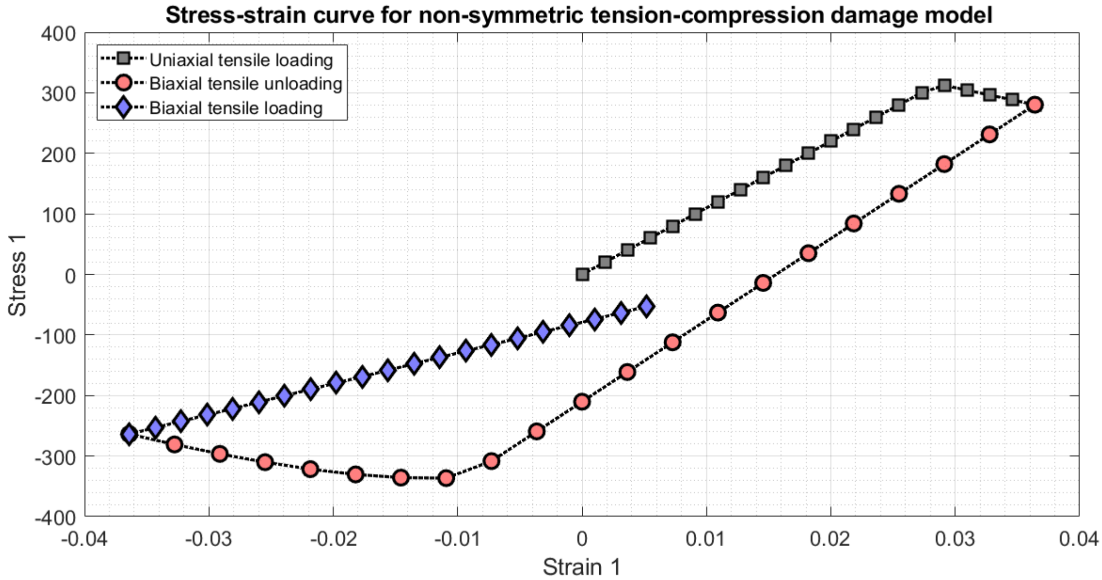

Figure 12: Stress-strain curve for the three proposed loading paths.

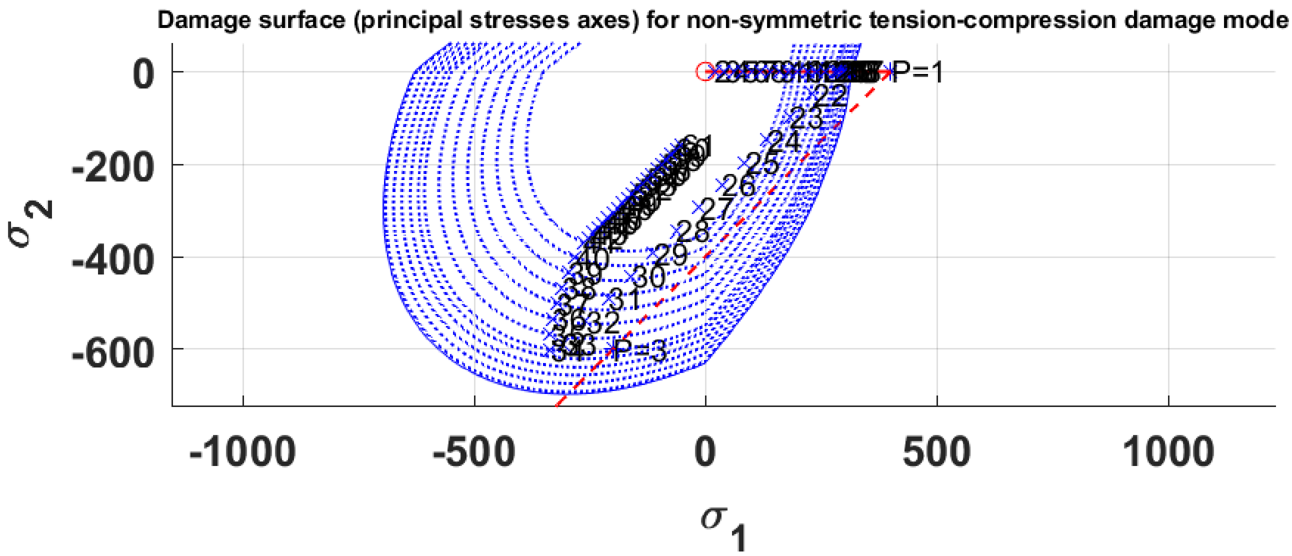

Figure 13: Stress paths and damage surfaces.

2.2.3 Bi-axial tensile loading

Again, points falling out of the domain are seen in the stress domain, all along a same line, apparently. When studying the stress-strain curve, it is noticed again that nearly the same shape is obtained as in Fig. 10 but in this case the values are much different. As can be appreciated, when only uni-axial loading and unloading, the damage by compression is reached at a far larger strain (in absolute value) that at the present case. In uni-axial tensile loading, damage by compression is produced for $\epsilon = -0.07$, and in the present case the damage is at $\epsilon = -0.04$, meaning that when there is tension in both axes, the capacity of the material is diminished. This could also be justified from the stress path, as done before.

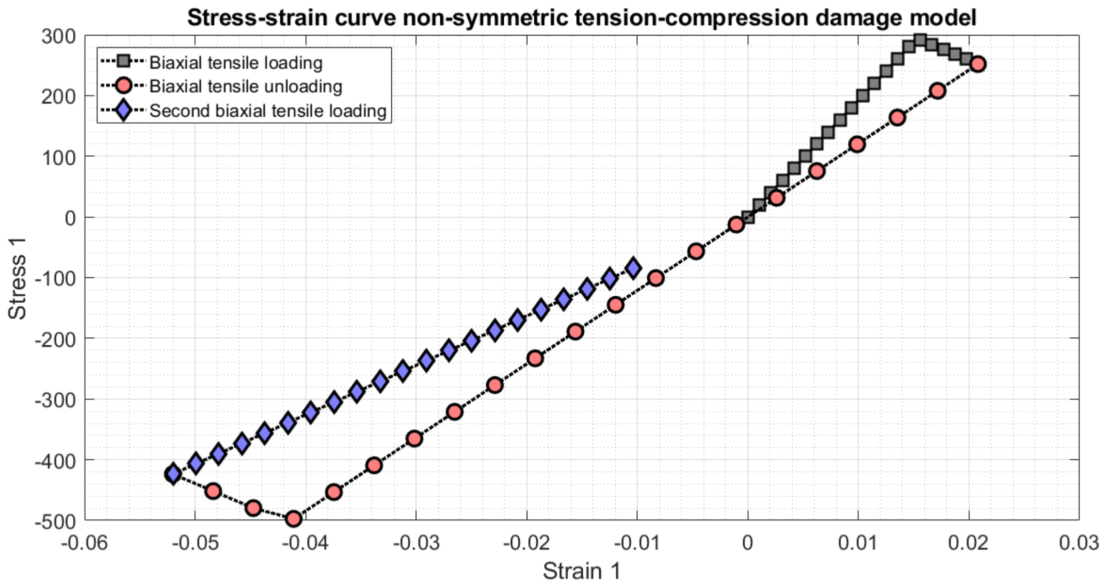

Figure 14: Stress-strain curve for the three proposed loading paths.

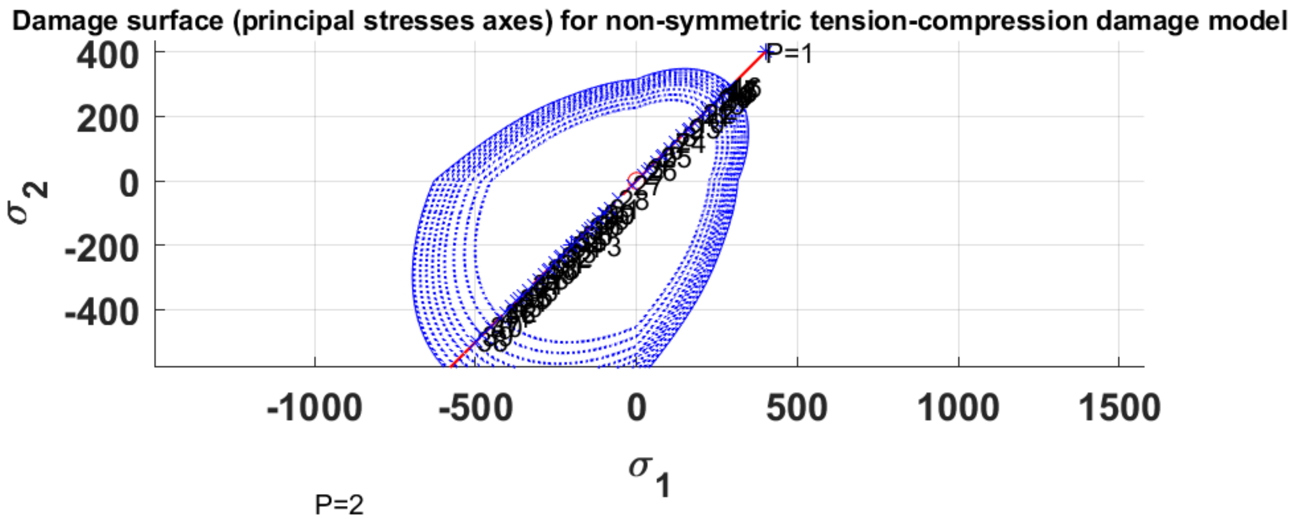

Figure 15: Stress paths and damage surfaces.

3 Rate dependent models

In this section the models will be adapted and characterized so as to take into account the rate with which the loads are applied, and then time will be considered as an independent variable and not as a parameter (rate dependent models). Therefore all the quantities of interest will have a dependency on time that will be accentuated by the election of the other variables that play a role in the calculations. Apart from the mentioned time (that will relate σ with $\partial\epsilon/\partial t$), the viscosity parameter η and the numerical integration parameter α will be studied.

For the current computations, a value of $\eta = 0.3$ and $\alpha = 0.5$ (the default value, since it is second order accurate) have been considered. However, some parameters have changed their value to a new one. Now, $n = 10, H = 0.4, \sigma_y = 200$.

The loading paths have been changed so that now it is clearer to see the effect of changing these parameters, which in some cases is only a slight difference. Now, starting from the point $(0, 0)$ the loading paths will be the following.

$$(\sigma_1^1, \sigma_2^1) = (200, 0) \longrightarrow (\sigma_1^2, \sigma_2^2) = (400, 0) \longrightarrow (\sigma_1^3, \sigma_2^3) = (100, 0)) \tag{3}$$

3.1 Effect of $\alpha, \eta, \dot{\epsilon}$ parameters on the stress-strain curves

Before commenting on the results of the graphics, however, it is important to think over which will be its effect and how to asses its correctness on the code.

The parameter which can be assessed more easily is the viscosity. By knowing the relation $(r) = 1 - q(r)/r$, and assuming that the $q(r)/r$ ratio diminishes when η increases, it follows that for lower values of viscosity, the damage will be higher and thus the loss of the capacity of the material to carry stresses will be more accentuated for lower values of viscosity. This is what Fig. 16 tells us, as for lower values of η a lower stress corresponds to the same strain. Another clear sign that the code is correct is the case in which $\eta = 0$, which corresponds to the non-viscous case, meaning that the rate independent damage model is recovered when $\eta = 0$ is fixed in the rate dependent model.

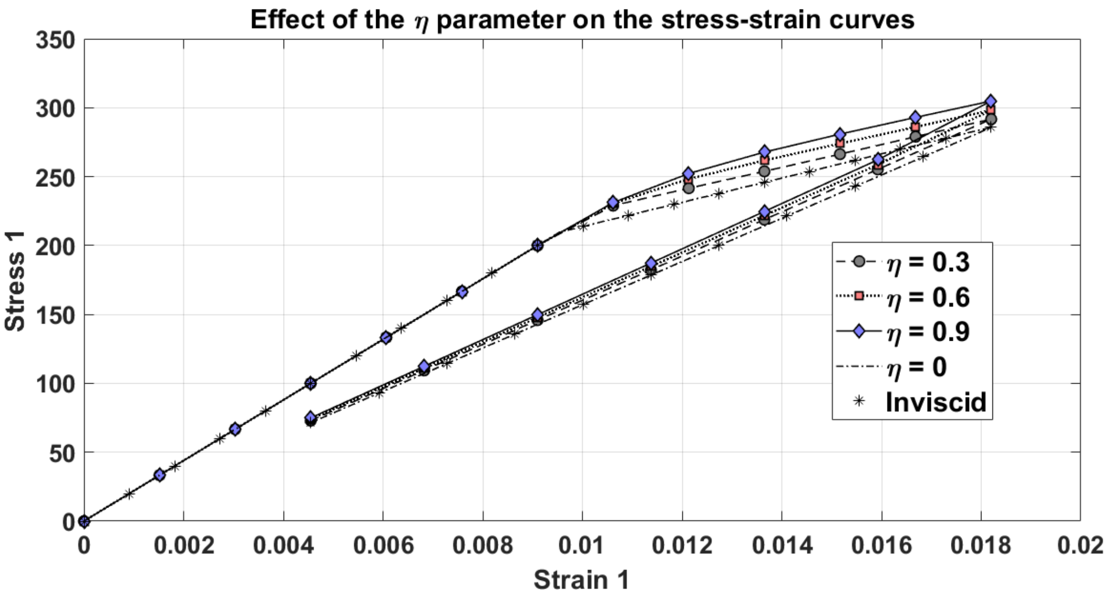

Figure 16: Effect of η on the stress-strain curve.

The main conclusions that Fig. 17 shows us when studying the effect of α on the stress-strain curve is that for the implicit integration ($\alpha = 1$), the curve reaches the the solution of the rate independent problem for the non-viscous case.

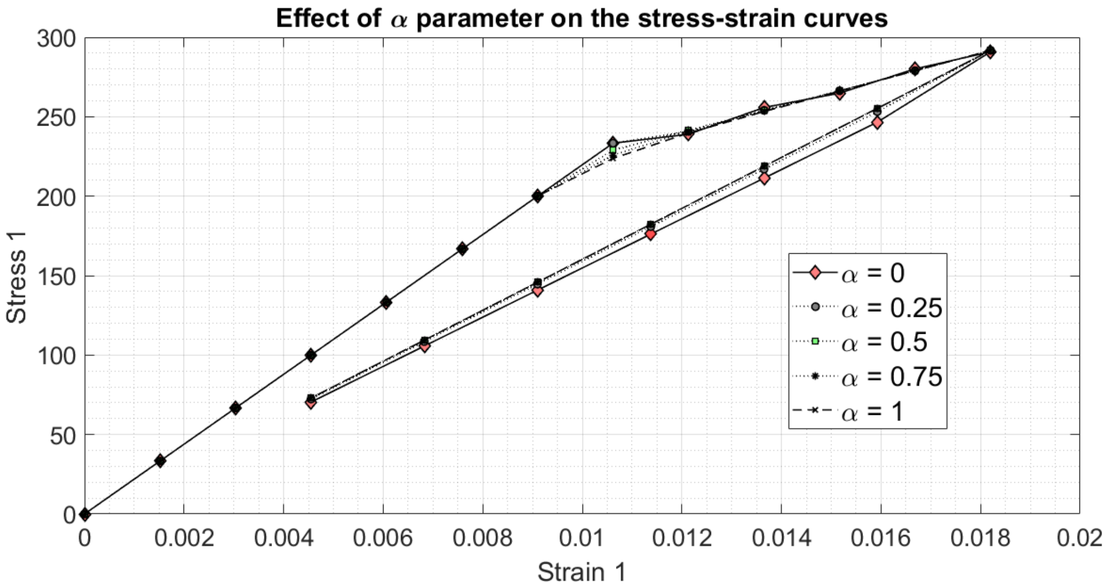

Figure 17: Effect of α on the stress-strain curve.The extreme values have been emphasized for clarity.

Eventually, Fig. 18 shows as expected the effect of applying stresses over a short period of time. As it was explained in lectures, high strain rates increases the elastic properties of the material under study, hence appreciating in the figure a minor loss of stress capacity. On the other hand, low strain rates are associated with the energy-damping aspects of the material characterization and hence the lower stress state for a same strain.

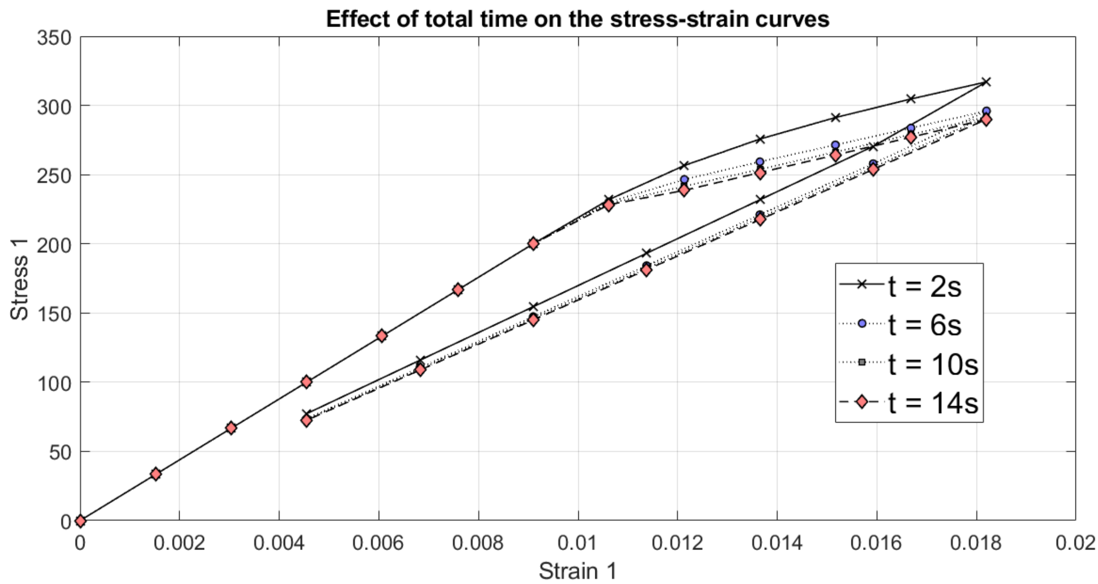

Figure 18: Effect of $\dot{\epsilon}$ on the stress-strain curve.The extreme values have been emphasized for clarity.

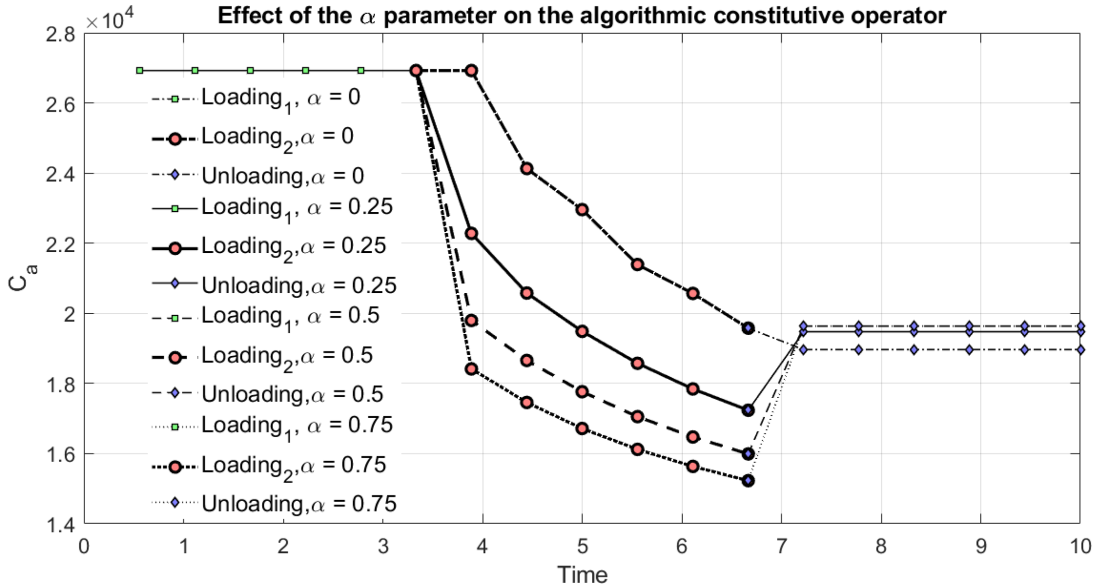

Figure 19: Evolution along time of the $\mathbb{C}^{vd}_{alg}$ constitutive operator for different values of α. The values of the loading path which produces damage are highlighted.

3.2 Effect of α on the constitutive operators

The analytic tangent and algorithmic tangent operators are related to each other by the following expression,

$$\mathbb{C}^{vd}_{alg,n+1} = \mathbb{C}^{vd}_{tan,n+1} - \frac{1}{\frac{\eta}{\alpha \Delta t} + 1} f(d'(r_{n+1})) \tag{4}$$

Where $f(d'(r_{n+1}))$ is a function that depends on the damage ratio so that if there is no damage evolving $f = 0$. With this expression, two important conclusions can be extracted that will be used to assess the graphical results.

- The algorithmic and analytical tangent operators match whenever there is no damage evolving or $\alpha = 0$.
- For increasing values of α, higher will be the subtraction from the $\mathbb{C}^{vd}_{tan,n+1}$ part so that the difference between both graphs will be more noticeable.

This can be seen from Figures 19 and 20. When $\alpha = 0$, both graphics are the same. Then, when the yielding stress is reached during the second loading path (red markers in the figures), damage starts producing and hence both graphics start differing one from the other for $\alpha \neq 0$. This difference becomes more relevant in Fig. 19 for increasing values of α, as justified before. The second loading path values have been highlighted in the previous graph for higher clarity.

Eventually, when the unloading starts (blue markers), $f(d'(r_{n+1})) = 0$ and both graphics become the same again for the unloading path.

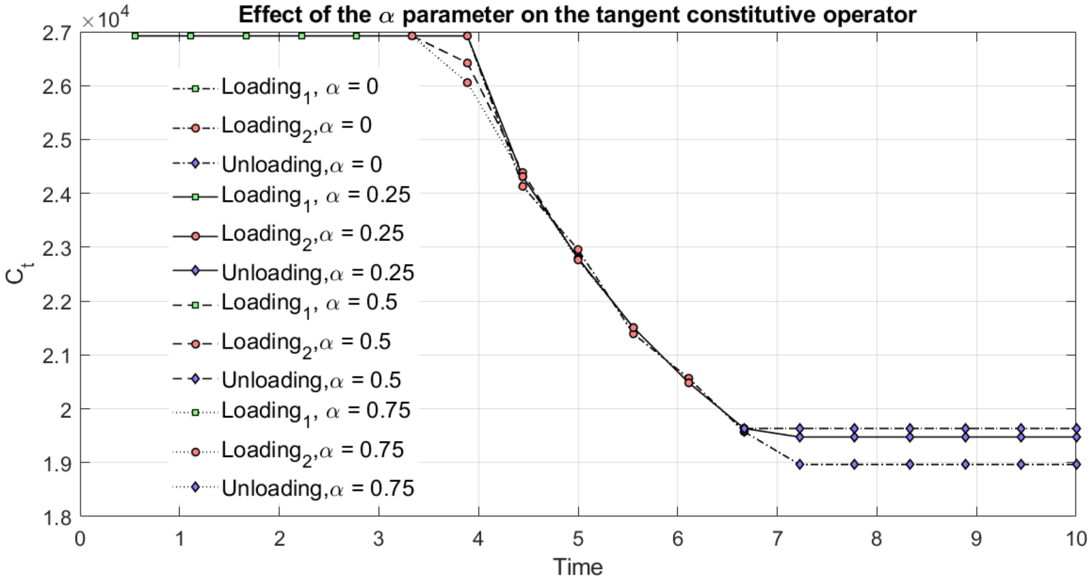

Figure 20: Evolution along time of the $\mathbb{C}_{tan}^{vd}$ constitutive operator for different values of α.

4 Concluding remarks

This work has laid down interesting conclusions over the behavior of certain materials under stresses exceeding their ultimate strength. It has been studied the code response for different rate dependent and rate independent models. More specifically, the following conclusions have been reached regarding the assessment of the developed code:

- For rate independent models, the stress points remain inside the elastic domain or on the boundary of the damage surface. In turn, points lying outside the domain have been seen for the rate dependent models.
- The Young's Modulus has been seen to diminish when damage has been produced in the material micro-structure, both for compression and tension. When applying models for only tension, this damage was only produced for values of the principal tension exceeding σ_y. On the other hand, the non-symmetric damage model is able to predict the failure of the material for compression loading.
- The effects of the viscosity parameter have been computed and explained according to their theoretical effects, that is, the capacity of the viscosity to alter the elastic properties of the material. The strain rates have a similar effect on the stress-strain curve as shown as well. The influence these parameters have on the damage variable will determine the magnitude of the alteration. The ability of the models to recover the rate independent solution has been also studied through the convergence of the η parameter to the non-viscous solution.

As a note on future work, it would be necessary to update the representation of the stress paths as in the majority of cases it becomes difficult to distinguish the results.

A Appendix

The following functions show the modified code without the majority of comments, to improve the reading purposes, as it is not necessary anymore to keep the comments.

A.1 Main program, *main_nointeractive*

```
clc
global X Y Vector_tan Vector_alg
close all

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Program for modelling damage model
% (Elemental gauss point level)
% -----------------
% Developed by J.A. Hdez Ortega
% 20-May-2007, Universidad Politécnica de Cataluña
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%profile on

addpath('AUX_SUBROUTINES')

YOUNG_M = 20000 ;
POISSON = 0.3;
HARDSOFT_MOD = -0.4;
q_inf = 3;
YIELD_STRESS = 300;
ntype= 2 ;
MDtype =2;
n = 2;
HARDTYPE = 'LINEAR' ; %{LINEAR,EXPONENTIAL}
VISCOUS = 'NO' ;
eta = 0;
TimeTotal = 10;
ALPHA_COEFF = 0.5;

nloadstates = 3 ;
SIGMAP = zeros(nloadstates,2) ;
SIGMAP(1,:) =[400 0];
SIGMAP(2,:) =[-1000 -1400];
SIGMAP(3,:) =[-200 -600];

istep = 10*ones(1,nloadstates) ;


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'

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 q_inf 0]            ;

[sigma_v,vartoplot,LABELPLOT_out,TIMEVECTOR]=damage_main(Eprop,ntype,istep,...
    strain_history,MDtype,n,TimeTotal);

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}) ;
    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,sigma_u) ;
```

A.2 Function *Modelos_de_dano1*

```
function [rtrial] = Modelos_de_dano1 (MDtype,ce,eps_n1,n)

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

elseif (MDtype==2)  %* Only tension
    sigma_bar = eps_n1*ce;
    sigma_bar_plus = (sigma_bar + abs(sigma_bar))/2;
    rtrial = sqrt (sigma_bar_plus*eps_n1');

elseif (MDtype==3)  %*Non-symmetric
    sigma_bar = eps_n1*ce;
    sigma_bar_plus = (sigma_bar + abs(sigma_bar))/2;
    wf = (sigma_bar_plus(1) + sigma_bar_plus(2) + ...
        sigma_bar_plus(4))/(abs(sigma_bar(1)) + ...
        abs(sigma_bar(2)) + abs(sigma_bar(4)));
    rtrial = sqrt(eps_n1*ce*eps_n1')*(wf+(1-wf)/n);
end
```

A.3 Function *plotcurvesNEW*

```
function plotcurvesNEW(DATA,vpx,vpy,LABELPLOT,vartoplot,sigma_u)
% Plot stress vs strain (callback function)
% -----------------------------------------
global X Y
subplot(2,1,2)
hold on
grid on
xlabel(vpx);
ylabel(vpy);


switch  vpx
    case 'STRAIN_1'
        strx = 'X(i) = DATA.strain(i,1);' ;
        %strx = 'X(i) = max(DATA.strain(i,1),DATA.strain(i,2));' ;
    case 'STRAIN_2'
        strx = 'X(i) = DATA.strain(i,2);' ;
        %strx = 'X(i) = min(DATA.strain(i,1),DATA.strain(i,2));' ;
    case '|STRAIN_1|'
        strx = 'X(i) = abs(DATA.strain(i,1));' ;
        %strx = 'X(i) = abs(max(DATA.strain(i,1),DATA.strain(i,2)));' ;
    case '|STRAIN_2|'
        strx = 'X(i) = abs(DATA.strain(i,2));' ;
        %strx = 'X(i) = abs(min(DATA.strain(i,1),DATA.strain(i,2)));' ;
    case  'norm(STRAIN)'
        strx = 'X(i) =sqrt((DATA.strain(i,1))^2 + (DATA.strain(i,2))^2)) ;';
    case  'TIME'
        strx = 'X(i) =DATA.TIMEVECTOR(i) ;';
    otherwise
        for iplot = 1:length(LABELPLOT)
            switch vpx
                case LABELPLOT{iplot}
                    %strx =  ['X(i) = vartoplot{i}(',num2str(iplot),') ;'];
            end
        end
end

X = 0 ;
for i = 1:size(DATA.strain,1)
    eval(strx) ;
end

switch  vpy
    case 'STRESS_1'
```

```
        stry = 'Y(i) = DATA.sigma_v{i}(1,1);' ;
        %stry = 'Y(i) = max(DATA.sigma_v{i}(1,1),DATA.sigma_v{i}(2,2));' ;
    case 'STRESS_2'
        stry = 'Y(i) = DATA.sigma_v{i}(2,2);' ;
        %stry = 'Y(i) = min(DATA.sigma_v{i}(1,1),DATA.sigma_v{i}(2,2));' ;
    case '|STRESS_1|'
        %stry = 'Y(i) = abs(max(DATA.sigma_v{i}(1,1),DATA.sigma_v{i}(2,2)));' ;
        stry = 'Y(i) = abs(DATA.sigma_v{i}(1,1));' ;
    case '|STRESS_2|'
        %stry = 'Y(i) = abs(min(DATA.sigma_v{i}(1,1),DATA.sigma_v{i}(2,2)));' ;
        stry = 'Y(i) = abs(DATA.sigma_v{i}(2,2));' ;
    case  'norm(STRESS)'
        stry = 'Y(i) = sqrt((DATA.sigma_v{i}(1,1))^2+(DATA.sigma_v{i}(2,2))^2);' ;
    case  'DAMAGE VAR.'
        stry = 'Y(i) = sqrt((DATA.sigma_v{i}(1,1))^2+(DATA.sigma_v{i}(2,2))^2);' ;

    otherwise

        for iplot = 1:length(LABELPLOT)
            switch vpy
                case LABELPLOT{iplot}
                    stry =  ['Y(i) = vartoplot{i}(',num2str(iplot),') ;'];
            end
        end

end

Y = 0 ;
for i = 1:length(DATA.sigma_v)
    try
        eval(stry);
    end
end

for i=1:length(X)
    text(X(i),Y(i),num2str(i));
end

for i = 1:length(DATA.sigma_v)
        s1(i) = DATA.sigma_v{i}(1,1);
        s2(i) = DATA.sigma_v{i}(2,2);
end

ct = (length(X)-1)/3;
figure
plot(X(1:ct+1),Y(1:ct+1),':ks',...
```

```
    'LineWidth',2,...
    'MarkerSize',10,...
    'MarkerEdgeColor','k',...
    'MarkerFaceColor',[0.5,0.5,0.5])
hold on

plot(X(ct+1:2*ct + 1),Y(ct+1:2*ct + 1),':ko',...
    'LineWidth',2,...
    'MarkerSize',10,...
    'MarkerEdgeColor','k',...
    'MarkerFaceColor',[1,.5,.5])

plot(X(2*ct + 1:end),Y(2*ct + 1:end),':kd',...
    'LineWidth',2,...
    'MarkerSize',10,...
    'MarkerEdgeColor','k',...
    'MarkerFaceColor',[0.5,.5,1])

set(gca,'FontSize',14)
xlabel('Strain 1');
ylabel('Stress 1');
strain_damage_t = interp1(Y(end-8:end),X(end-8:end),sigma_u);
strain_damage_c = interp1(Y(ct+1:2*ct -2),X(ct+1:2*ct -2),-sigma_u);
ShadePlotForEmpahsis([strain_damage_t X(end)],'r',0.5);
ShadePlotForEmpahsis([strain_damage_c X(2*ct + 1)],'r',0.5);
legend({'Uniaxial tensile loading','Uniaxial tensile unloading',...
    'Second uniaxial tensile loading','Damaged region'},'Fontsize',...
    12,'Location','Northwest');
title('Stress-strain curve');

grid on; grid minor
```

A.4 Function *rmap_dano1*

```
function [sigma_n1,hvar_n1,aux_var,Ce_tan,Ce_alg] = rmap_dano1 (eps_n1,...
    hvar_n,Eprop,ce,MDtype,n)

hvar_n1   = hvar_n;
tau_n     = hvar_n(4);
r_n       = hvar_n(5);
q_n       = hvar_n(6);
E         = Eprop(1);
H         = Eprop(3);
sigma_u   = Eprop(4);
hard_type = Eprop(5);
viscpr    = Eprop(6);
eta       = Eprop(7);
alpha     = Eprop(8);
q_inf     = Eprop(9);
dt        = Eprop(10);

r0 = sigma_u/sqrt(E);
zero_q=1.d-6*r0;
                                                                              %*
[rtrial] = Modelos_de_dano1 (MDtype,ce,eps_n1,n);

if viscpr == 1 % Viscous case
    tau_alpha = (1-alpha)*tau_n + alpha*rtrial;

    if(tau_alpha > r_n) % Inelastic state
        fload=1; %damage
        r_n1 = ((eta - dt * (1-alpha))*r_n + dt * tau_alpha) /...
            (eta + alpha * dt);
    else % Elastic state
        fload=0;
    end
else % No viscous case
    if rtrial > r_n % Inelastic state
        fload = 1;
        r_n1 = rtrial;
    else % Elastic state
        fload = 0;
    end
end


if fload
    if hard_type == 0
```

```
        q_n1 = r0 + H*(r_n1 - r0);
    else
        q_n1= q_inf - (q_inf - r0) * exp(H*r0/(q_inf - r0)*(1-r_n1/r0));
    end

    if(q_n1<zero_q)
        q_n1=zero_q;
    end

else

    r_n1= r_n  ;
    q_n1= q_n  ;

end

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

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

if viscpr ==1
    if (rtrial > r_n)
        Ce_tan = (1-dano_n1)*ce;
        Ce_alg = (Ce_tan) -...
        (((alpha*dt)/(eta+alpha*dt))*(inv(rtrial))*...
        (q_n1 - H*r_n1)/(r_n1)^2)*((ce*eps_n1')*(ce*eps_n1')');
    else
        Ce_tan = (1-dano_n1)*ce;
        Ce_alg = Ce_tan;
    end
else
        Ce_tan = (1-dano_n1)*ce;
        Ce_alg = Ce_tan;
end

if viscpr
    hvar_n1(4)= rtrial;
end
hvar_n1(5)= r_n1 ;
hvar_n1(6)= q_n1 ;

aux_var(1) = fload;
aux_var(2) = q_n1/r_n1;

aux_var(1) = fload;
aux_var(2) = q_n1/r_n1;
```

A.5 Function *dibujar_criterio_dano1*

```
function hplot = dibujar_criterio_dano1(ce,nu,q,tipo_linea,MDtype,n)
%*******************************************************************
%*                  PLOT DAMAGE SURFACE CRITERIUM: ISOTROPIC MODEL

tetha=[0:0.01:2*pi];

D=size(tetha);
m1=cos(tetha);
m2=sin(tetha);
Contador=D(1,2);

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

if MDtype==1

    for i=1:Contador
        sT = [m1(i) m2(i) 0 nu*(m1(i) + m2(i))];
        radio (i) = q/sqrt(sT*(ce\sT'));
    end


elseif MDtype==2

    for i=1:Contador
        sT = [m1(i) m2(i) 0 nu*(m1(i) + m2(i))];
        sT_Macaulay = [(sT(1) + abs(sT(1)))/2, (sT(2) + abs(sT(2)))/2 0 ...
            (sT(4) + abs(sT(4)))/2];
        radio (i) = q/sqrt(sT_Macaulay*(ce\sT'));
    end

elseif MDtype==3

    for i=1:Contador
        sT = [m1(i) m2(i) 0 nu*(m1(i)+m2(i))];
        sT_Macaulay = [(sT(1) + abs(sT(1)))/2, (sT(2) + abs(sT(2)))/2 0 ...
            (sT(4) + abs(sT(4)))/2];
        wf = (sT_Macaulay(1) + sT_Macaulay(2) + sT_Macaulay(4))/(abs(sT(1)) +...
            abs(sT(2)) + abs(sT(4)));
        radio (i) = q/(sqrt(sT*(ce\sT'))*(wf+(1-wf)/n));
    end

end
```

```

for i=1:Contador
    s1(i)=radio(i)*m1(i);
    s2(i)=radio(i)*m2(i);
end

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

A.6 Function *damage_main*

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

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

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

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

totalstep = sum(istep) ;

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

[ce]    = tensor_elastico1 (Eprop, ntype);

eps_n1  = zeros(mstrain,1);

hvar_n  = zeros(mhist,1)  ;

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

```

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

for  iload = 1:length(istep)
    % Load states
    for iloc = 1:istep(iload)
        i = i + 1 ;
        TIMEVECTOR(i) = TIMEVECTOR(i-1)+ delta_t(iload) ;
        Eprop(10) = delta_t(iload);
        % Total strain at step "i"
        % ------------------------
        eps_n1 = strain(i,:) ;

        [sigma_n1,hvar_n,aux_var,Ce_tan,Ce_alg] = rmap_dano1(eps_n1,hvar_n,...
            Eprop,ce,MDtype,n);

        Vector_tan(i) = Ce_tan(1,1);
        Vector_alg(i) = Ce_alg(1,1);

        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

        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 ;

        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
```