Damage Model

Name: Oriol Falip i Garcia
Date: 12/04/2019

Introduction

Linear elasticity is probably the most well known and used theory when dealing with engineering problems. From Civil engineering, with big structures (bridges, buildings, silos or other structures), to Industrials, with mechanical components of an engine or a machine. Although those are just some examples, there are many other fields where linear elasticity is used.

However, nowadays engineers must face with more complex problems where linearity fails and is not enough, so new theories must arise to go beyond this limit. Those theories are able to capture richer phenomena such as large deformations where the system undergoes a permanent plastic deformation, the failure of a structure because of internal cracks due to large internal stresses arising from the loads applied, buckling, geometric non-linearities…

In this work, the so called Damage Model will be studied. The main goal of this model is to capture the effects produced by micro cracks that appear due to the internal stresses, making the structure to progressively suffer damage until, if it is the case, the failure of the system. This model is able to reproduce a wide variety of material behaviour, such as hardening, softening, traction/compression non-symmetry…
In what follows all those properties will be shown.

Objectives

This work has as main objectives showing the knowledge acquired about Damage Model during the course and also the implementation of a code which solves numerically this model. Part of the code is provided and only needs to be completed to implement different features. With the implementation and posterior presentation and analysis of the results a well understanding of the principal features of the model will be ensured.

Results

Damage Model allows to reproduce different material behaviour, providing a wide variety of phenomena. Different features will be shown and implemented progressively, starting by the simplest one, which is the symmetric model. Then, hardening and softening laws will be introduced, followed by only-tension and non-symmetric behaviour. This previous features will be implemented for rate independent cases at first and then, to finish this work, rate dependent behaviour will be reproduced.

Symmetric

In the symmetric case, the system has a linear elastic response with no distinction between compression and tension. This linear behaviour holds until a limit stress is reached (yield stress). At this point, if loads are increased, micro-cracks will damage the body and the system will undergo an inelastic deformation. *Figure 1.* shows the domain of loading states where the system behaves elastic (interior of the blue curves) for two material with different properties.

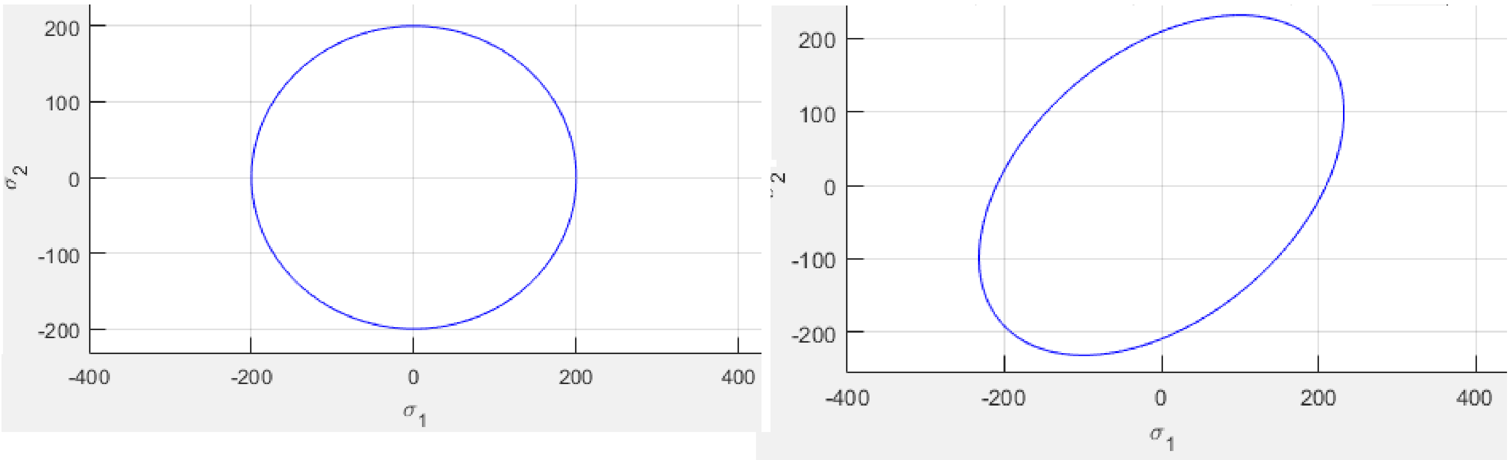

Figure 1. Elastic domain for an undamaged body with Poisson's ratio ν=0 (left) and ν=0.3 (right).

The previous figures shows the differences between materials with different Poisson's ratios. When there is no Poisson's effect, the elastic domain is a circumference with radios equal to the yield stress (σ_y), in this case σ_y=200. However, materials with non-zero ν, the elastic domain is elliptic since displacements in one direction will also produce a displacement in the other direction with its corresponding stress.

If this system is loaded as proposed in *Figure 2.*

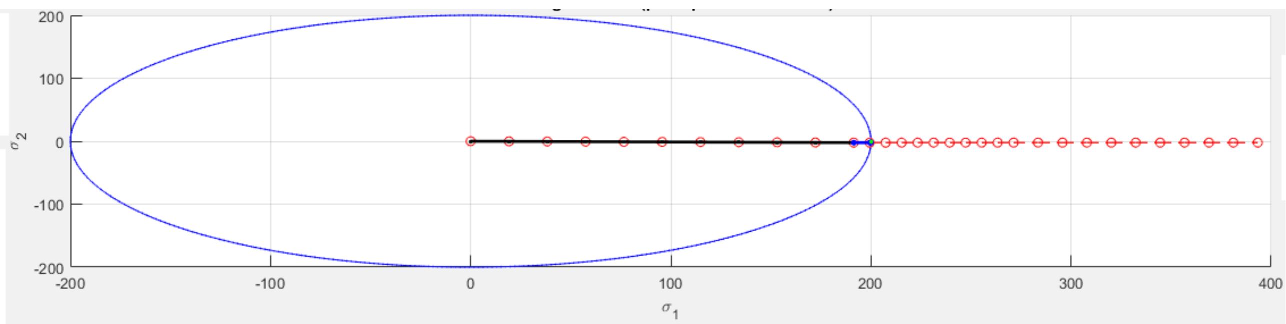

Figure 2. Representation of the loading path proposed (red dots) and material response (blue, dark, green dots) for a material with E=100, σ_y=200 and ν=0.

It must be remarked that the previous surface looks like an ellipse, in reality is a circumference. The reason for which it looks like an ellipse is that x-axis and y-axis are not at the same scale.

The apparent stress and displacement are:

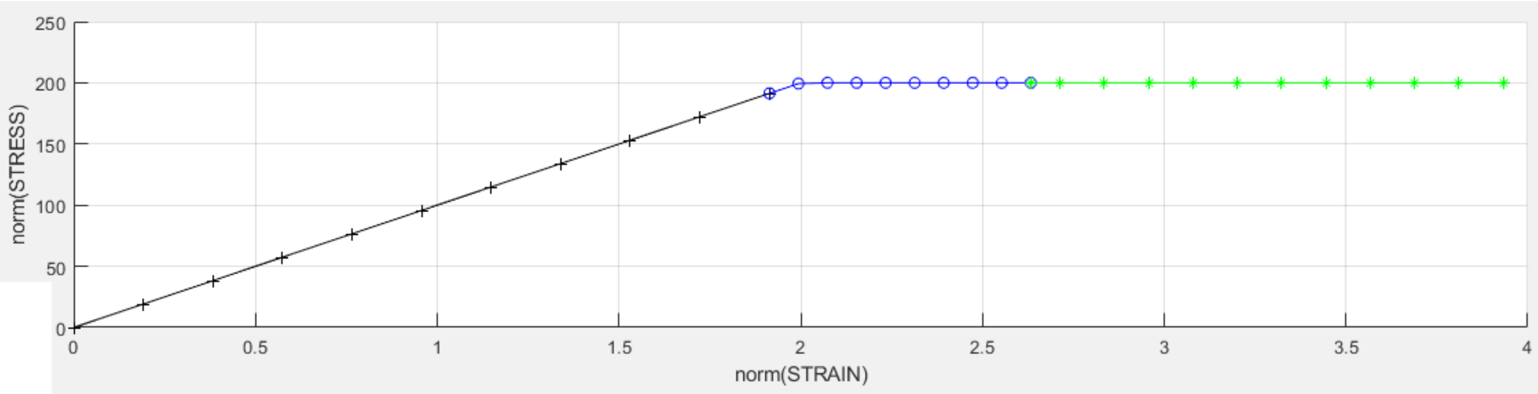

Figure 3. Relation between stress and strain when the system is loaded according to *Figure 2.*

The proposed path corresponds to an uni-axial tensile loading in the x-direction until σ_y is reached and further loading beyond this limit (red line).
From *Figure 3.* is seen that the elastic response is linear until yield stress is reached (black line). At this point, if the system is loaded, a pure inelastic response occurs and stresses no longer increase, even tough displacements enlarge (blue and green line).

An other important result is the one shown in *Figure 2.* When loading the system, the response is elastic until the blue line is reached. This blue line corresponds to the '**damage surface**'. At this point, the yield stress is reached and further loading implies damaging the body. Stress states beyond this damage surface are not allowed, that the reason why, even though further loading, blue and green dots accumulate at this point and stresses remain constant.

This model makes use of an internal variable (r) which is used to ensure irreversibility when our body is damaged. The evolution of 'r' is restricted by laws of thermodynamics which imposes that 'r' must always increase or remain constant, but never decrease. For the previous case, the evolution of 'r' is:

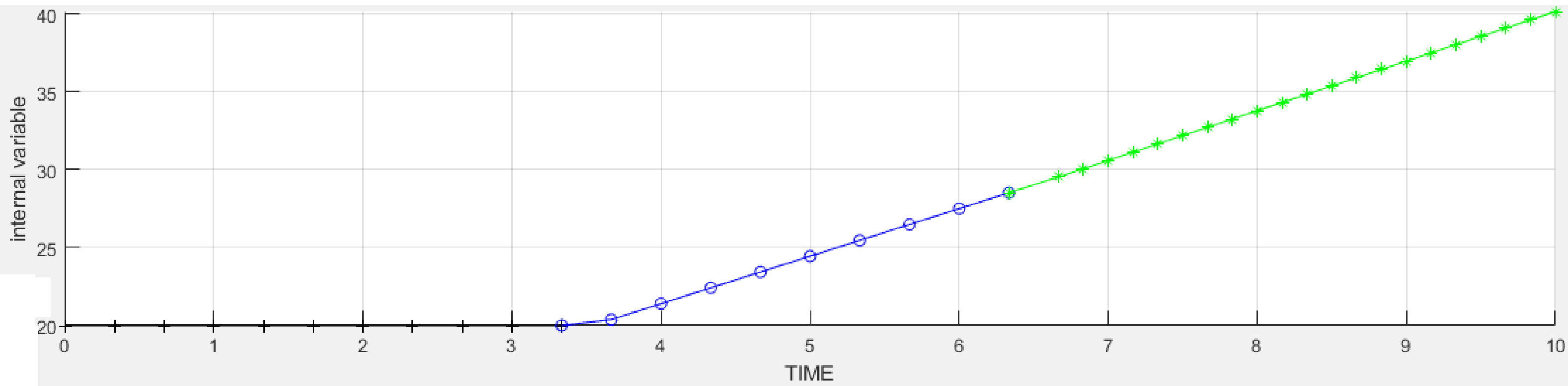

Figure 4. Internal variable 'r' versus time for the load path in *Figure 2*.

As it can be seen, the evolution of 'r' fulfils the restriction mentioned before. That is a necessary condition to ensure that the model is properly implemented.

Perfect inelasticity may not be enough to reproduce the real behaviour of materials. In some cases, when materials suffer damage, **hardening/softening** of the material may occur. This phenomena produces the material to modify its strength, making it stronger if it has a hardening response or weaker if it is the case of softening. Once this features are implemented, the results obtained are:

Linear hardening

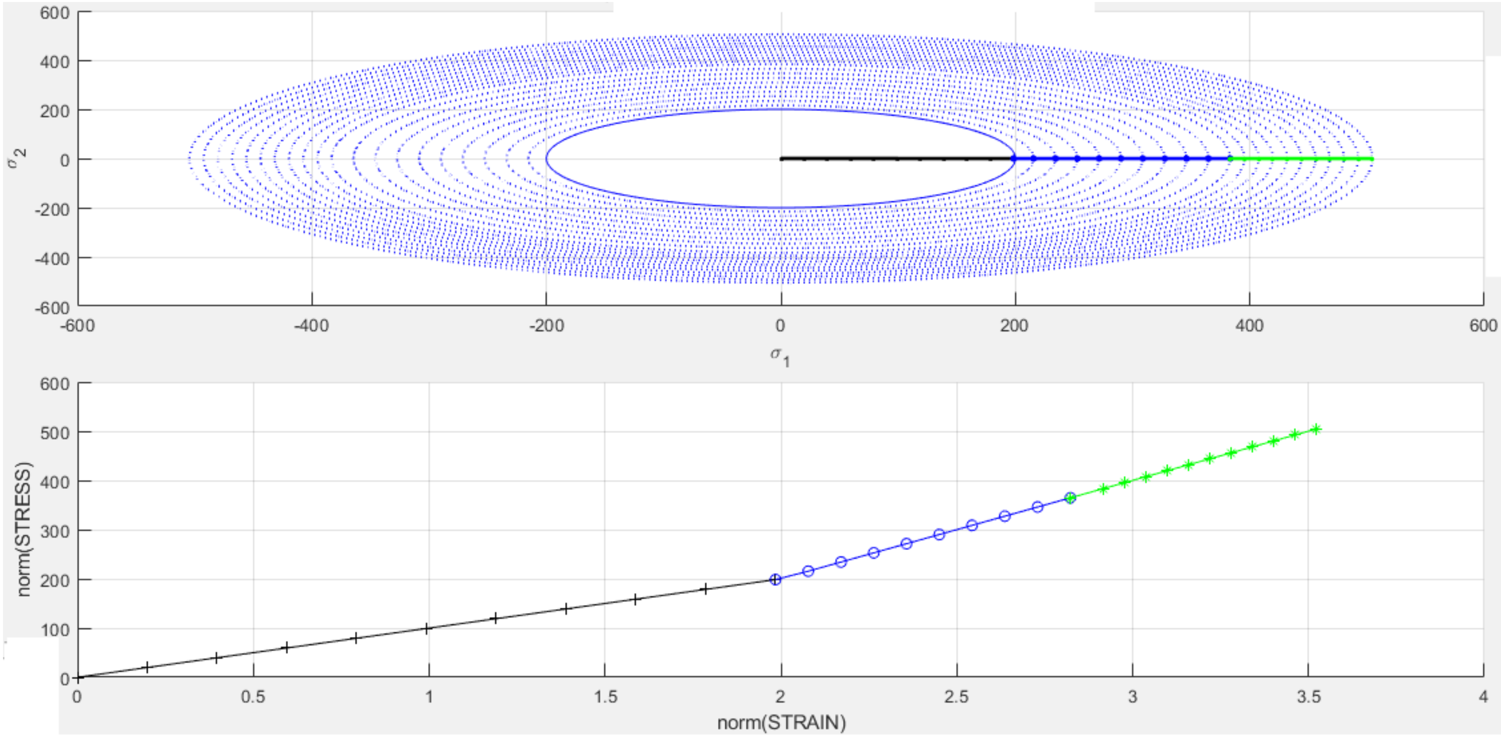

Figure 5. Elastic domain during time and loading path (top) and relation between stress and strain during the loading (bottom) for a material with E=100, σ_y=200 and ν=0.

Previous figure shown two main difference with respect to the pure inelastic case. First of all, the elastic response is the same as in the previous case. However, when damage surface is reached, the response is different. In this case, contrary to the inelastic behaviour, the stresses increase due to the linear hardening law.

Secondly, given that points beyond the 'damage surface' are not feasible, damage surface must enlarge in order to ensure that all load states are inside or right in the surface. The fact that when loading beyond yield stress, load points are, all the time, exactly in the 'damage surface' ensures that the model has been implemented properly.

A more realistic behaviour is the softening of the material. Intuitively, every day experience shows that when materials are damaged, its strength is reduced, producing a softening effect. In this case, the model results are:

Linear softening

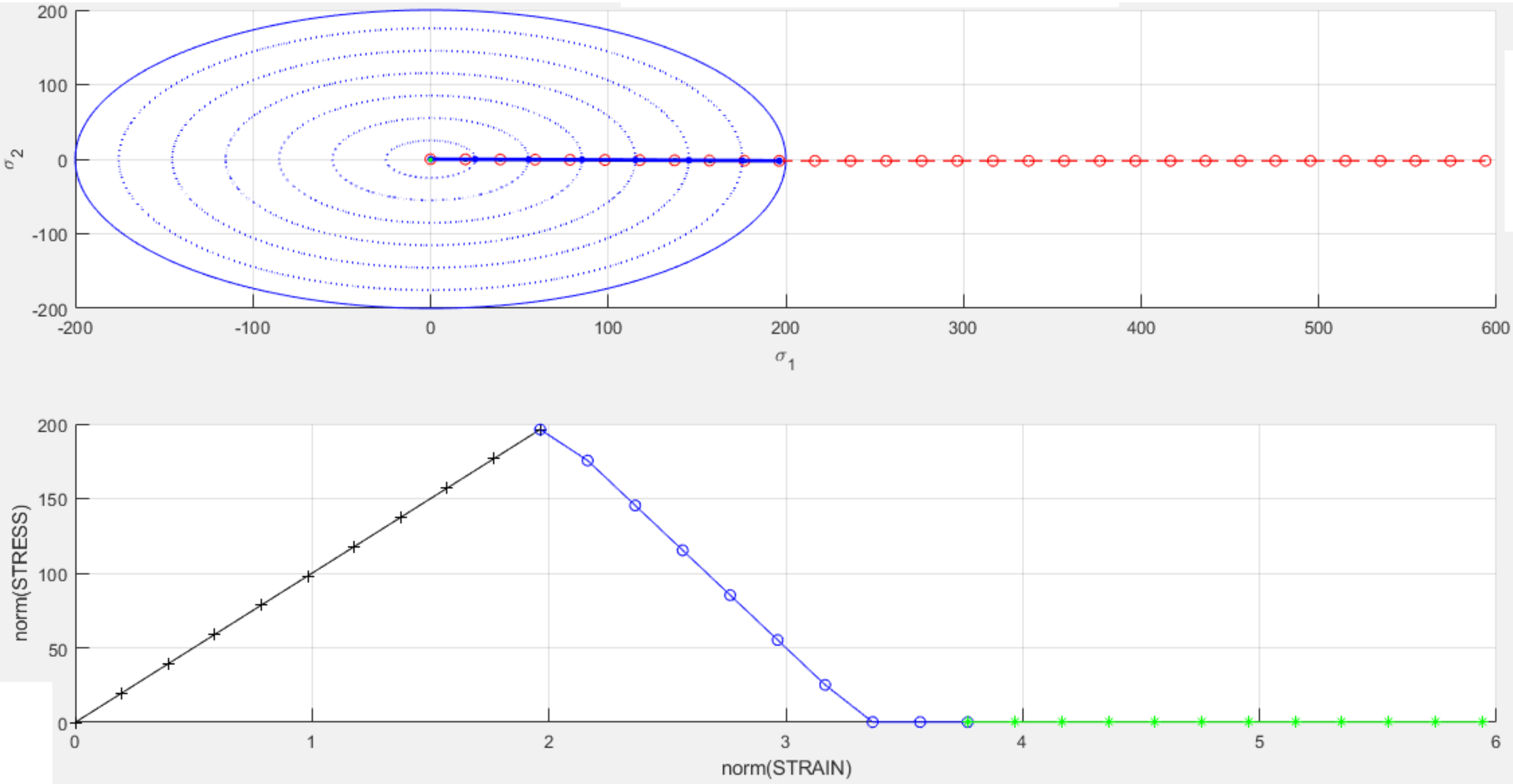

Figure 6. Elastic domain during time with proposed load state (red line) and stress response (blue) (top), and relation between stress and strain during the loading (bottom) for a material with E=100, σ_y=200 and ν=0.

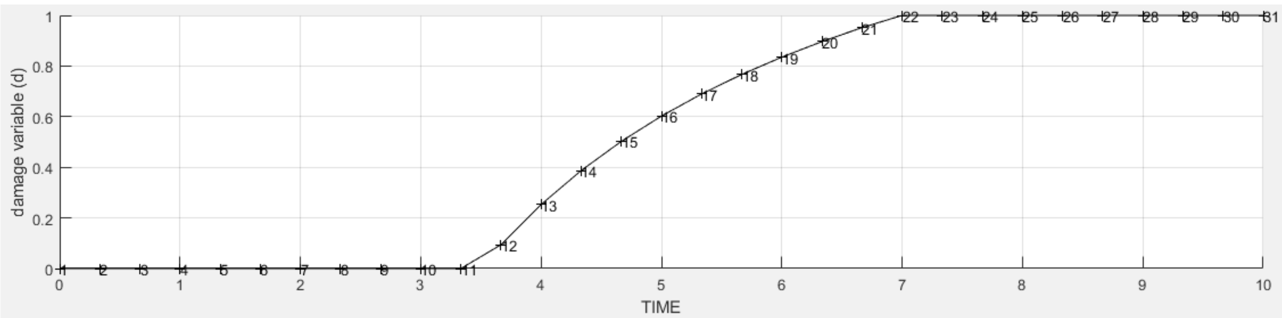

Figure 6.2. Damage versus time.

In this case, the behaviour is the same as in linear hardening with the main difference that stresses are reduced when the system is loaded beyond σ_y. This weakening happens until all its strength is lost, given that negative stress has no sense, once this point is reach, the stresses remain constant at σ=0.

An other important difference, is the evolution of the 'damage surface'. Now, loading beyond yield stress produces the 'damage surface' to 'shrink'. Given that points beyond blue line are not feasible, 'damage surface' pushes the stress points with it, enforcing those points to lie in the surface.

From *Figure 6.2.* we see the evolution of our damage variable versus time. As it can be seen, damage variable doesn't increase at the beginning. This is because yield stress has not been reached yet. Damage increase when loads beyond σ_y are applied. A feature which ensures that this model is well implemented is that damage maximum value is 1, since this means that our system is fully damaged. An other important fact is that even though stresses at the end are zero, due to softening, there is no healing, that is, once our system is damaged, it can't recover its original state.

Because of the fact that linear law has no limit, strength may increase indefinitely due to hardening. For that reason a linear response may not be enough to reproduce the behaviour of our material so other laws must be used to reproduce accurately the real behaviour. For instance, it may happen that hardening or softening may happened only until a limit is reached. In this case, an exponential law may be used to reproduce this behaviour.

Exponential hardening

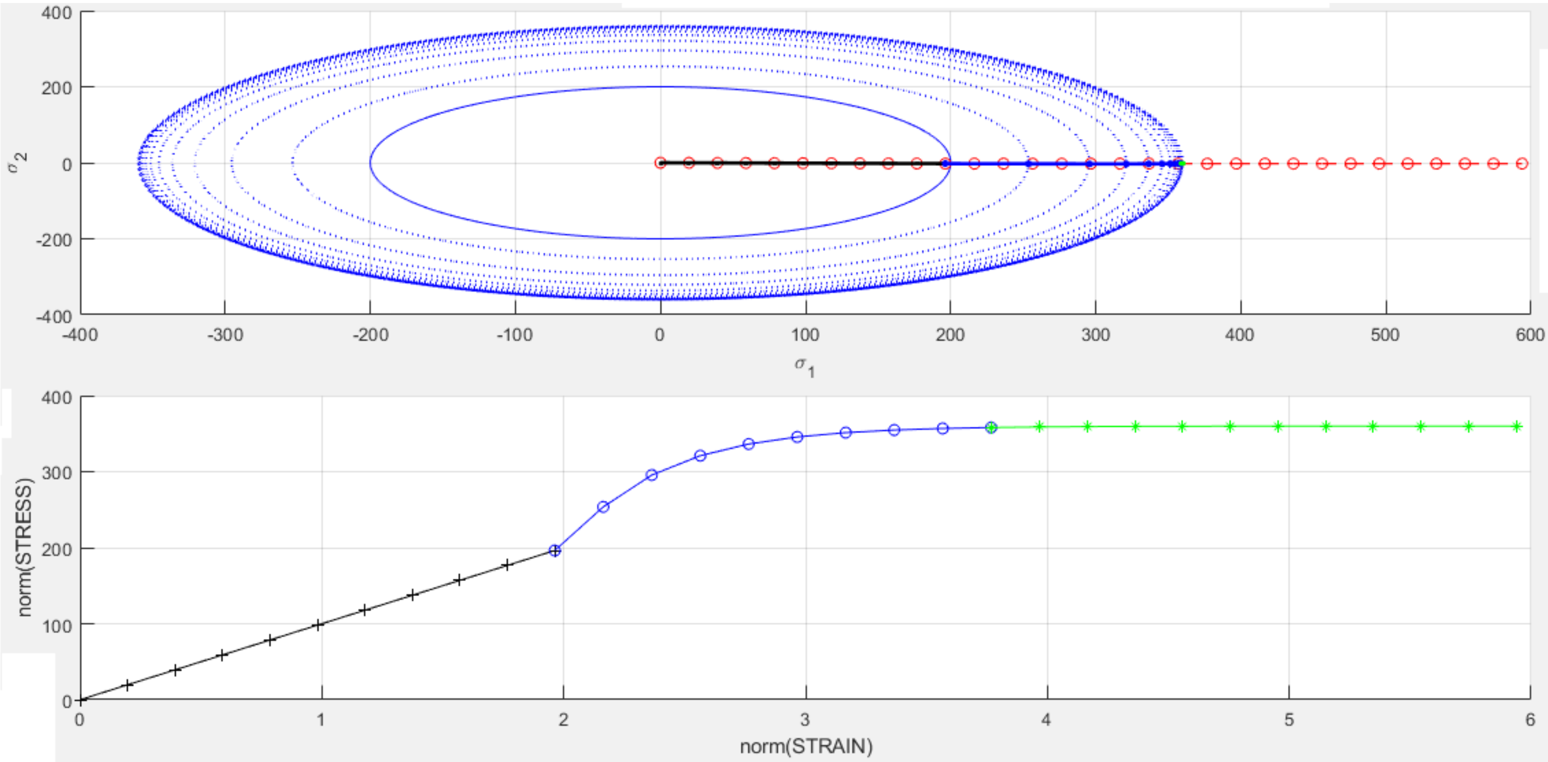

Figure 7. Elastic domain during time with proposed load state (red line) and stress response (blue) (top), and relation between stress and strain during the loading (bottom) for a material with E=100, σ_y=200 and ν=0.

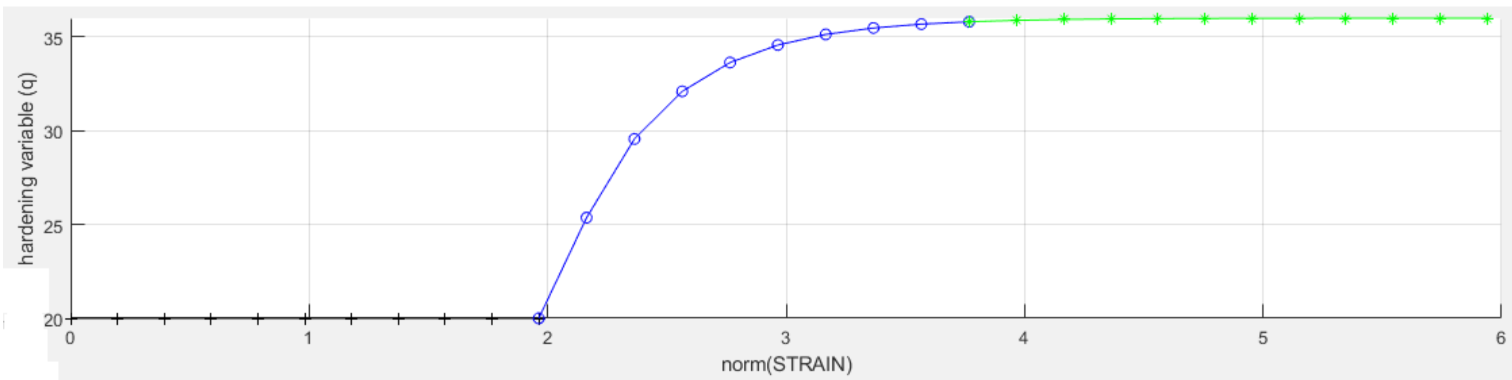

Figure 8. Hardening variable against strain for an exponential hardening law.

Previous figures show the same behaviour as in linear hardening except because of the fact that now hardening has a limit. *Figure 7.* shows that stresses increase due to hardening until a limit is reached, at this point, stresses remain constant. Regarding the evolution of the 'damage surface' it also enlarges in order to allow points beyond the initial 'damage surface'. **Hardening variable 'q'** controls the evolution and size of the damage surface. In *Figure 8.* the evolution of this hardening variable is represented. As it can be seen, hardening remains constant until yield limit is reached. At this point, if loads increase, hardening variable increases exponentially. While this variable increases, also the 'damage surface' enlarges.

Exponential softening

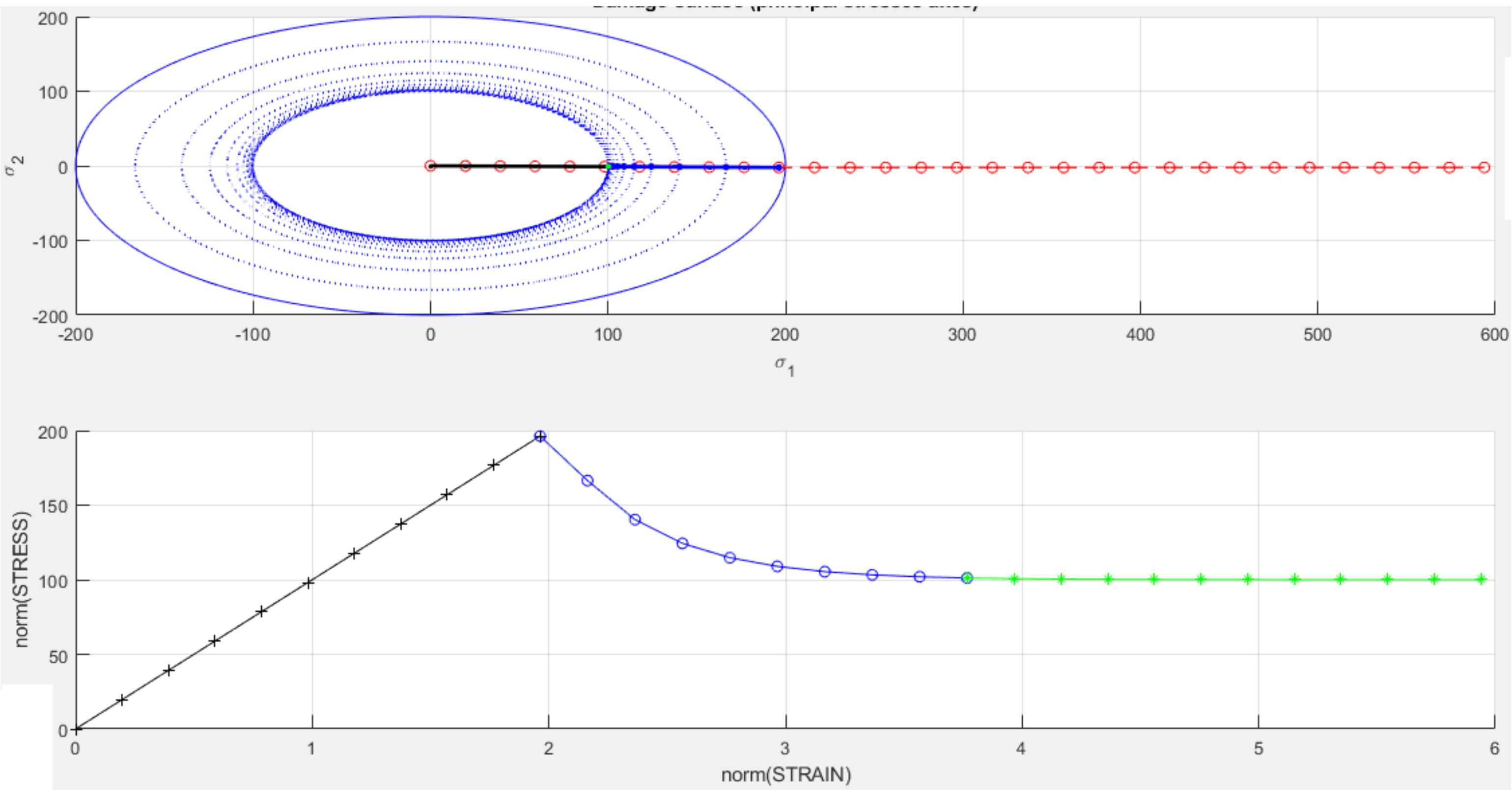

Figure 9. Elastic domain during time with proposed load state (red line) and stress response (black, blue, green) (top), and relation between stress and strain during the loading (bottom) for a material with E=100, σ_y=200 and ν=0.

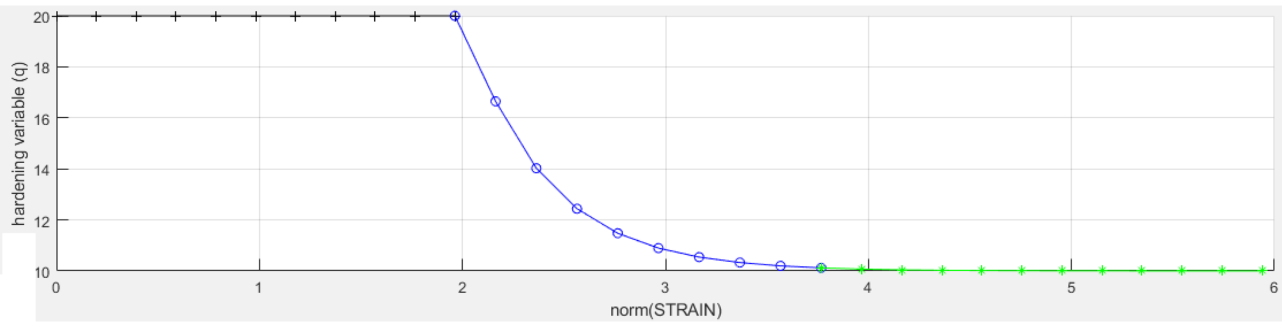

Figure 10. Hardening variable against strain for an exponential softening.

Exponential softening has the same properties as exponential hardening mentioned before.

In all previous examples, only uniaxial tensile loading has been simulated. The reason is that the purpose of those examples was to show the behaviour of the system during the elastic loading and loading beyond yield stress. Is important to remark that his fact doesn't mean any lose of generalization.

The following example corresponds to an uniaxial tensile loading beyond yield stress and posterior unloading to the equilibrium state for a material with exponential softening.

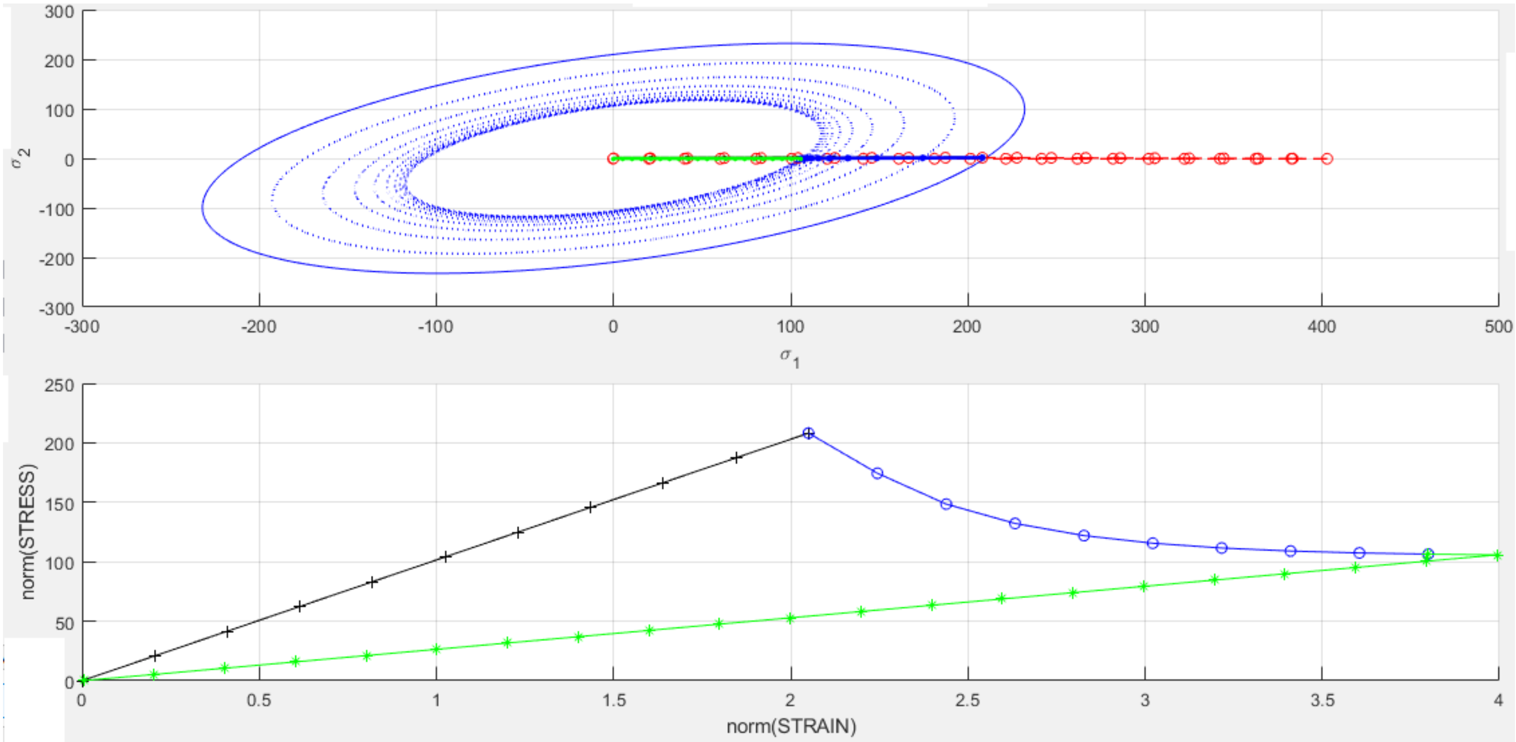

Figure 11. Elastic domain during time with proposed load state (red line) and stress response (black, blue, green) (top), and relation between stress and strain during the loading (bottom) for a material with E=100, σ_y=200 and ν=0.3.

The purpose of the previous example is to show the response of the material when it is loaded and suffers damage with posterior unloading. The remarkable fact here is that when the system is fully unloaded it always returns to the origin, no matter how much it has been loaded. This is an important feature because it means that there are no permanent displacements due to damage, as contrary with other models such as Plasticity models where, once yield stress is achieved, plastic deformation become permanent.

Tension only

When we want to reproduce a material that is only damaged when tension loads are applied, and no damage is produced when compressions loads are applied, symmetric model is not able to reproduce this behaviour so only tension model is build.

As it can be shown in *Figure 12.* damage surface is only defined when positive stresses (traction) are applied. No matter how much compressive loads are applied, no damage will be done.
It can also be seen the effect of Poisson's effect in the damage surface.

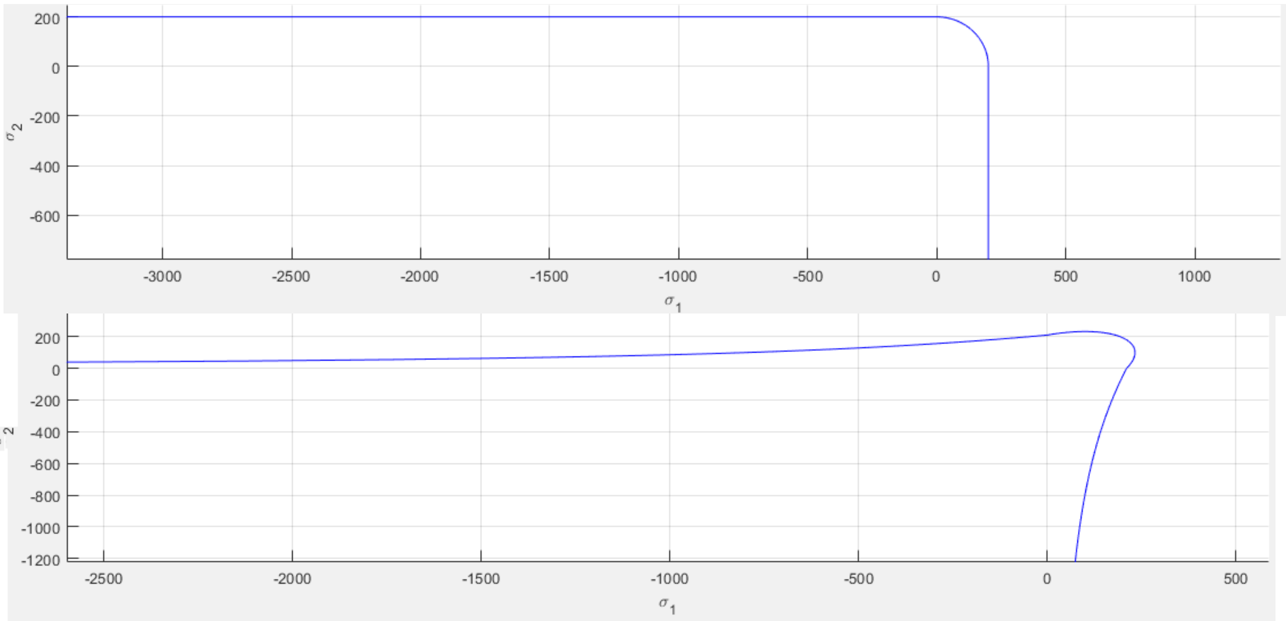

Figure 12. Initial elastic domain and damage surface for a material with E=100, σ_y=100 and ν=0 (top) and ν=0.3 (bottom)

In order to prove that compressive loads behave elastically for any value of the stress the following figure provides the evidence.

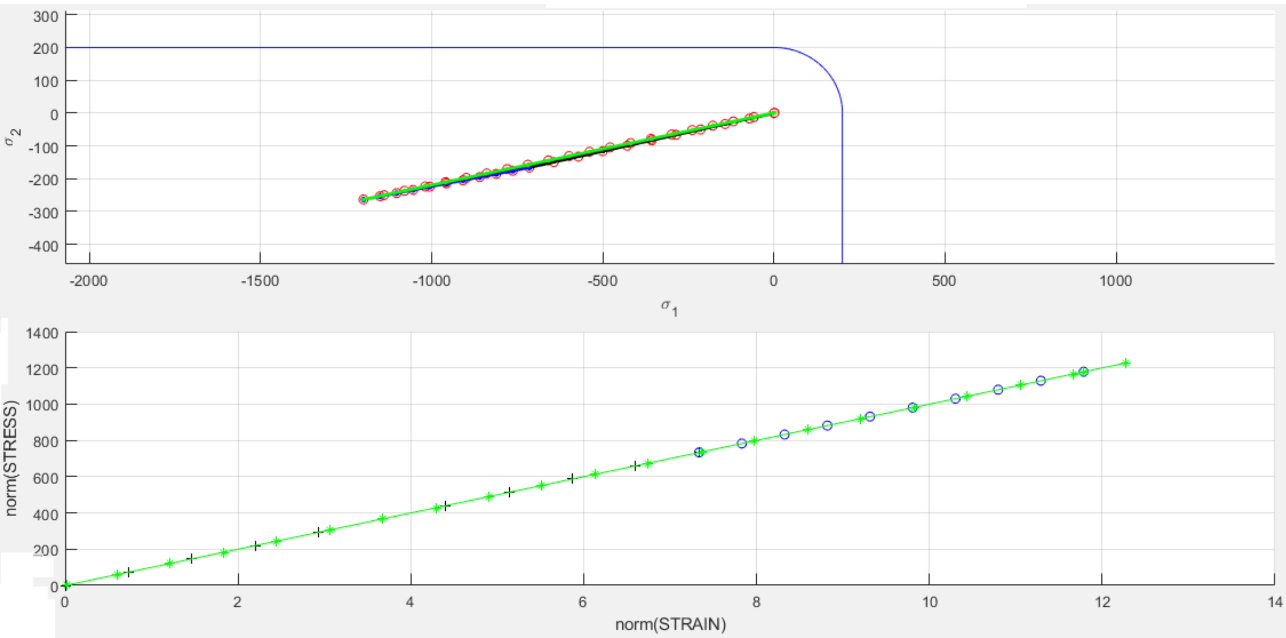

Figure 13. Biaxial pure compressive loading path (top) and relation between stresses and displacements (bottom) for a material with E=200, σ_y=100 and ν=0.

If instead of compressive loads, tensile loads are applied, as expected, linear response is obtained until yield stress is achieved, after that, the response is not linear. In this case, our material has exponential hardening.

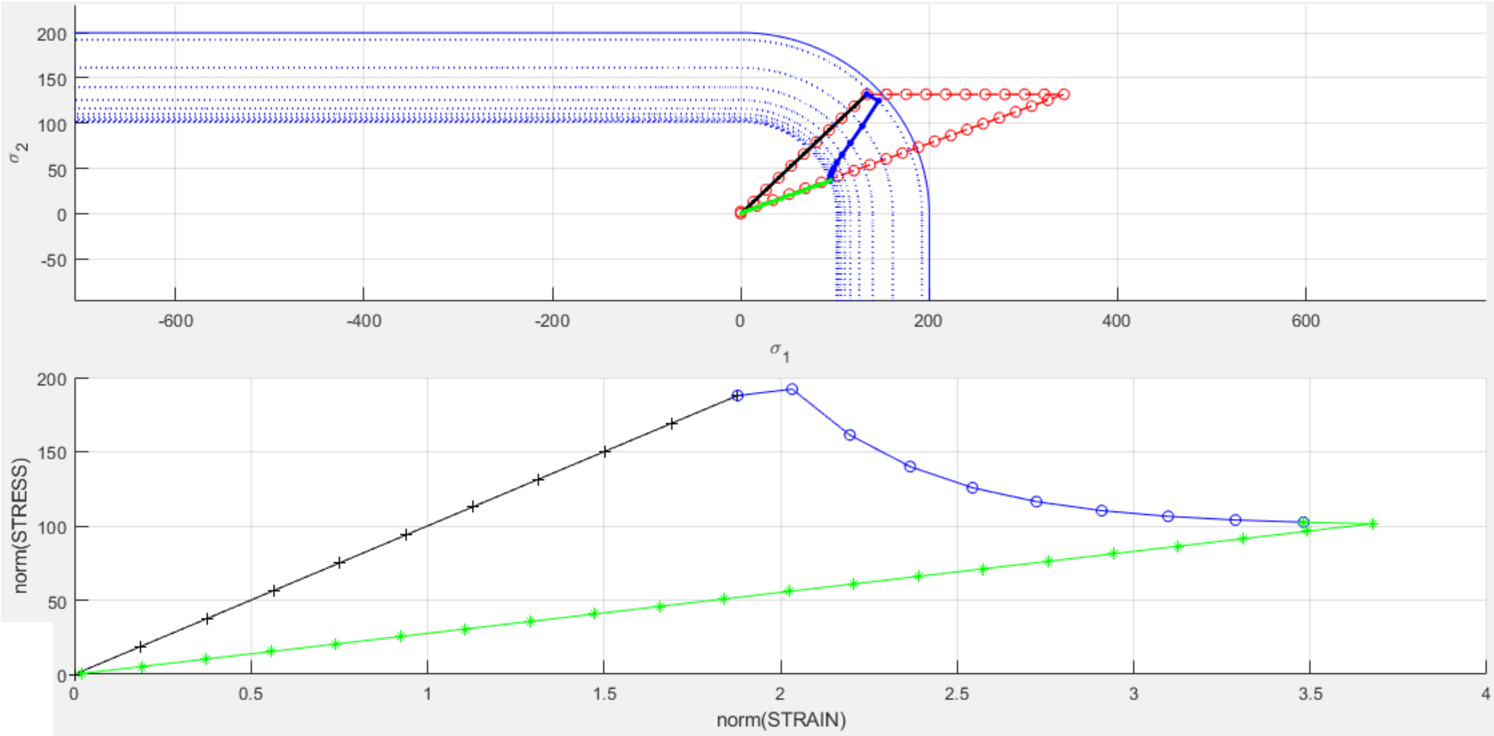

Figure 14. Loading path and damage surface for each time (top) and relation between stresses and displacements (bottom) for a material with E=200, σ_y=100 and ν=0.

In this case, material behaves with exponential softening when σ_y is reached. This shows that damaging and softening happens when tensile loads are applied. With that figure and the previous one, it is possible to ensure that the implementation of this model has been done properly.

Contrary to the previous cases this example tries to include all features and be a general problem. Here, the loading path is in both directions, softening effects are observed and also unloading effects with no permanent displacements.

Non-Symmetric

In many cases, only-tension model is not enough to reproduce the real behaviour of our system. The fact that compressive loads has no limit may not be accurate enough so as to get good results for our problem. An example of that is concrete. Concrete can hold compressive loads to a limit which is larger than the limit for tensile loads. This means that when yield stress is much bigger for compression loads and lower for tensile lodes.

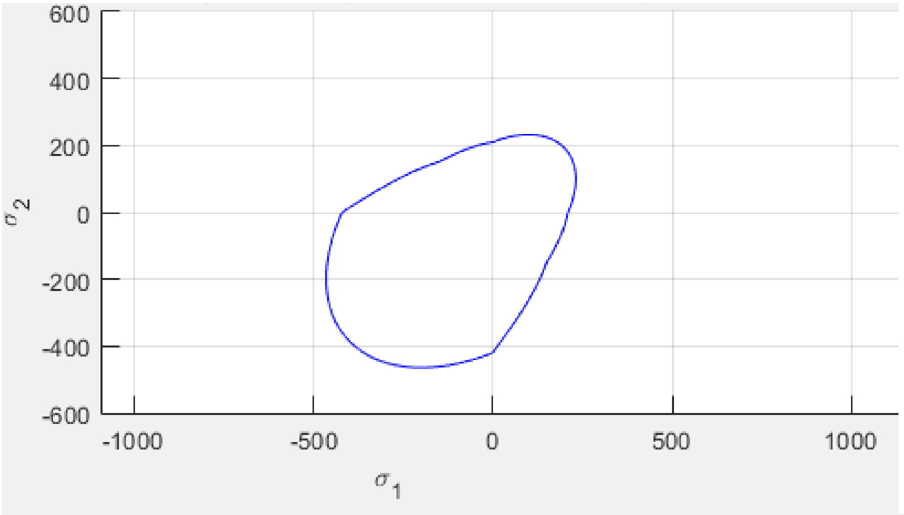

Figure 15. Elastic domain and damage surface for undamaged material with ν=0.3.
From the previous figure, we clearly see the non-symmetric behaviour. Damage surface is closer to the origin for tensile loads, and wider for compressible ones. This means that our system will start suffering damage earlier to tensile loads than for compressible loads.

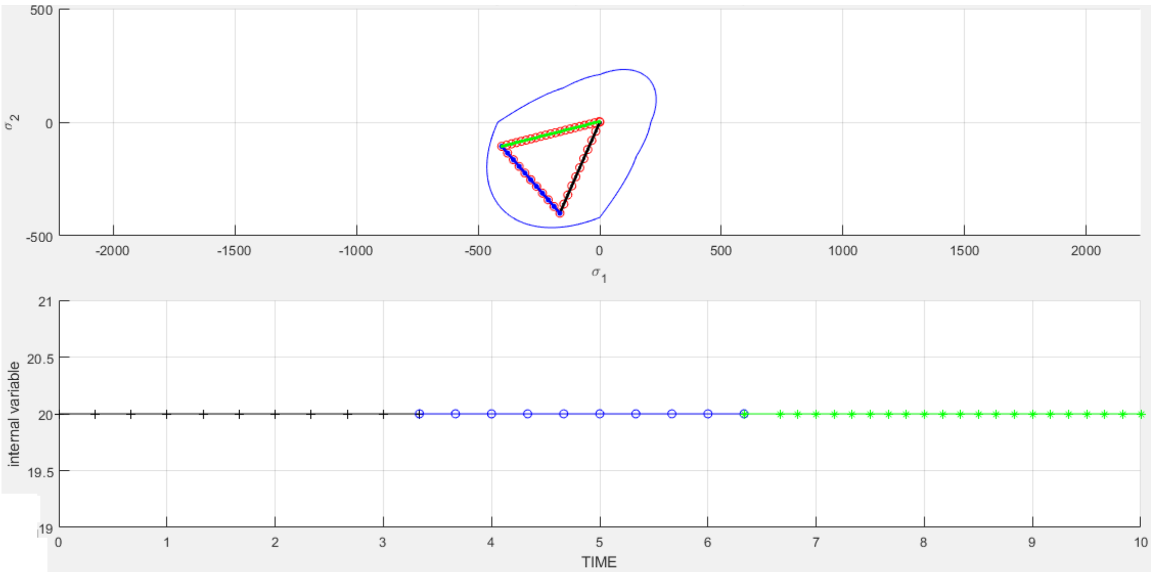

Figure 16. Load path and damage surface for a non-symmetric parameter n=2 and ν=0.3 (top) and internal variable evolution over time (bottom)

The previous example shows the fact that when loads are inside the damage surface, the internal variable doesn't vary, no matter what arbitrary path is chosen. When the damage surface is reached:

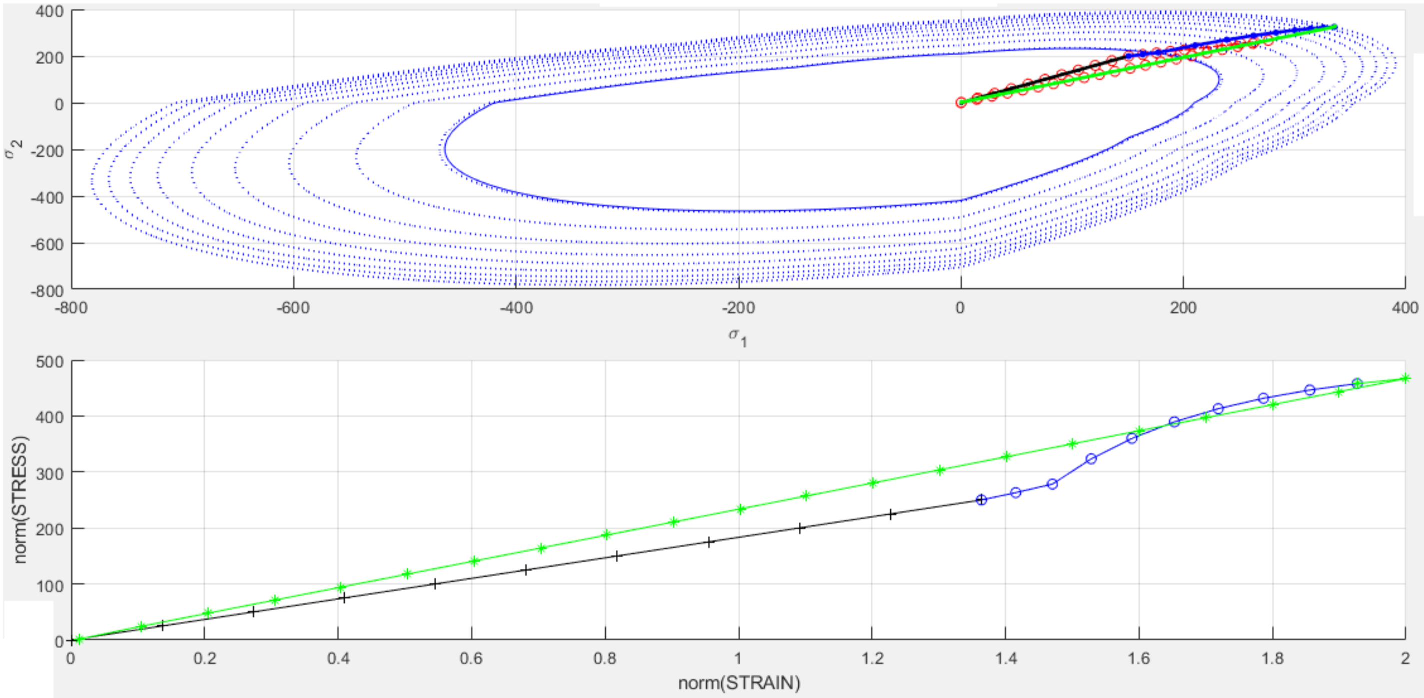

Figure 17. Load path and damage surface for a non-symmetric parameter n=2 and ν=0.3 (top) and relation between stresses and strains(bottom).

Again, it is clearly seen that hardening makes the damage surface to enlarge, and stresses behave exponentially when those loads are beyond yield stress. Also the non-symmetry of the problem is observed.

All the previous part covers the Damage model for a rate independent problems. Now, we will focus on rate dependent model.

Rate Dependent

Rate dependent Damage model can be easily implemented starting with the previous code by Perzyna's Regularization which consist in changing the evolution law of the internal variable. From the following figure we see the new behaviour.

Compared with the rate independent case, now points outside the surface are allowed. As it can be see in *Figure 18.* there are points which are outside the damage surface at that time step. This is because our system has η=0.3

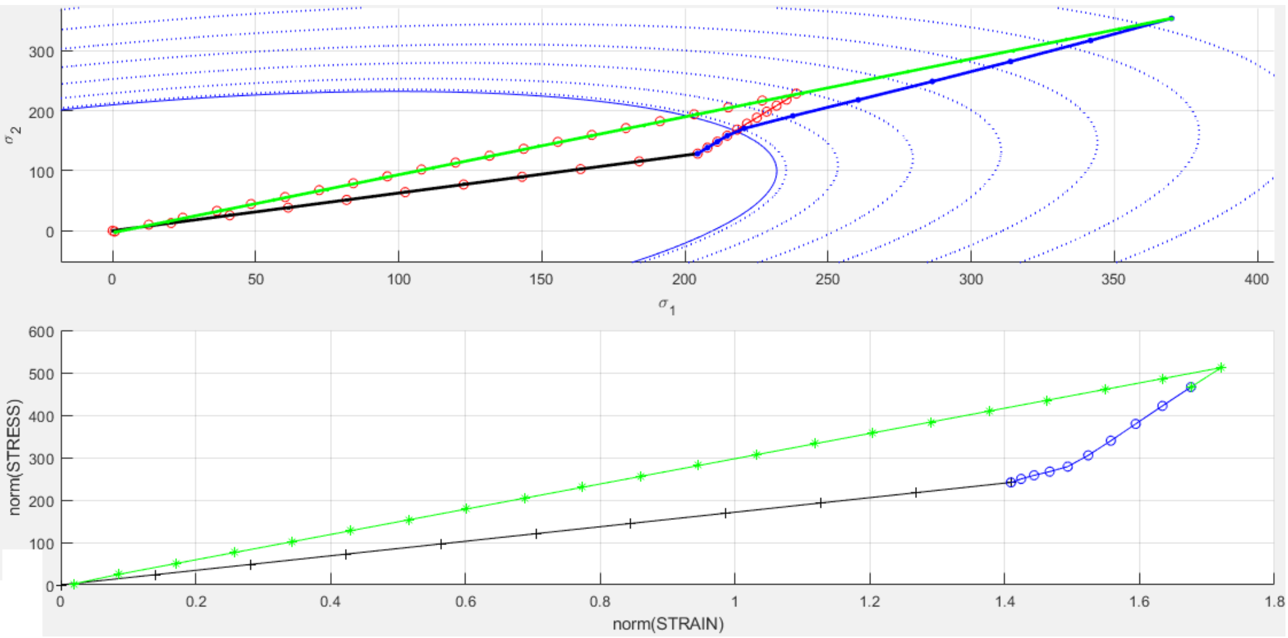

Figure 18. Load path and damage surface for a symmetric system and ν=0.3 (top) and relation between stresses and strains(bottom).

If we load our system uniaxially by applying a load σ=200, then σ=300 and a final load remains at σ=300. For different values of viscosity we have:

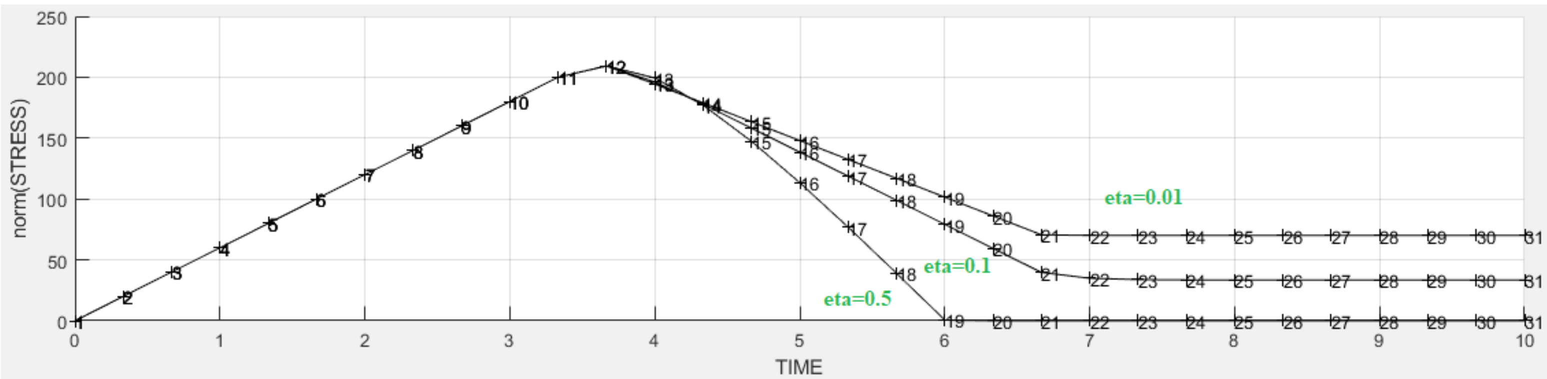

Figure 19. Evolution of the stresses during time for a system with linear softening.

Previous figure shows a material with linear softening behaviour, with softening parameter H=-1. As it can be seen, strains evolve differently depending on the viscosity. During elastic loading, all systems have the same behaviour, however, during damage process, systems evolve differently.

Conclusions

The previous work proved that the implementation of the Damage Model had been done properly since we have been able to reproduce the expected behaviour of the model. Although in some cases the examples chosen where not the most generic ones, this doesn't translates in a lost in generalization. The only reason for which specific examples had been used (uniaxial loading for instance) is because its behaviour is intuitive and allows to find errors in case there are.

Intuitive variables such as stresses and strains reproduced the behaviour of a system which presents damage. Regarding other variables (internal, hardening, damage variable), it has been proved that our implementation fulfilled the expected behaviour and restrictions imposed by thermodynamic laws. With that, all objectives stated at the beginning have been accomplished during the realization of this work.

Annexes

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

                                .
                                .
                                .
%*   --------->    fload=1 : damage (compute algorithmic constitutive tensor)
fload=0;
 r0 = sigma_u/sqrt(E);
 zero_q=1.d-3*r0;
 q_infinit=1.8*r0; %Variable which controls exponential hardening/softening
if(rtrial > r_n)
    %*   Loading

    fload=1;
    delta_r=rtrial-r_n;
    r_n1= rtrial  ;
    if hard_type == 0
        %  Linear
        q_n1= q_n+ H*delta_r;
    else
        A=5;  %exponential
        q_n1=q_n -(q_infinit-r0)*( exp(A*(1-r_n1/r0))-exp(A*(1-r_n/r0)));
    end

    if(q_n1<zero_q)
        q_n1=zero_q;
    end
```

This piece of code controls the hardening. In this case, exponential hardening had to be implemented. What it had to be done was to implement the formula for the hardening variable, introducing a new variable called **q_infinit**

Regarding the function which plots the damage surface:

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

.

.

.

```
elseif MDtype==2
        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

        m11(i)=m1(i)*(m1(i)>0);
        m22(i)=m2(i)*(m2(i)>0);
        radio(i)= q/sqrt([m11(i) m22(i) 0 nu*(m11(i)+m22(i))]*ce_inv*[m1(i) m2(i) 0 ...
            nu*(m1(i)+m2(i))]');
        m1=cos(tetha);                       %*
        m2=sin(tetha);
        s1(i)=radio(i)*m1(i);
        s2(i)=radio(i)*m2(i);

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

This is for only tension.

For non-symmetric:

```
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

        m11(i)=m1(i)*(m1(i)>0);
        m22(i)=m2(i)*(m2(i)>0);


        theta1=m11(i)+m22(i)+nu*(m11(i)+m22(i));
        theta2=abs(m1(i))+abs(m2(i))+abs(nu*(m1(i)+m2(i)));
        thetaAux=theta1/theta2;
        naux=2;  %cal canviar-ho també a la funció Modelo_de_danol

        radio(i)= q/((thetaAux+(1-thetaAux)/naux)*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);
```

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


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

elseif (MDtype==2)  %* Only tension
    sigmaAux =ce*eps_n1';
    sigmames=(sigmaAux+abs(sigmaAux))/2;
    sigmabarra=sigmames;
    rtrial= sqrt(sigmabarra'*eps_n1');
elseif (MDtype==3)  %*Non-symmetric
    sigmaAux =ce*eps_n1';
    sigmames=(sigmaAux+abs(sigmaAux))/2;
    theta2=abs(sigmaAux(1))+abs(sigmaAux(2))+abs(sigmaAux(3))+abs(sigmaAux(4));
    theta1=sigmames(1)+sigmames(2)+sigmames(3)+sigmames(4);
    thetaAux=theta1/theta2;
    naux=2;
    rtrial=(thetaAux+(1-thetaAux)/naux)*sqrt(sigmaAux'*inv(ce)*sigmaAux);
end
%*************************************************************************
return
```

For viscous model the same code has been duplicated to avoid errors in the previous one. Then some changes had been done. New variables have been introduced given that we need values from previous steps in order to use theta-methods to integrate the evolution equations.

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

hvar_n1 = hvar_n;
r_n     = hvar_n(5);
q_n     = hvar_n(6);
E       = Eprop(1);
nu      = Eprop(2);
H       = Eprop(3);
sigma_u = Eprop(4);
hard_type = Eprop(5) ;
viscpr = Eprop(6);
eta = Eprop(7);       %new variables introduced
alpha = Eprop(8);

     .
     .
     .

[rtrialAnterior] = Modelos_de_dano1(MDtype,ce,eps_n,n);
[rtrial] = Modelos_de_dano1(MDtype,ce,eps_n1,n);           %Theta methods
rtrial_n_alpha = rtrialAnterior*(1-alpha)+rtrial*alpha;
%*****************************************************************************************
```

```
%*****************************************************************************************
    .
    .
    .

if (rtrial_n_alpha > r_n)
    %*   Loading
    fload = 1;
    delta_r=rtrial_n_alpha-r_n;
    r_n1 = (((eta - delta_t*(1-alpha))*r_n)/(eta + alpha*delta_t)) + ((delta_t*rtrial_n_alpha)/(eta + alpha*delta_t));
if hard_type == 0
       %  Linear
    q_n1= q_n+ H*delta_r;
else
   A=5;  %exponential
   q_n1=q_n -(q_infinit-r0)*( exp(A*(1-r_n1/r0))-exp(A*(1-r_n/r0)));
end

if(q_n1<zero_q)
   q_n1=zero_q;
end
if(q_n1<zero_q)
   q_n1=zero_q;
end
```