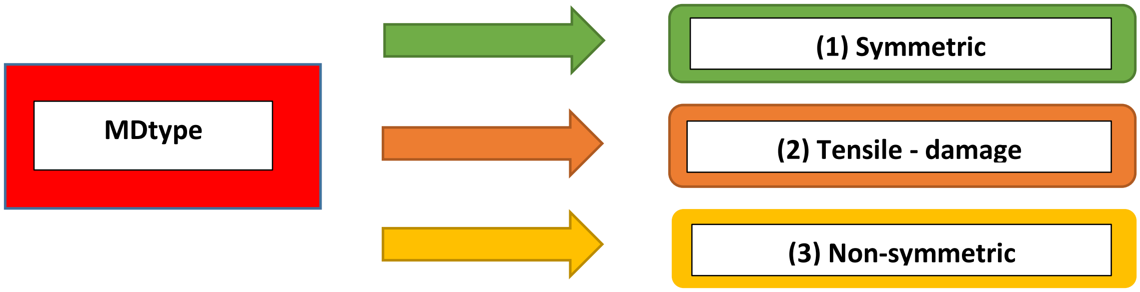
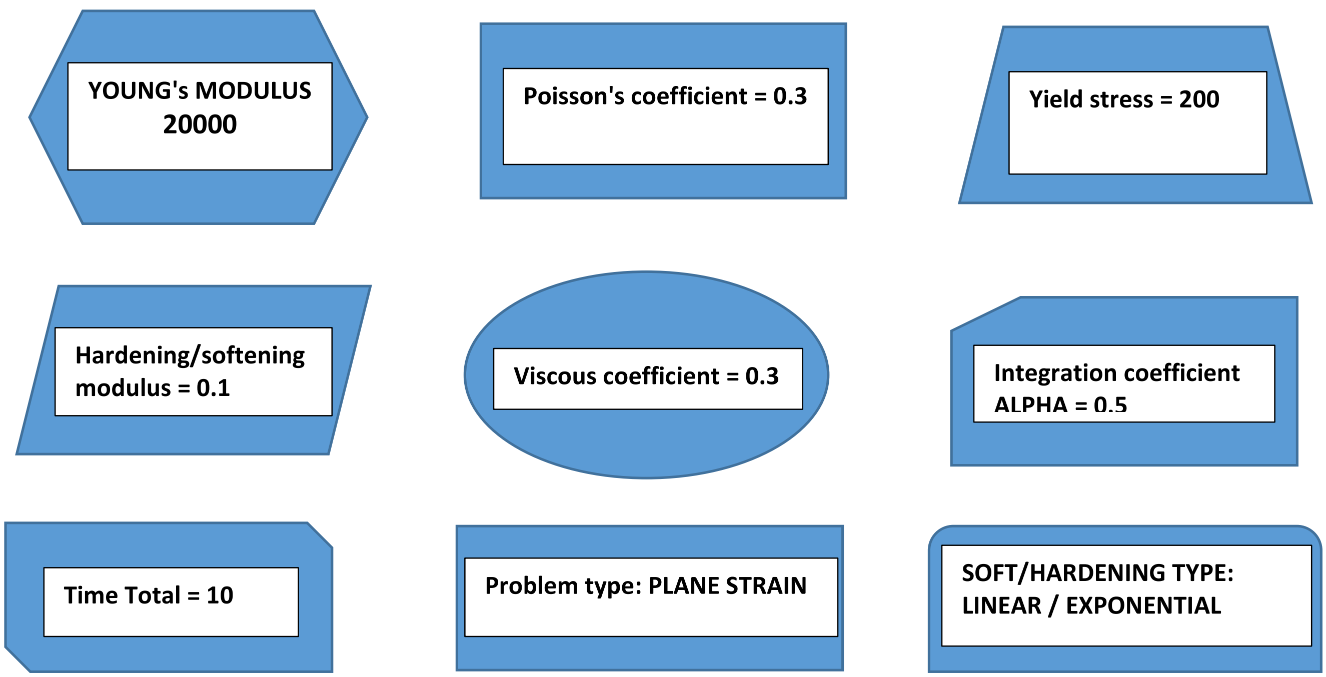
UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH

Computational Solid Mechanics

Seyed MohammadReza Attar Seyedi

APRIL 2017





In this program there are three model types: 1- Symmetric 2- Tensile-damage 3- Non symmetric**.** Fist type is already solved and we should work with those two types.

Mainnoninteractive.m: is the main part of code. There are a lots of variable like Young's Modulus, Poisson’s coefficient, Yield stress, Hardening/softening modulus, viscous coefficient, Integration coefficient ALPHA as I showed in The upper part these variable is very important and changing of each variable can effect in the plot. After running the code we can see our plot and result.

Rmap_dano1.m: integration algorithm for isotope damage model.

Damage_main.m: main calculation part on the other hands for calculating update stress and tensor.

Modouls_de_dano1: damage criterion surface.

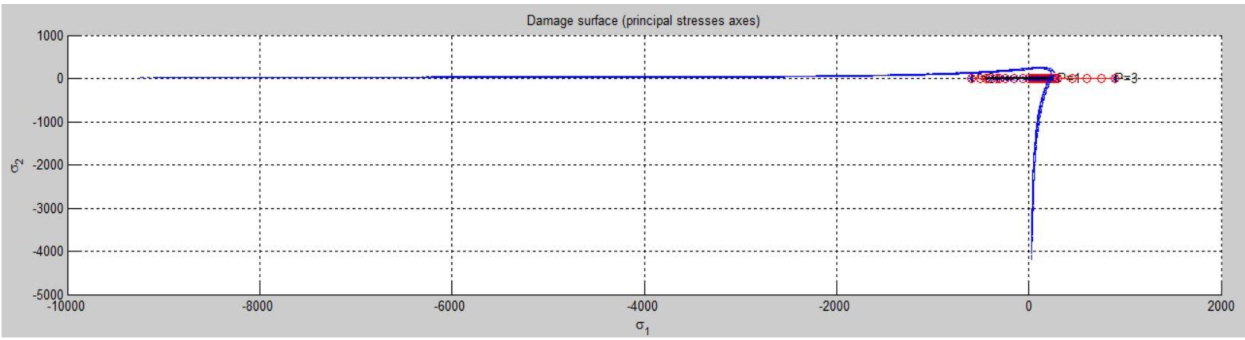
Dibujar Criteriodano1.m: plotting damage surface.
Plotlinsurf.m: plot elastic domain limit and stress pass.
PlotcurvesNEW.m: plot stress and strain.
Tensor_elastico.m: elastic constitutive tensor.
Subplot.m: create axes in tiled positions.
Hold.m: for Holding the graph.

(1) Symmetric

First case is symmetric that already solved and this case is MDtype = 1

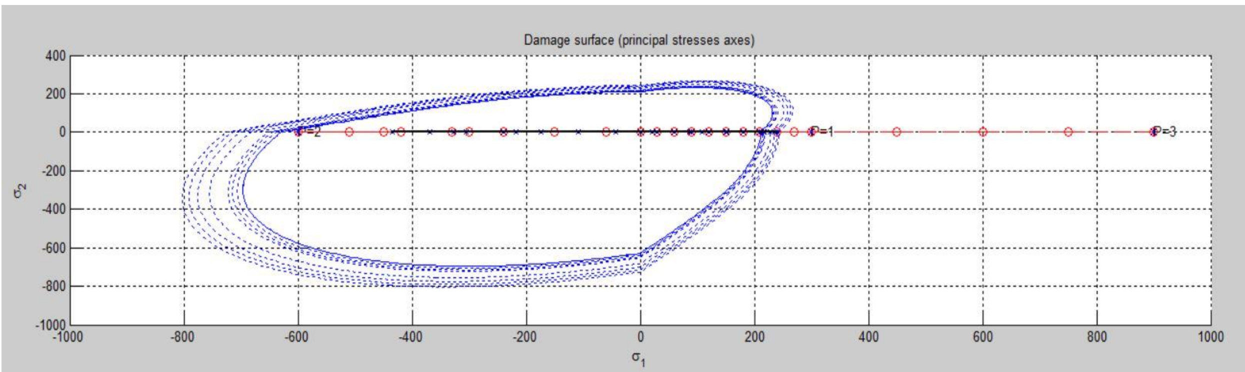
(2) Tensile – damage

Second case is only tension MDtype = 2 that we are writing code for plotting damage surface and damage criterion surface.



(3) Non-symmetric

Non Symmetric is Third one MDtype = 3 we should implement code for plotting part and damage criterion surface.



1

$$\Delta\sigma 1^{(1)} = \alpha \qquad \Delta\sigma 2^{(1)} = 0$$

$$\Delta\sigma 1^{(2)} = -\beta \qquad \Delta\sigma 2^{(2)} = 0$$

$$\Delta\sigma 1^{(3)} = \gamma \qquad \Delta\sigma 2^{(3)} = 0$$

(Uniaxial tensile loading)

(Uniaxial tensile unloading / Compressive loading)

(Uniaxial compressive unloading / tensile loading)

2

$$\Delta\sigma 1^{(1)} = \alpha \qquad \Delta\sigma 2^{(1)} = 0$$

$$\Delta\sigma 1^{(2)} = -\beta \qquad \Delta\sigma 2^{(2)} = -\beta$$

$$\Delta\sigma 1^{(3)} = \gamma \qquad \Delta\sigma 2^{(3)} = \gamma$$

(Uniaxial tensile loading)

(Biaxial tensile unloading / compressive loading)

(Biaxial compressive unloading / tensile loading)

3

$$\Delta\sigma 1^{(1)} = \alpha \qquad \Delta\sigma 2^{(1)} = \alpha$$

$$\Delta\sigma 1^{(2)} = -\beta \qquad \Delta\sigma 2^{(2)} = -\beta$$

$$\Delta\sigma 1^{(3)} = \gamma \qquad \Delta\sigma 2^{(3)} = \gamma$$

(Biaxial tensile loading)

(Biaxial tensile unloading /compressive loading)

Biaxial compressive unloading / tensile loading)

$\alpha = 300$

$\beta = 600$

$\gamma = 900$

In our program our Yield stress is equal 200 and we want to try that if we give value more than Yield stress what we have in our result and we can see the manner of graph after plotting.

When yield stress pass 200 if our $H > 0$ we have hardening or expansion and if our $H < 0$ is softening or contraction.

We are trying to give ($\alpha = 300$, $\beta = 600$, $\gamma = 900$) to see that the behavior of material after using this amount because I tried many case and I distinguish those value and effect better than pervious values and it is clear to realize the result in plot and we can see the behavior of material.

For the first case trying with (Uniaxial tensile loading), (Uniaxial tensile unloading / Compressive loading), (Uniaxial compressive unloading / tensile loading).

Second case (Uniaxial tensile loading), (Biaxial tensile unloading / compressive loading), (Biaxial compressive unloading / tensile loading).

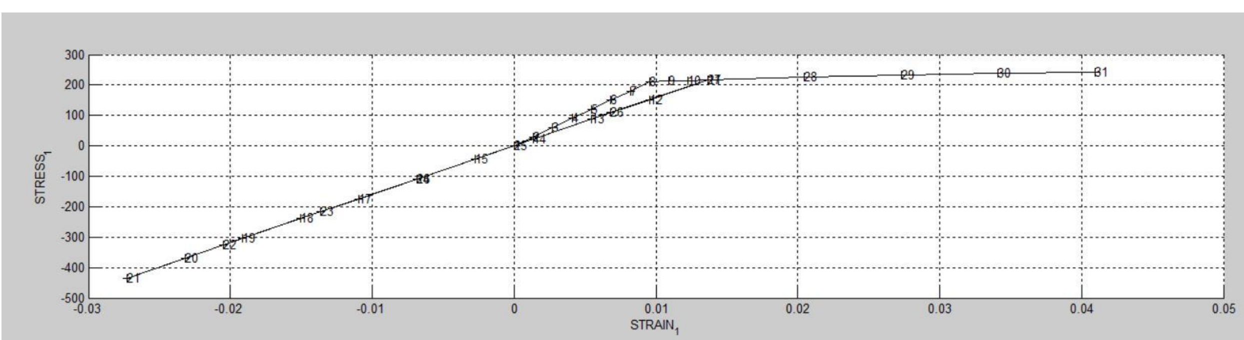
Third case (Biaxial tensile loading), (Biaxial tensile unloading /compressive loading), Biaxial compressive unloading / tensile loading).

After adding those variable in sigma 1, sigma 2, sigma 3 in code we need to plot $\sigma 1$ regards to $\sigma 2$.

The major thing for plotting is our type: Symmetric, only tension and non-symmetric.

We should decide to choose the type of Hard type Linear or Exponential that we need and It is important that we want to calculate $H > 0$ or $H < 0$.

After all we can compute and see the result (Strain 1, Stress 1).



Tensile – damage model

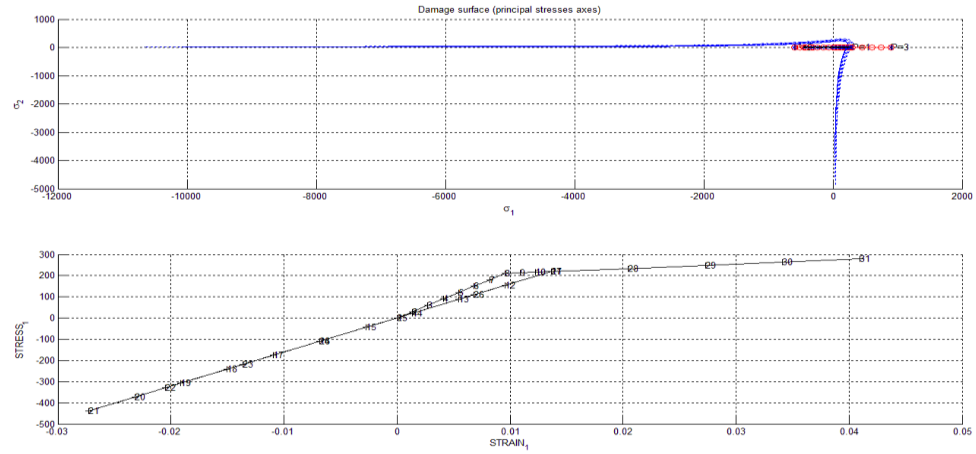


Figure 1: Linear Hardening (1)

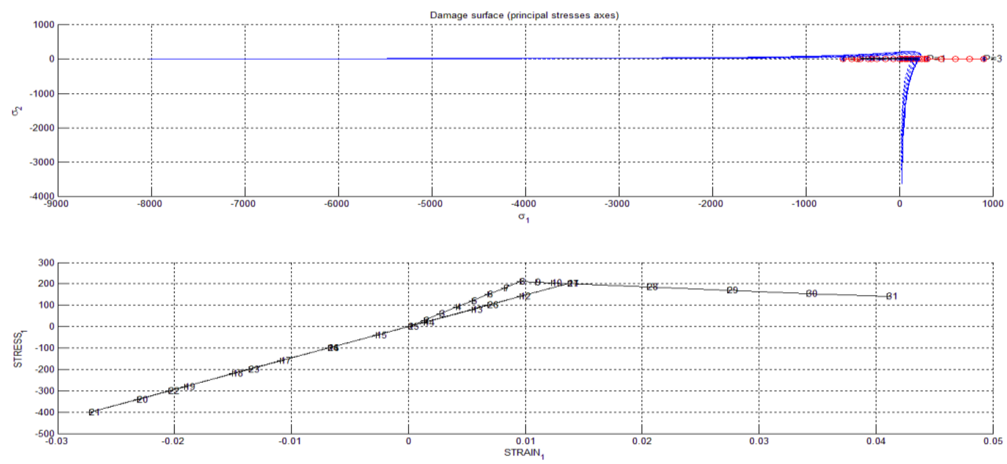


Figure 2: Linear Softening (1)

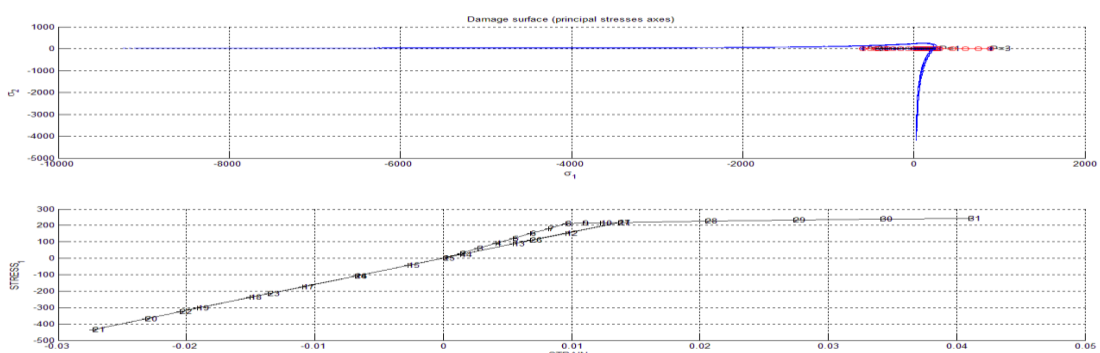


Figure 3: Exponential Hardening (1)

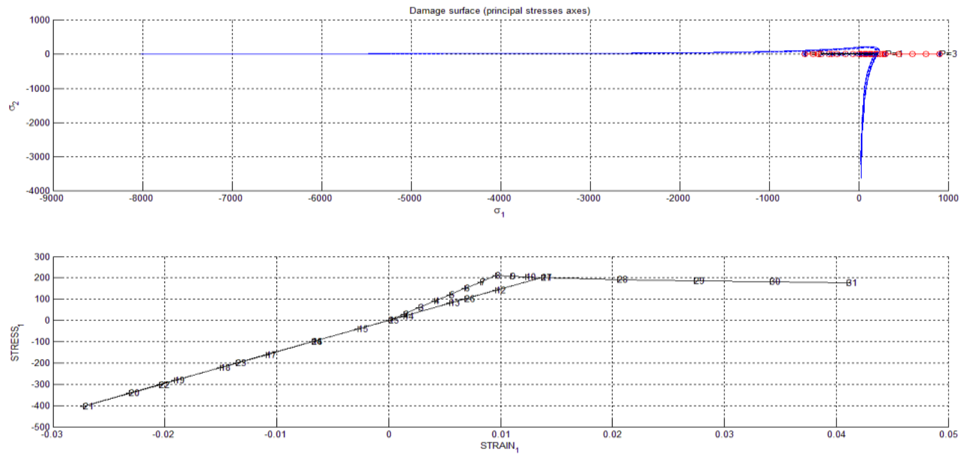


Figure 4: Exponential Softening (1)

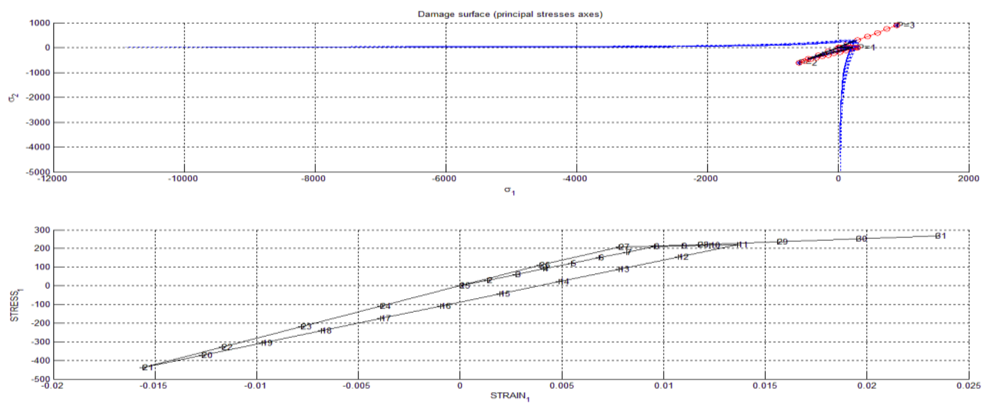


Figure 5: Linear Hardening (2)

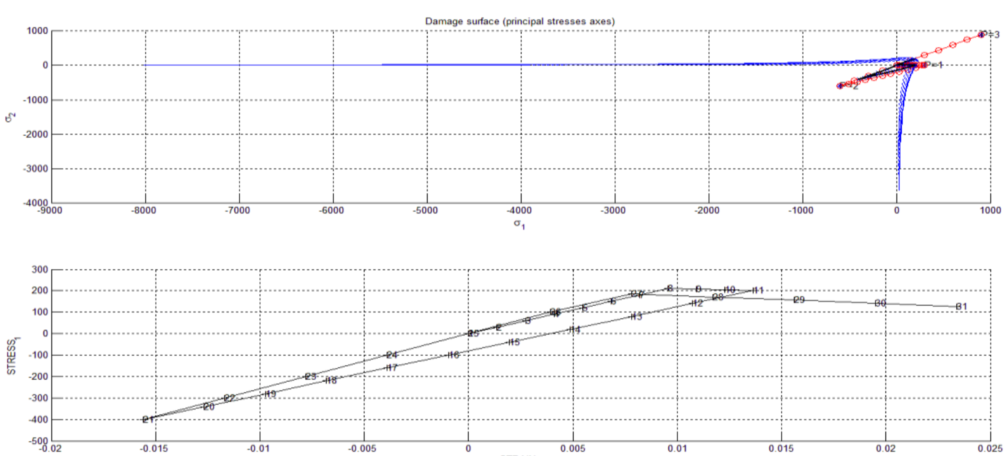


Figure 6: Linear Softening (2)

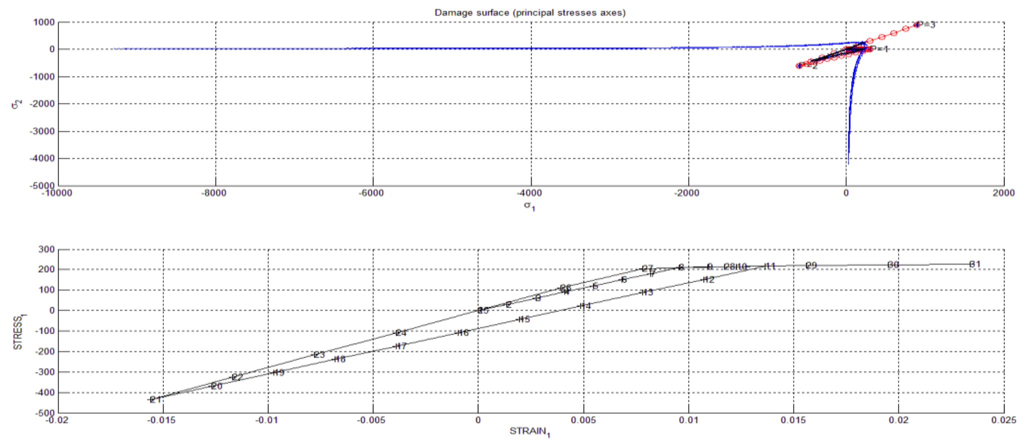


Figure 7: Exponential Hardening (2)

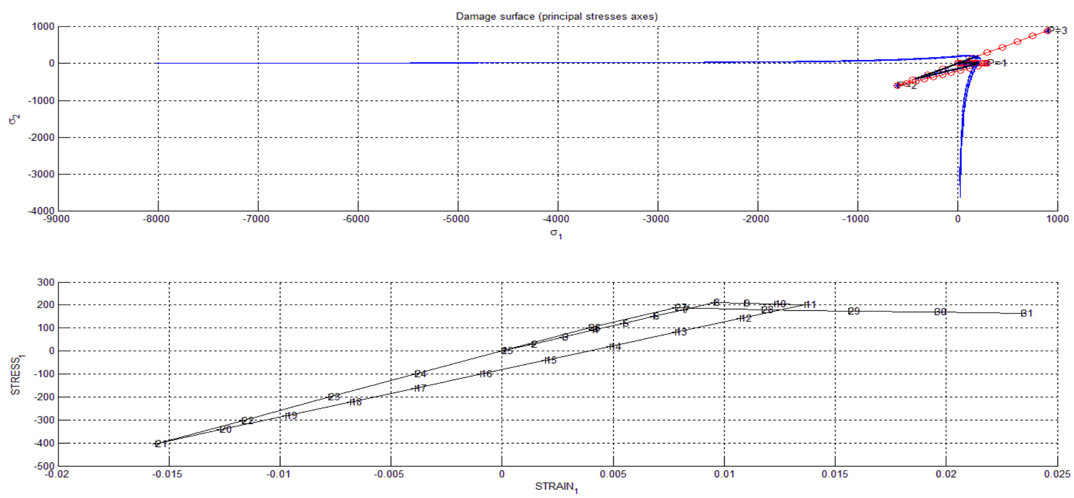


Figure 8: Exponential Softening (2)

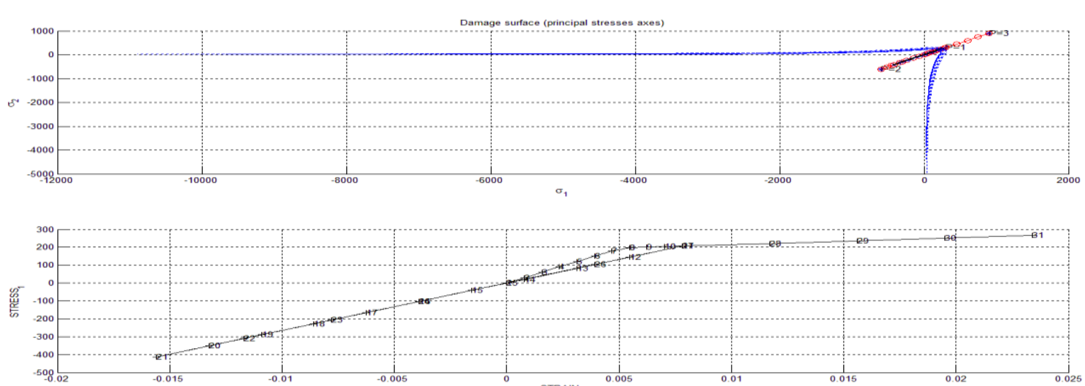


Figure 9: Linear Hardening (3)

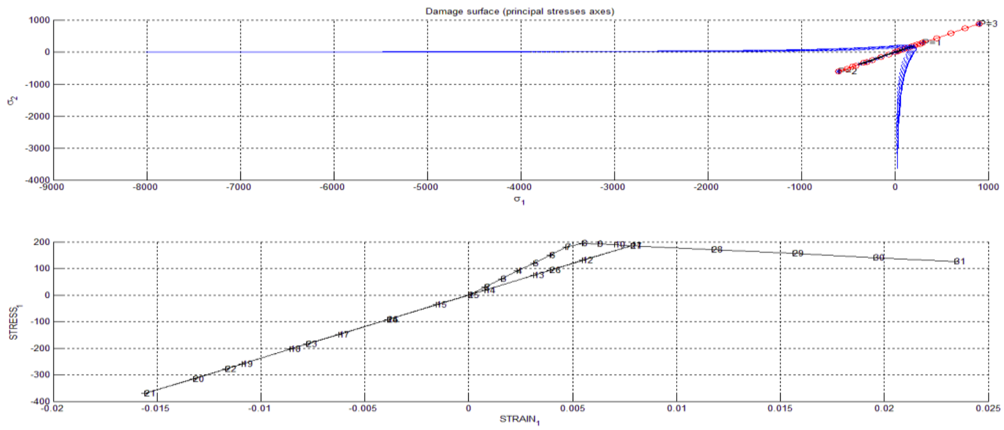


Figure 10: Linear Softening (3)

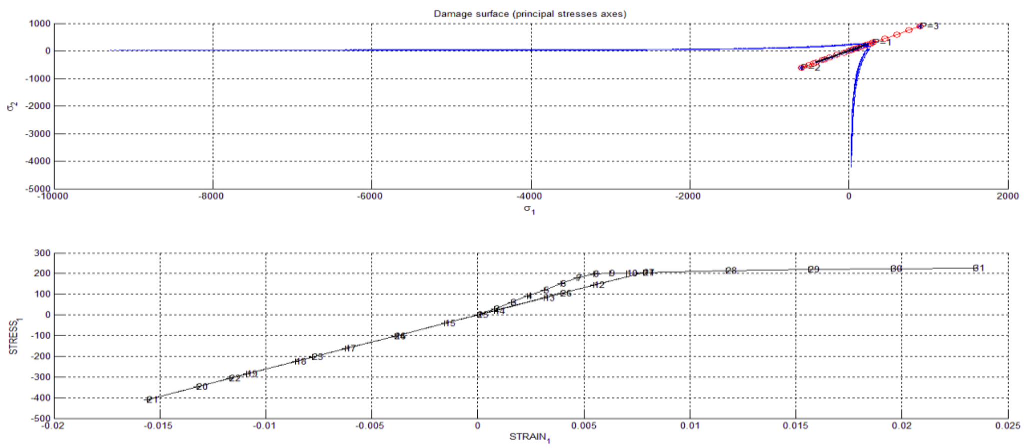


Figure 11: Exponential Hardening (3)

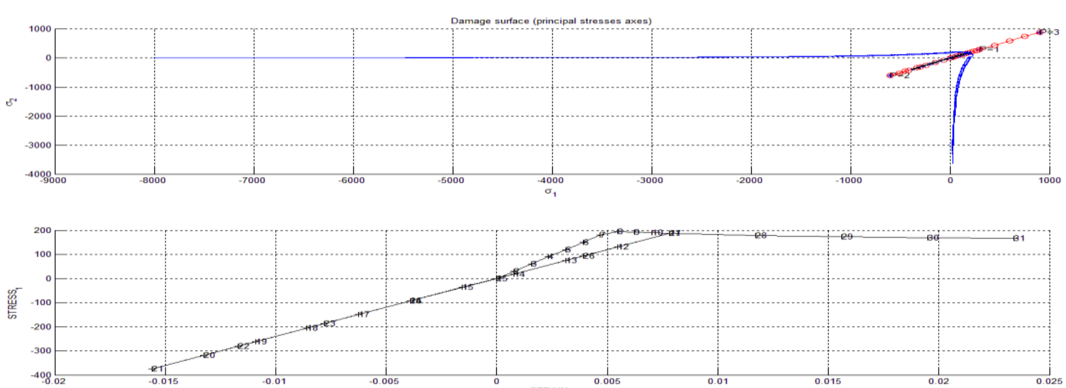


Figure 12: Exponential Softening (3)

Non-Symmetric tension – compression model

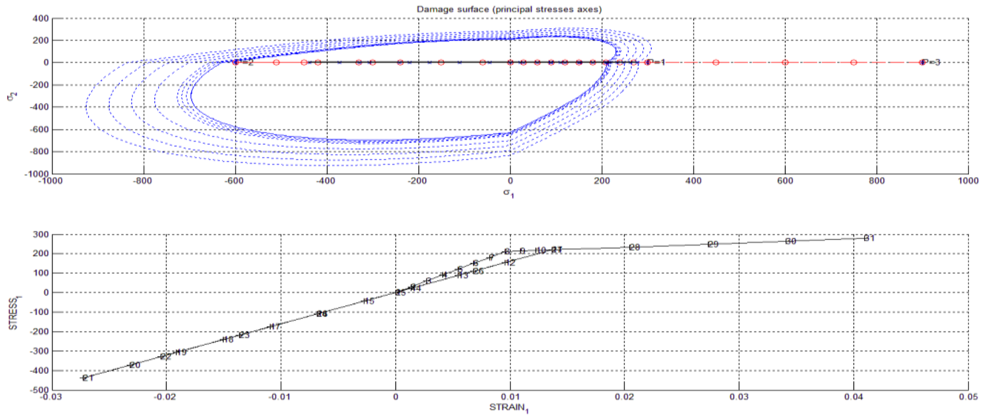


Figure 13: Linear Hardening (1)

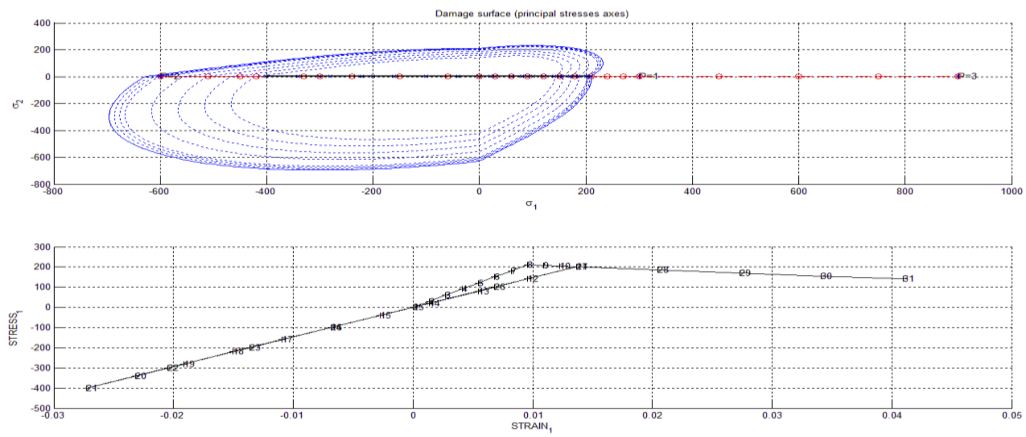


Figure 14: Linear Softening (1)

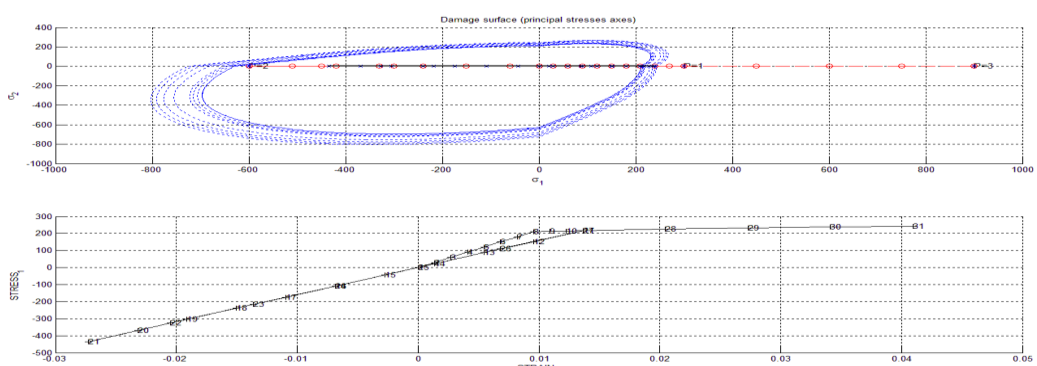


Figure 15: Exponential Hardening (1)

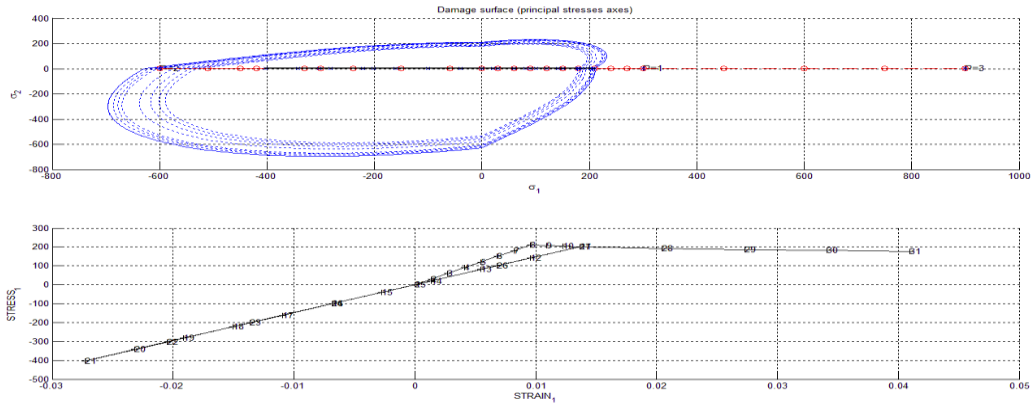


Figure 16: Exponential Softening (1)

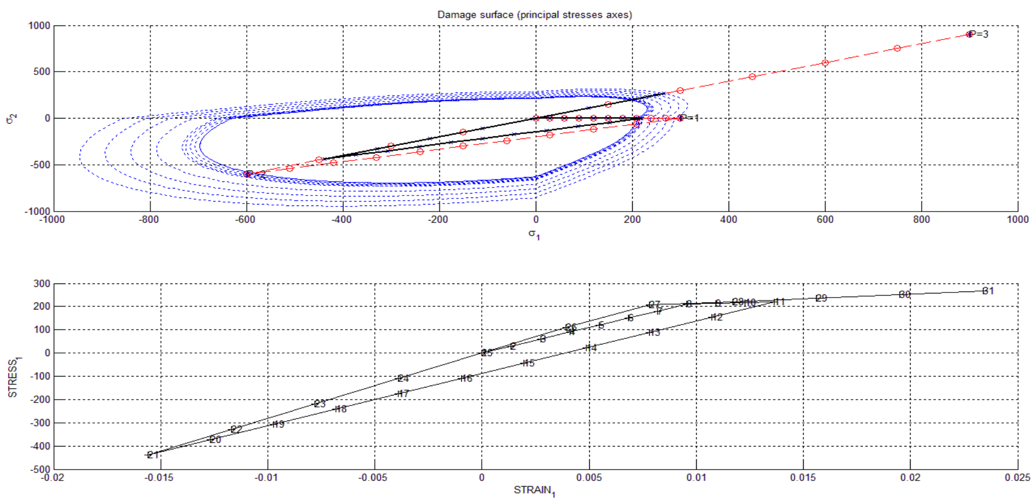


Figure 17: Linear Hardening (2)

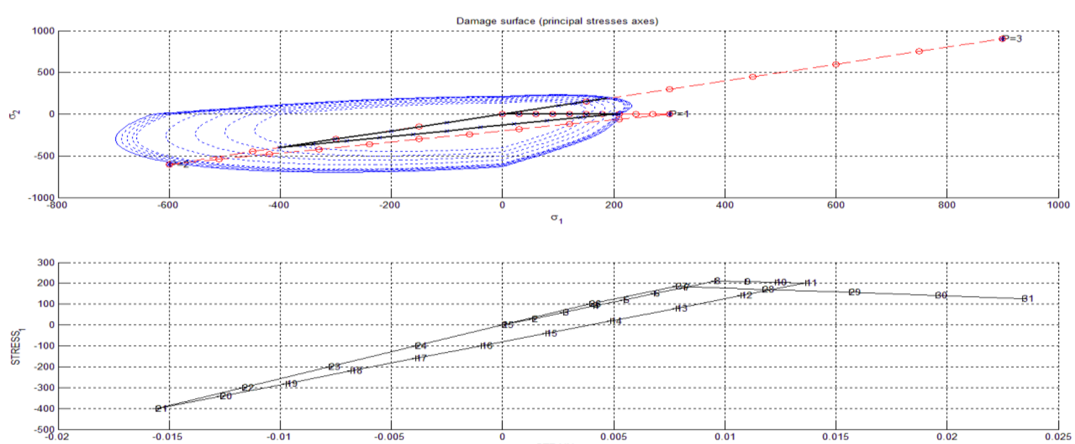


Figure 18: Linear Softening (2)

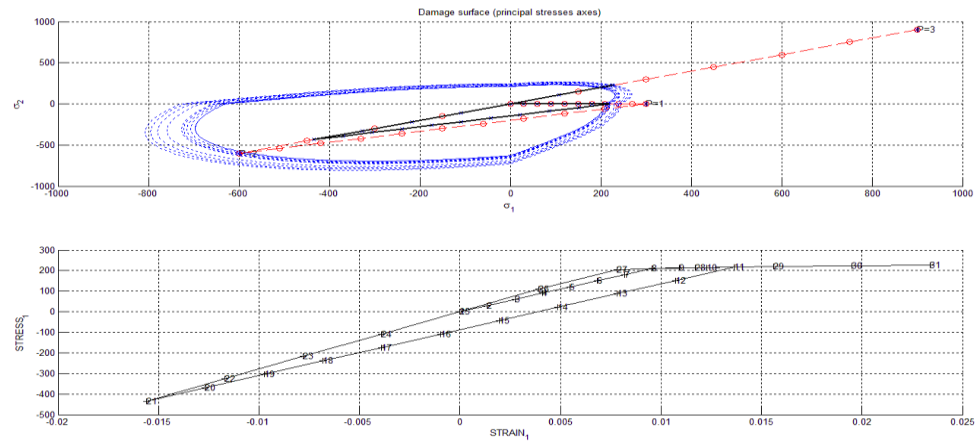


Figure 19: Exponential Hardening (2)

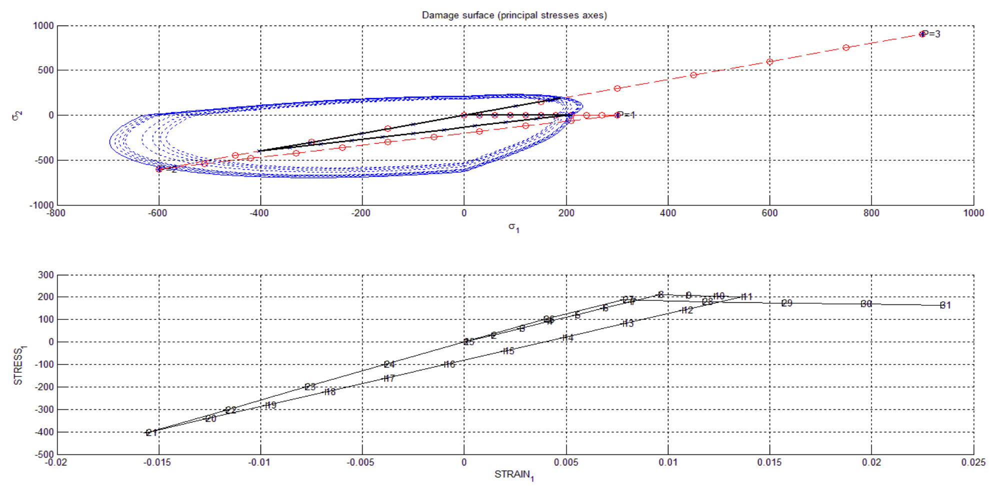


Figure 20: Exponential Softening (2)

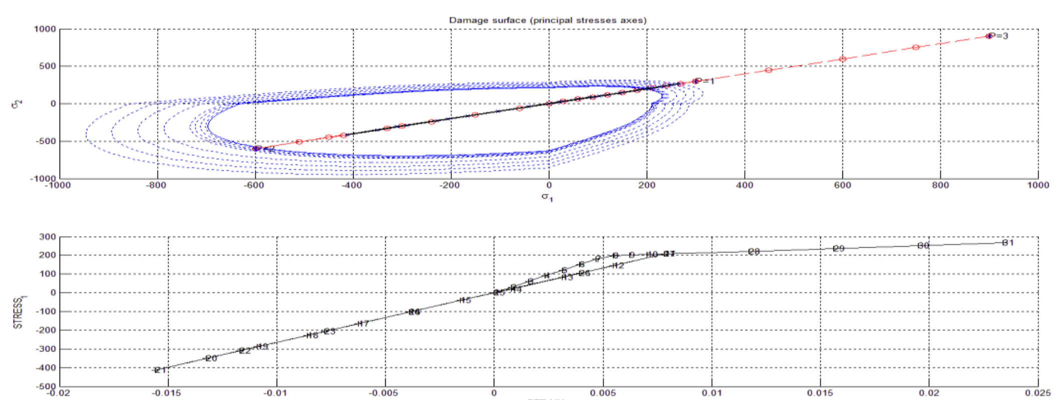


Figure 21: Linear Hardening (3)

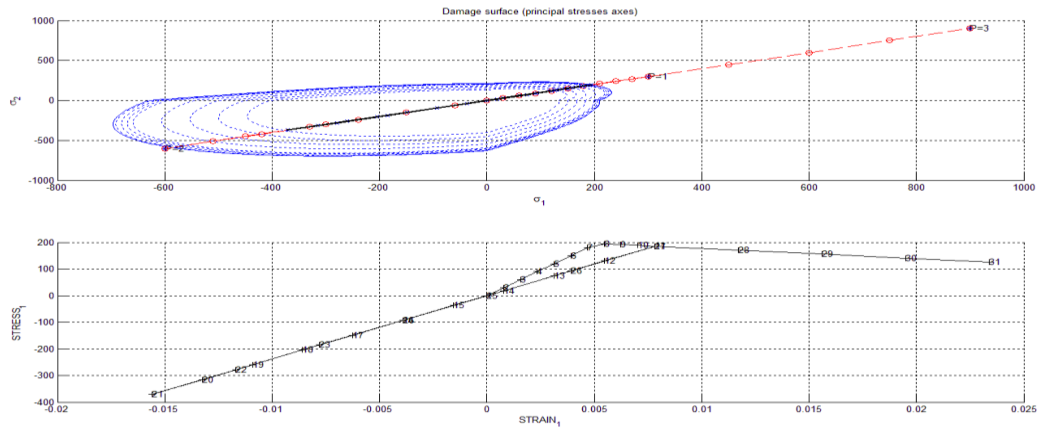


Figure 22: Linear Softening (3)

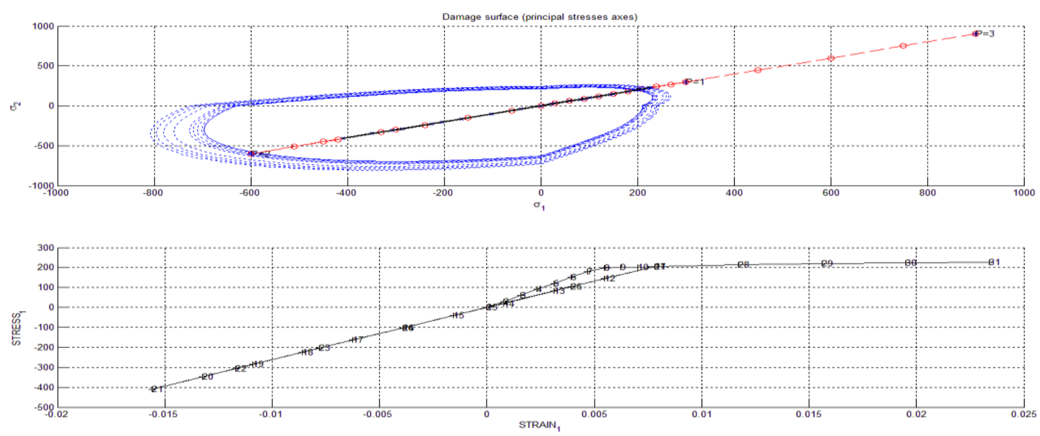


Figure 23: Exponential Hardening (3)

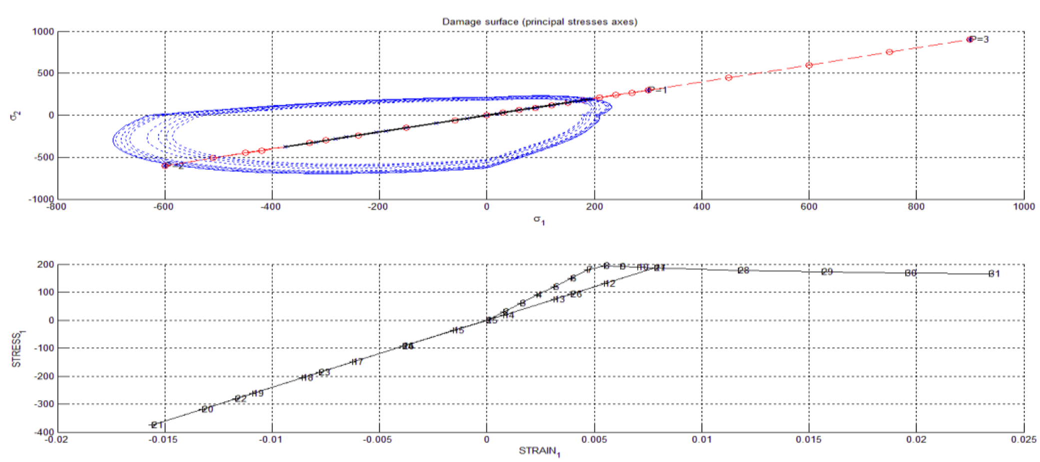
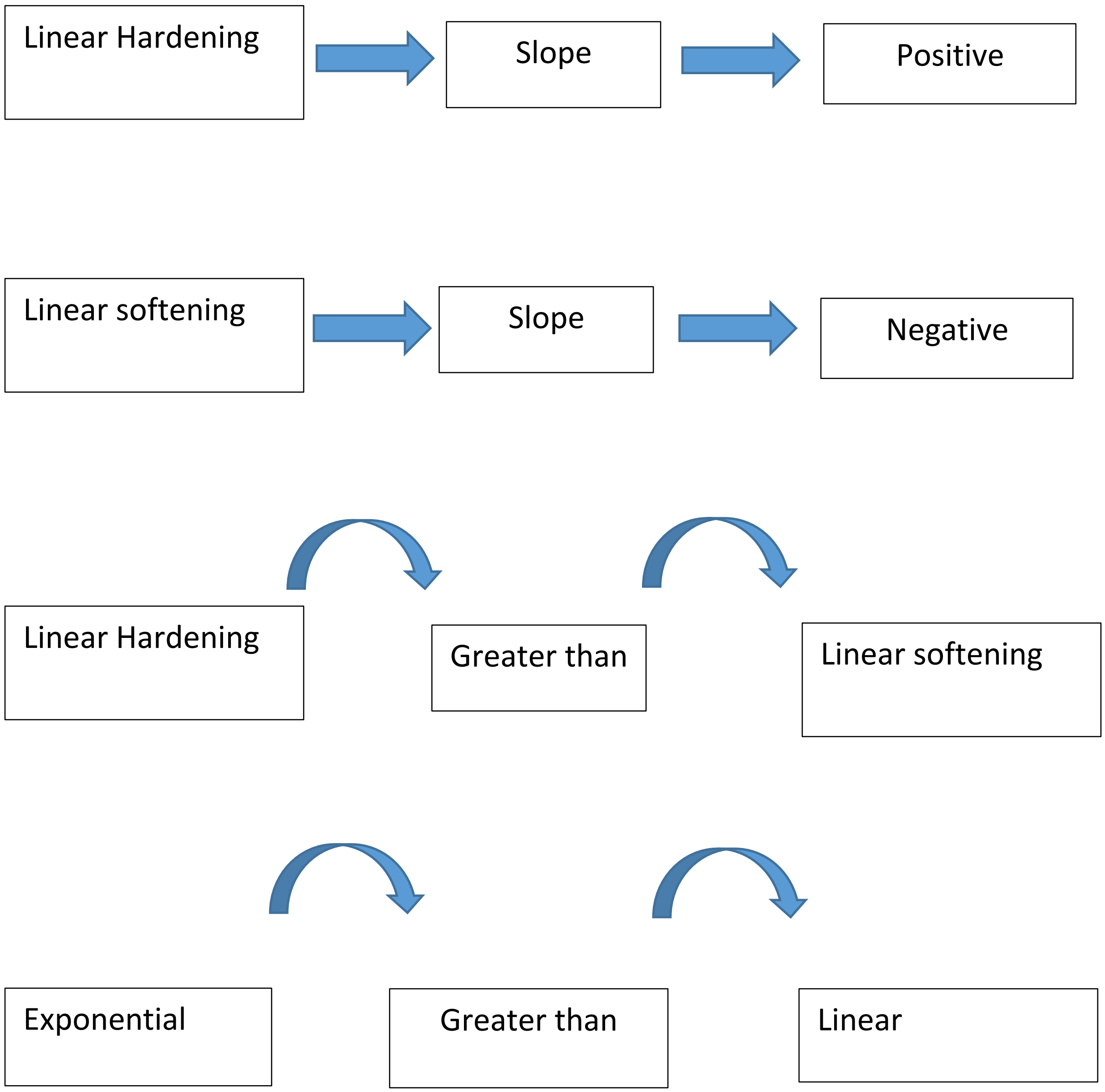


Figure 24: Exponential Softening (3)

As we consider we can realize that:



It is very clear when we see the plot and can distinguish the difference of the only tension and non-symmetric.

When we compare between two condition understanding that linear and exponential behavior the same in the graph in some condition we do not have hardening.

Annex 1 (Dibujar)

```
function hplot = dibujar_criterio_dano1(ce,nu,q,tipo_linea,MDtype,n)
ce_inv=inv(ce);
c11=ce_inv(1,1);
c22=ce_inv(2,2);
c12=ce_inv(1,2);
c21=c12;
c14=ce_inv(1,4);
c24=ce_inv(2,4);
if MDtype==1
    tetha=[0:0.01:2*pi];

    %* RADIUS
    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) ;

    for i=1:Contador
         vec1=[m1(i) m2(i) 0 nu*(m1(i)+m2(i))];
        radio(i)= q /sqrt([vec1]*ce_inv*[m1(i) m2(i) 0 ...
            nu*(m1(i)+m2(i))]');

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

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


elseif MDtype==2

    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) ;

     for i=1:Contador

         vec1=[m1(i) m2(i) 0 nu*(m1(i)+m2(i))];

         vec1m=(vec1+abs(vec1))/2;
```

For only tension MDtype = 2

We use D, m1, m2 as pervious step

We should use McAuley bracket

For this vector then use it for radio (i)

```
       radio(i)= q /sqrt([vec1m]*ce_inv* [vec1]');

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

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

      end

     hplot =plot(s1,s2,tipo_linea);

elseif MDtype==3

      tetha=[0:0.01:2*pi];
         %* RADIUS
   D=size(tetha);
   m1=cos(tetha);
   m2=sin(tetha);
   Contador=D(1,2);
```

After using McAuley bracket for the vector

We should use it for thetaa and thetaa is equal to sum of McAuley bracket of vector divide to sum of abs vector

Then use it for radio (i)

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

     for i=1:Contador

         vec1=[m1(i) m2(i) 0  nu*(m1(i)+m2(i))];


         vec1m=(vec1+abs(vec1))/2;


         thetaa =(sum (vec1m)) / (sum ( abs(vec1)));


  radio(i)= q /sqrt([vec1]*ce_inv*[vec1]')/((thetaa+(1-thetaa)/n));


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

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

end
return
```

Annex 2 (Modouls_de_dano1)

```
if (MDtype==1)        %* Symmetric

  rtrial= sqrt(eps_n1*ce*eps_n1');


elseif (MDtype==2)  %* Only tension

  vecT = eps_n1 *ce;

  vecTm =(vecT+abs(vecT))/2;

  rtrial= sqrt( vecTm * eps_n1') ;

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

   vect = eps_n1 * ce;

   vectm= (vect+abs(vect))/2;

   thetaaa = sum (vectm)/ sum (abs (vect));

   rtrial= (thetaaa+(1-thetaaa)/n)* sqrt(eps_n1*ce*eps_n1');
```

First we create vector

Then use McAuley bracket for vector then use it for rtrial

We have extra theta here and follow pervious step

Annex 3 (Modouls_de_dano1)

```
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

        if H > 0
            q_inf = r0 + (r0-zero_q) ;
            A = (H*r0) / (q_inf- r0) ;

        q_n1 = q_inf-(q_inf-q_n)*exp((abs(A))*(1-(r_n1/r_n)));

        else

        q_min = zero_q ;

        A= (H*r0) / (q_min- r0) ;

        q_n1 = q_min -(q_min-q_n)*exp((abs(A))*(1-(r_n1/r_n)));


        end
```

For H > 0 we should define q_inf (q max), A and then use it for our q_n1

In other way we have q min

And again we should define it and use it in q_n

Annex 4 (Mainnoninteractive)

```
% INPUTS
% ****************

% YOUNG's MODULUS
% ---------------
YOUNG_M = 20000 ;
% Poisson's coefficient
% -----------------------
POISSON = 0.3 ;
% Hardening/softening modulus
% ---------------------------
HARDSOFT_MOD =   0.1 ;
% Yield stress
% ------------
YIELD_STRESS = 200 ;
% Problem type  TP = {'PLANE STRESS','PLANE STRAIN','3D'}
% ------------------------ = 1              =2          =3
% ------------
ntype= 2 ;
% Model    PTC = {'SYMMETRIC','TENSION','NON-SYMMETRIC'} ;
%                      = 1          = 2          = 3
% ---------------------------------------------------
MDtype =2;
% Ratio compression strength / tension strength
% ---------------------------------------------
n = 3 ;
% SOFTENING/HARDENING TYPE
% ------------------------
HARDTYPE = 'EXPONENTIAL' ; %{LINEAR,EXPONENTIAL}
% VISCOUS/INVISCID
% ------------------------
VISCOUS = 'NO' ;
% Viscous coefficient ----
% ------------------------
eta = 0.3 ;
% TimeTotal (initial = 0) ----
% ------------------------
TimeTotal = 10 ;
% Integration coefficient ALPHA
% ------------------------
ALPHA_COEFF = 0.5 ;
% Points ---------------------------
%313% ----------------------------------
nloadstates = 3 ;
SIGMAP = zeros(nloadstates,2) ;
SIGMAP(1,:) =[ 300   0];
SIGMAP(2,:) =[-600  0];
SIGMAP(3,:) =[ 900   00];
```

We define our sigma 1, 2, 3

In this part