J2 Computational plasticity Assignment
Computational Solid Mechanics
Master of Science in Computational Mechanics 2016

Paris Dilip Mulye

May 26, 2016

Rate Independent Perfect Plasticity Model

Material parameters used for this model are, $\sigma_y = 200, E = 200e3, K = 0, H = 0, \nu = 0.3$. The applied undamaged uniaxial (σ_{11}) stress loading is [0, 800, 0, -800, 0, 800] and 20 sub steps were used for each step. Figure 1 and Figure 2 show the result. Note that all results are based on strain driven models. But due to poisson's ratio ν, the strains will not be uniaxial.

It is very interesting to figure out the relationship between σ_{11}^{dev}, σ_{11}, and σ_y. In case of uniaxial loading, let's say $\sigma = [x, 0, 0, 0, 0, 0]$ using Voigt's notation. Then, σ^{dev} becomes, $\sigma = \left[\frac{2x}{3}, \frac{-x}{3}, \frac{-x}{3}, 0, 0, 0\right]$. Since its magnitude (which equals $\sqrt{\frac{2}{3}}x$)is to be compared with $\sqrt{\frac{2}{3}}\sigma_y$, we get, for elastic zone,

$$x \leq \sigma_y \tag{1}$$

Now, to find the relationship between σ^{dev}, observing the ratio of components in σ^{dev}, it can be written as, $\sigma^{dev} = \left[p, \frac{-p}{2}, \frac{-p}{2}, 0, 0, 0\right]$, and its magnitude (which equals $\sqrt{\frac{3}{2}}p$) is to compared with same $\sqrt{\frac{2}{3}}\sigma_y$, we get, for elastic zone,

$$p \leq \frac{2}{3}\sigma_y \tag{2}$$

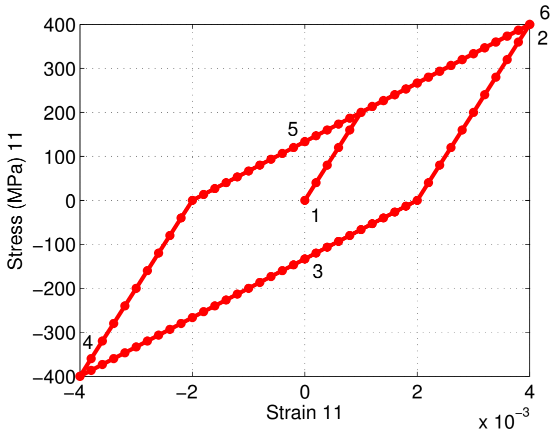

Figure 1: Perfect Plastic Model, σ_{11} vs ϵ_{11}

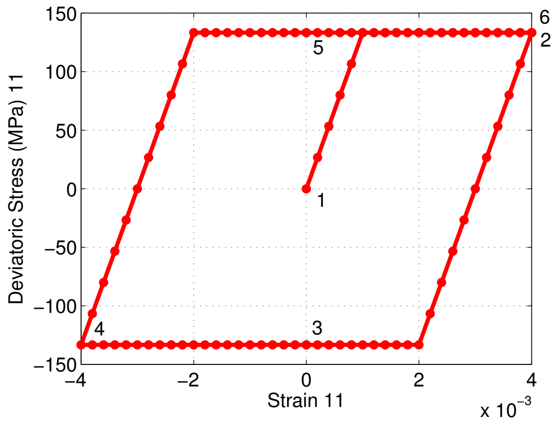

Figure 2: Perfect Plastic Model, σ_{11}^{dev} vs ϵ_{11}

The location at which σ_{11} deviates from elastic slope, has a Y coordinate of 200, which is the yield strength, as expected from (1). Also, the maximum stress that is achieved in the σ_{11}^{dev} was found to be 133.3, which is $\frac{2}{3}\sigma_y$, as expected from (2). Since, all calculations and relationships are done in deviatoric space, deviatoric stress follow the perfect plasticity phenomenon which is also evident from Figure 2.

Rate Independent Linear Isotropic Model

Same material parameters used for this model are, $\sigma_y = 200, E = 200e3, H = 0$. The isotropic hardening modulus K was varied as 50e3, 100e3 and 200e3. The applied undamaged uniaxial (σ_{11}) stress loading is [0, 800, 0, -800, 0, 800] and 20 sub steps were used for each step.

Linear hardening implies that the new slope after yield stress value would be constant, which can be seen from Figure 4 (red plot). Also, isotropic nature of hardening implies that elastic limit in tension and compression should be same. This is observed to be correct based on Y coordinates of point 2 (0.004,204.6) and point P1 (-0.0008,-207.7).

As expected, the higher values of K imply higher value of slope in deviatoric space, in a trivial case of $K = \infty$, the slope would become E.

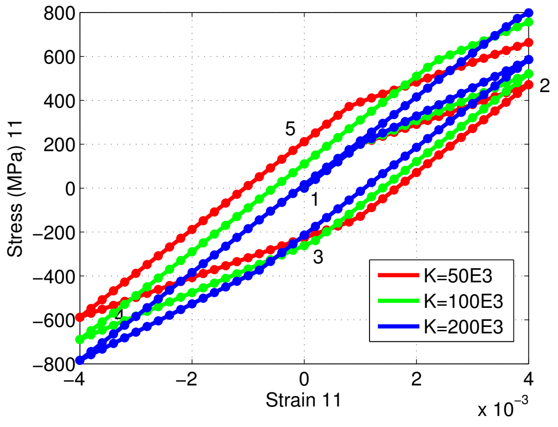

Figure 3: Linear Isotropic Model, σ_{11} vs ϵ_{11}

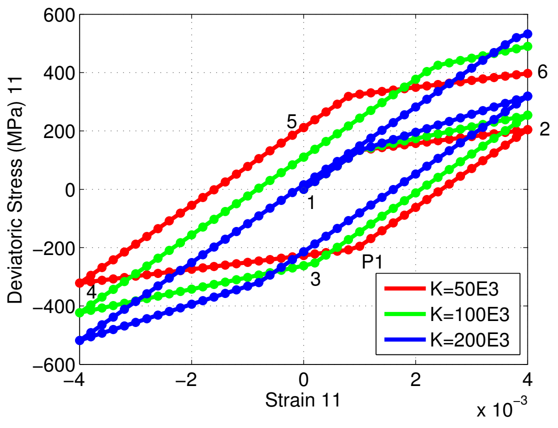

Figure 4: Linear Isotropic Model, σ_{11}^{dev} vs ϵ_{11}

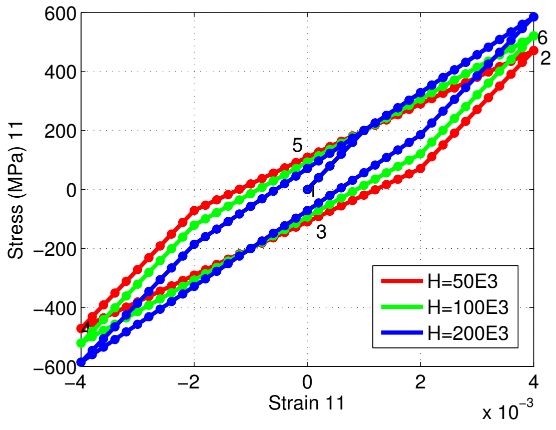

Figure 5: Linear Kinematic Model, σ_{11} vs ϵ_{11}

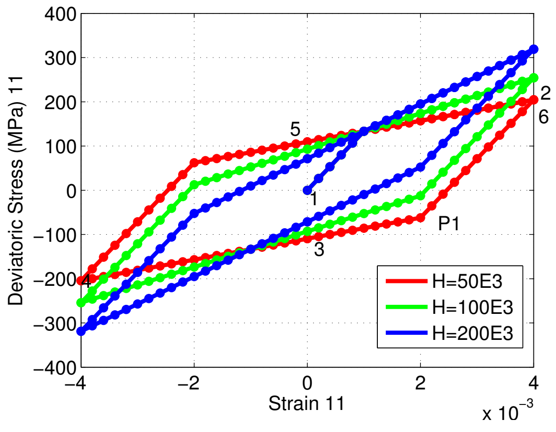

Figure 6: Linear Kinematic Model, σ_{11}^{dev} vs ϵ_{11}

Rate Independent Linear Kinematic Model

Same material parameters used for this model are, $\sigma_y = 200, E = 200e3, K = 0$. The kinematic hardening modulus H was varied as 50e3, 100e3 and 200e3.The applied undamaged uniaxial (σ_{11}) stress loading is [0, 800, 0, -800, 0, 800] and 20 sub steps were used for each step.

Linear hardening implies that the new slope after yield stress value would be constant, which can be seen from Figure 6. Also, kinematic nature of hardening implies that difference of elastic limit in tension and compression should be same and equal to $4/3\sigma_y$ (266.67). This is observed to be correct based on Y coordinates of point 2 (0.004,204.6) and point P1 (0.002,-62.1). It is expected, that as hardening modulus increases, the slope after yield would increase. A trivial case, where H=0 is perfectly plastic.

Rate Independent Nonlinear Isotropic Model

Same material parameters used for this model are, $\sigma_y = 200, E = 200e3, K = 0, H = 0, \sigma_\infty = 300, \delta = 1e3$. Since, only saturation law is to be modeled, $\Pi'(\xi) = (\sigma_\infty - \sigma_y)(1 - e^{-\delta\xi})$ without the added $K\xi$ was used. The applied uniaxial undamaged stress loading is σ_{11} = [0, 600, 0, -600, 0, 600] and 20 sub steps were used for each step.

The deviatoric stress approaches σ_∞ (300*(2/3)=200) nonlinearly when loaded in compression or tension (Figure 8). Also, once reached, it would behave as a perfect plastic material as if $\sigma_y = \sigma_\infty$. Also, it is expected that difference between Y coordinate of point 2 (183.7) and σ_∞ should be same as difference between point P1 (-184.8) and -σ_∞, since σ_∞ is the total margin for the yield surface to expand.

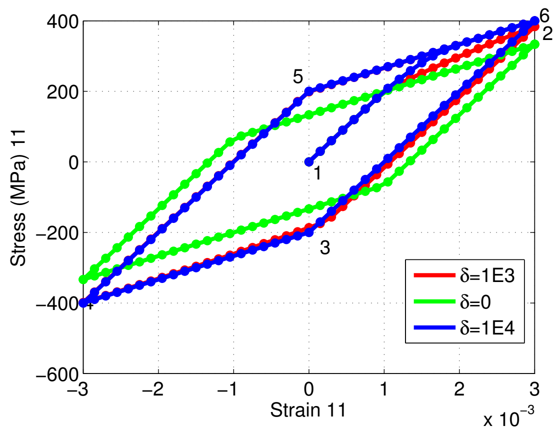

Figure 7: Nonlinear Isotropic Model, σ_{11} vs ϵ_{11}

To study variation with parameter δ, 3 simulations with δ=0, 1e3 and 1e4 were performed. A higher value would imply that σ_∞ is reached at a faster rate (Figure 8). A trivial case, $\delta = 0$, would imply, that $\Pi'(\xi) = 0$, which implies no isotropic hardening. Thus, the curve resembles perfect plasticity.

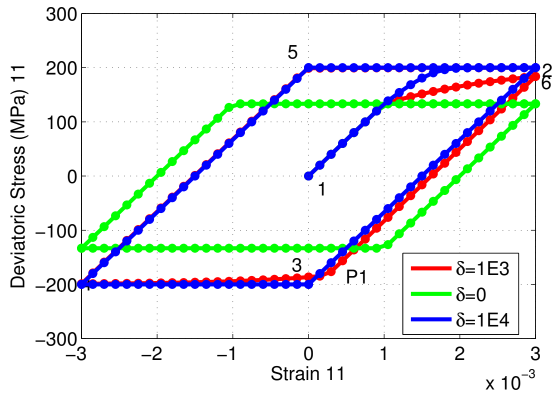

Figure 8: Nonlinear Isotropic Model, σ_{11}^{dev} vs ϵ_{11}

Rate Independent Nonlinear Isotropic, Linear Kinematic Model

The parameters used for this model are, $\sigma_y = 200, E = 200e3, K = 0, H = 50e3, \sigma_\infty = 300, \delta = 1e3$. The applied undamaged stress loading is [0, 600, 0, -600, 0, 600] and 20 sub steps were used for each step with a convergence tolerance of 1e-7 was used for Newton-Raphson method.

As loading increases, this model would approach linear kinematic hardening model as if $\sigma_y = \sigma_\infty$ which can be seen from Figure 9 and Figure 10.

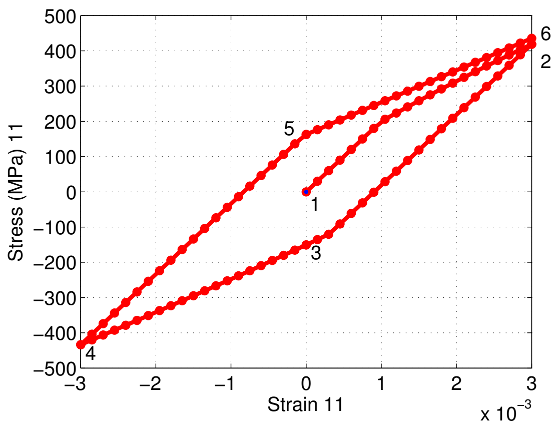

Figure 9: Nonlinear Isotropic Linear Kinematic Model, σ_{11} vs ϵ_{11}

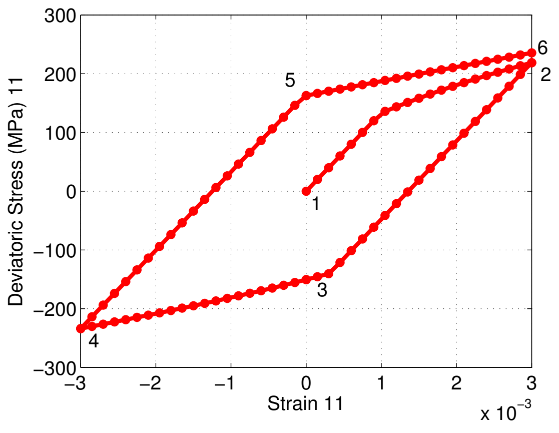

Figure 10: Nonlinear Isotropic Linear Kinematic Model, σ_{11}^{dev} vs ϵ_{11}

Rate Dependent Perfect Plasticity Model

The parameters used for this model are, $\sigma_y = 200, E = 200e3, K = 0, H = 0$. three simulations were run for η= 0, 1e4 and 3e4. The applied undamaged stress loading is [0, 600, 0, -600, 0, 600] time corresponding to these values is [0 1 2 3 4 5] and 20 sub steps were used for each step making dt=0.05.

The perfect plasticity rate dependent model (Figure 13) allow deviatoric stress to exceed $(2/3)\sigma_y$ by an amount which is determined by strain rate and viscosity parameter η. The dependence of η indicates (Figure 11 to 14) that at very low η, rate dependent model becomes equivalent to rate independent model, where . As η increases, the stagnant value of stress increases.

Another striking difference between this model and rate independent nonlinear isotropic hardening, is that even though they both exhibit nonlinear hardening, in case of rate independent model, the non linear effect gets over once, the stress reaches

σ_∞. But in case of this model, the nonlinearity is shown every time, the stress exceeds σ_y.

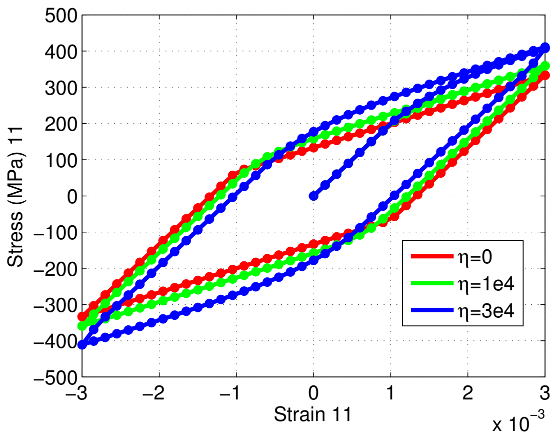

Figure 11: Rate Dependent Perfect Plasticity Model, σ_{11} vs ϵ_{11}

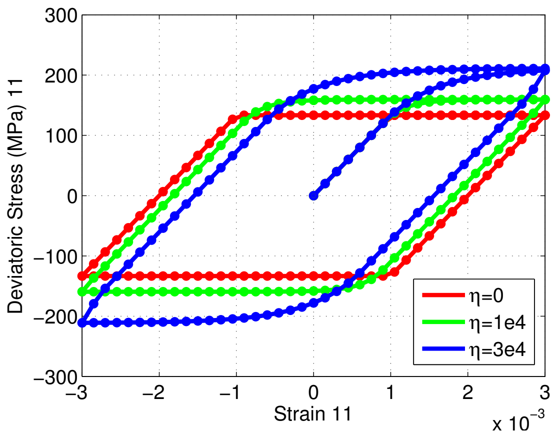

Figure 12: Rate Dependent Perfect Plasticity Model, σ_{11} vs Time

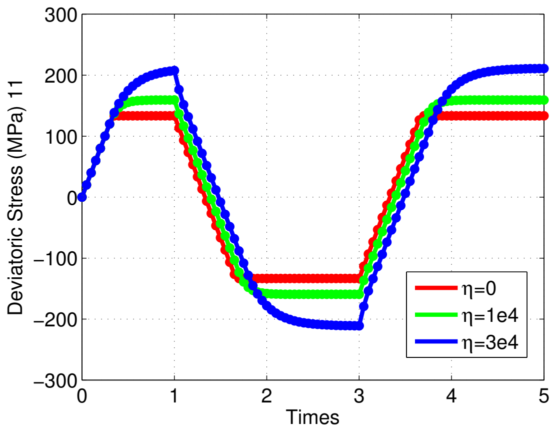

Figure 13: Rate Dependent Perfect Plasticity Model, σ_{11}^{dev} vs ϵ_{11}

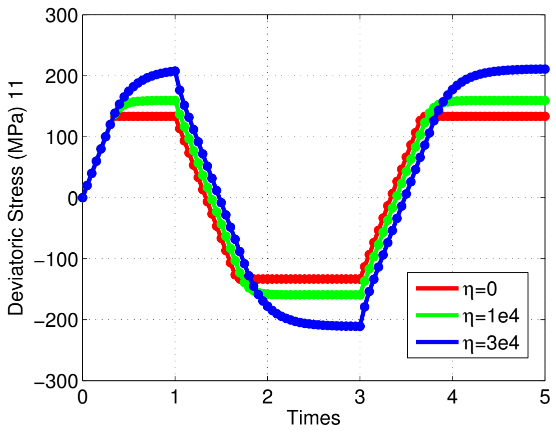

Figure 14: Rate Dependent Perfect Plasticity Model, σ_{11}^{dev} vs Time

The effect of changing strain rate was observed with variation in dt= 0.025, 1 and 5. This was done while keeping $\eta = 1e4$. It is seen that the as strain rate decreases (dt increases), the curve approaches rate independent perfect plasticity model (Figure 15 and Figure 16)

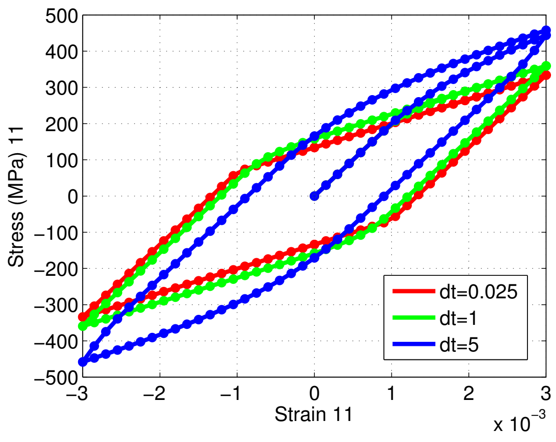

Figure 15: Rate Dependent Perfect Plasticity Model, σ_{11} vs ϵ_{11}

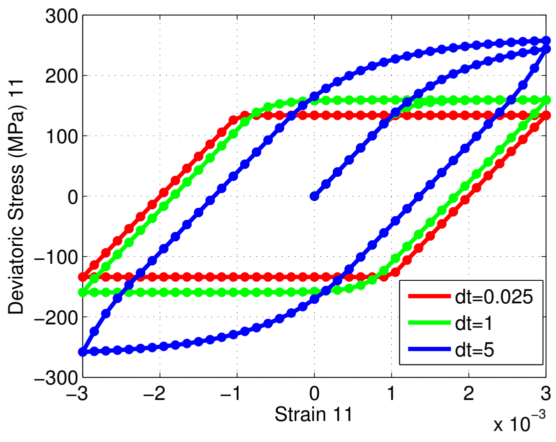

Figure 16: Rate Dependent Perfect Plasticity Model, σ_{11}^{dev} vs ϵ_{11}

Rate Dependent Linear Isotropic Model

The parameters used for this model are, $\sigma_y = 200, E = 200e3, H = 0, \eta = 1e4$. K was varied from 50e3, 100e3 and 200e3. The applied undamaged stress loading is [0, 800, 0, -800, 0, 800] time corresponding to these values is [0 1 2 3 4 5] and 20 sub steps were used for each step making dt=0.05.

Figure 17 and Figure 18 show stress-strain and stress-time plots with different K.

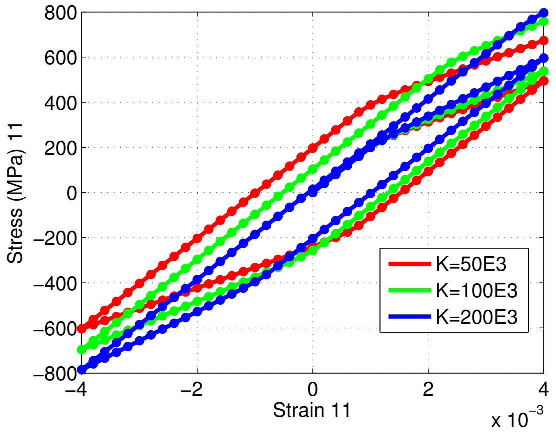

Figure 17: Rate Dependent Linear Isotropic Model, σ_{11} vs ϵ_{11}

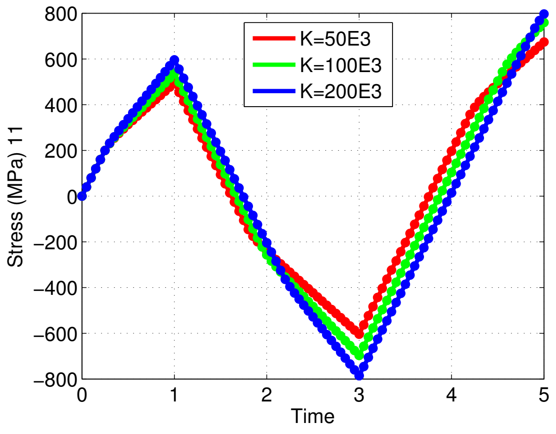

Figure 18: Rate Dependent Linear Isotropic Model, σ_{11} vs Time

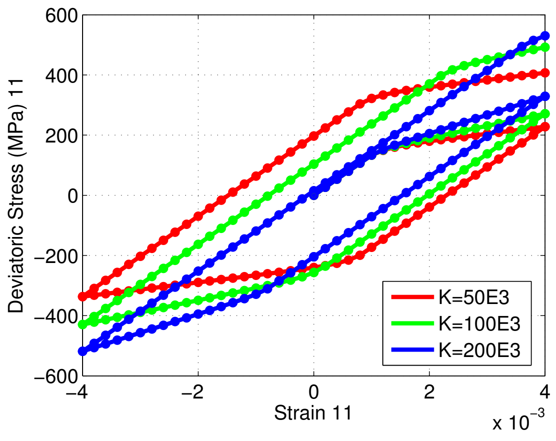

Figure 19: Rate Dependent Linear Isotropic Model, σ_{11}^{dev} vs ϵ_{11}

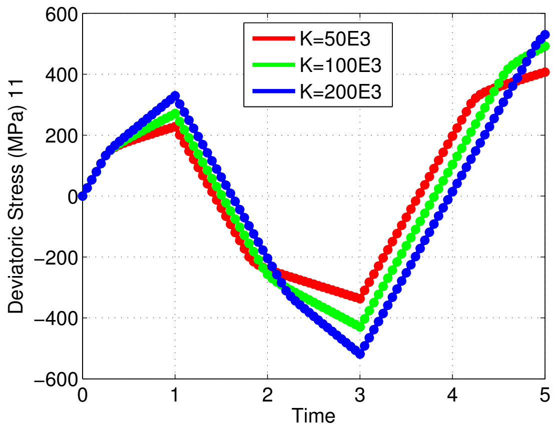

Figure 20: Rate Dependent Linear Isotropic Model, σ_{11}^{dev} vs Time

Rate Dependent Linear Kinematic Model

The parameters used for this model are, $\sigma_y = 200, E = 200e3, K = 0, H = 50e3$. In this model, η was varied from 1e4, 5e4 and 1e5. The applied undamaged stress loading is [0, 800, 0, -800, 0, 800] time corresponding to these values is [0 1 2 3 4 5] and 20 sub steps were used for each step making dt=0.05.

Figure 21 and Figure 22 show stress-strain and stress-time plots with different η. Similar observations can be made as that of rate dependent linear isotropic case. The model approached to time independent linear kinematic model as η becomes lesser. A very high value of η would make the material perfectly elastic. Also, it is seen from plots that the variation ofη affects more in case of kinematic hardening compared to isotropic case. Also, the stress space is enveloped by two inclined lines which is a typical scenario in case of kinematic hardening.

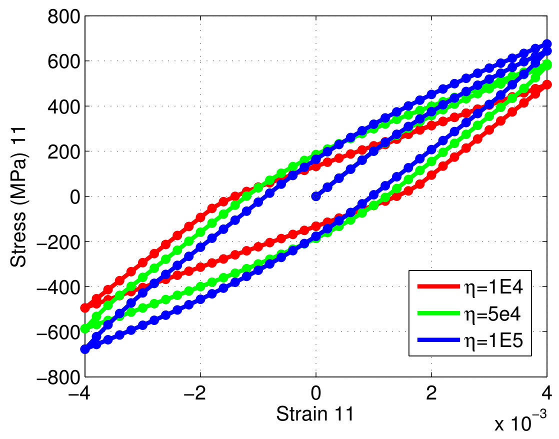

Figure 21: Rate Dependent Linear Kinematic Model, σ_{11} vs ϵ_{11}

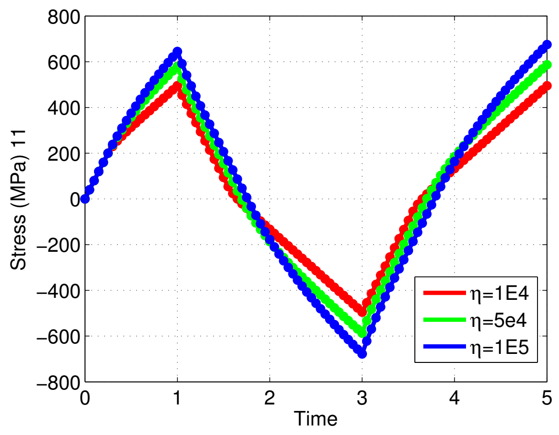

Figure 22: Rate Dependent Linear Kinematic Model, σ_{11} vs Time

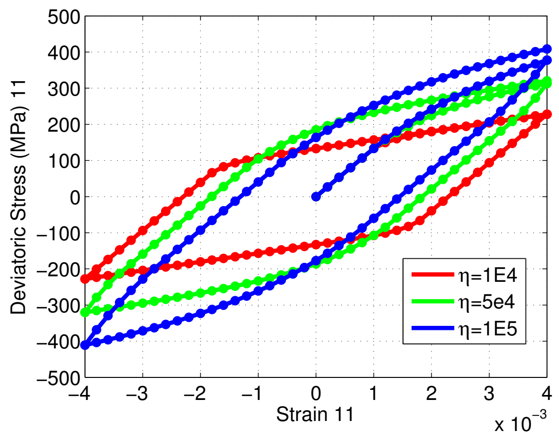

Figure 23: Rate Dependent Linear Kinematic Model, σ_{11}^{dev} vs ϵ_{11}

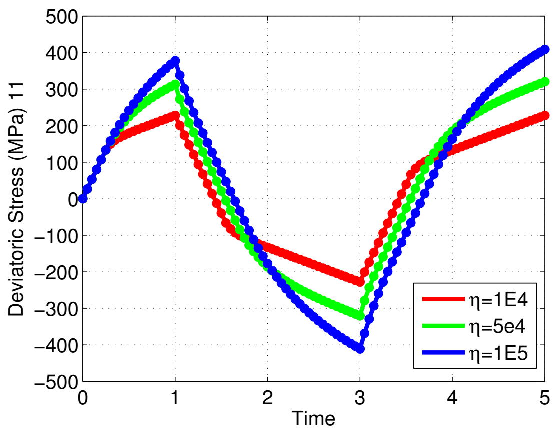

Figure 24: Rate Dependent Linear Kinematic Model, σ_{11}^{dev} vs Time

Rate Dependent Nonlinear Isotropic Model

The parameters used for this model are, $\sigma_y = 200, E = 200e3, K = 0, H = 0, cnvtol = 1e-7, \eta = 1e4$. The applied undamaged stress loading is [0, 800, 0, -800, 0, 800] time corresponding to these values is [0 1 2 3 4 5] and 20 sub steps were used for each step making dt=0.05.

Figure 25 show stress-strain behavior of rate independent and rate dependent model. The maximum stress that can be taken by the model is higher if it is a rate dependent model, since the viscous parameter allows extra allowance for stress to be outside yield surface. The isotropic nature can be proved by the fact that the stagnated stress value in the tension zone and that in compression zone are equal in magnitude, which is a typical scenario for isotropic models.

The variation in value of δ shows that, δ helps to raise the maximum stress that a material can take for a fixed value of viscous parameter. δ was varied from 0, 1e3 and 1e4.

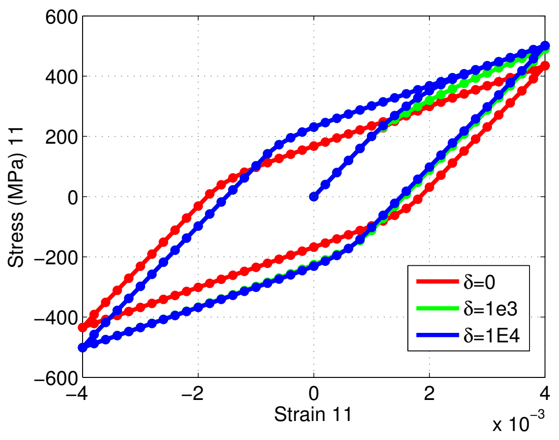

Figure 25: Rate Dependent Nonlinear Isotropic Model, σ_{11} vs ϵ_{11}

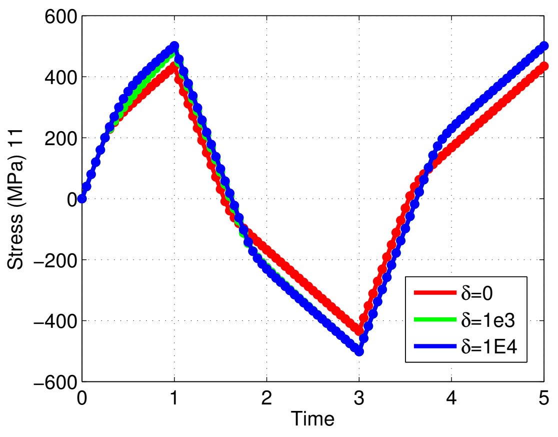

Figure 26: Rate Dependent Nonlinear Isotropic Model, σ_{11} vs Time

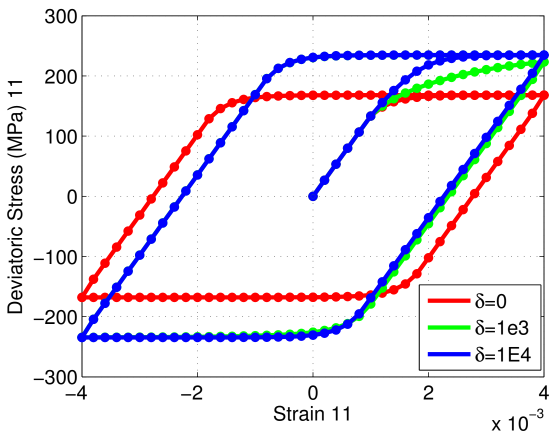

Figure 27: Rate Dependent Nonlinear Isotropic Model, σ_{11}^{dev} vs ϵ_{11}

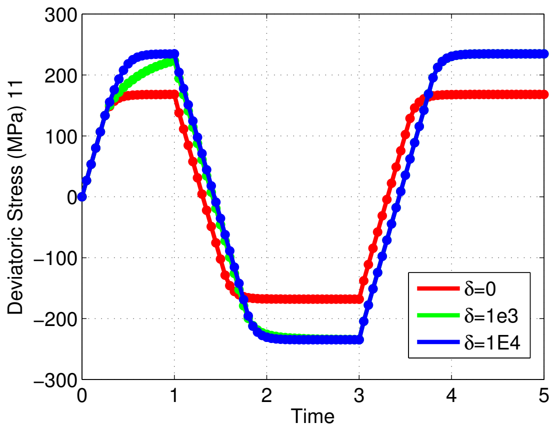

Figure 28: Rate Dependent Nonlinear Isotropic Model, σ_{11}^{dev} vs Time

Rate Dependent Nonlinear Isotropic Linear Kinematic Model

The parameters used for this model are, $\sigma_y = 200, E = 200e3, K = 0, H = 50e3, \delta = 1e3, cnvtol = 1e-7, \eta = 1e5$. The applied undamaged stress loading is [0, 400, 800] time corresponding to these values is [0 1 2] and 20 sub steps were used for each step making dt=0.05.

Figure 29 to Figure 30 compares the stress-time and stress-strain response of a rate dependent and rate independent model. As expected, the rate dependent model gives more allowance for stress.

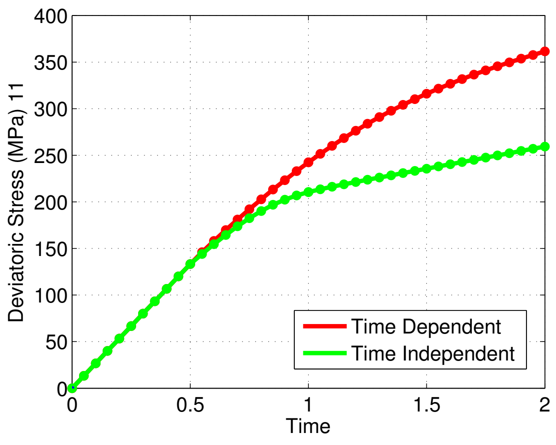

Figure 29: Nonlinear Isotropic Linear Kinematic Model, Comparison, σ_{11} vs ϵ_{11}

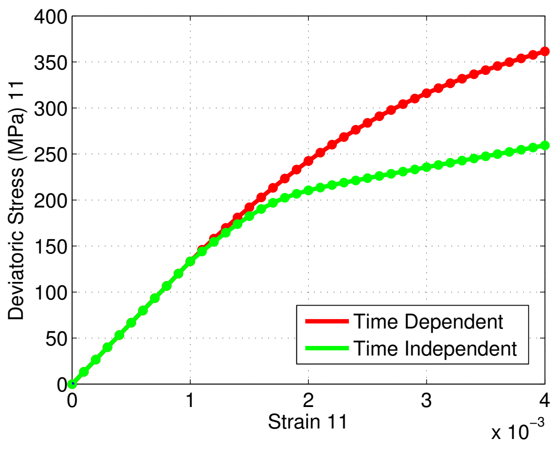

Figure 30: Nonlinear Isotropic Linear Kinematic Model, Comparison, σ_{11}^{dev} vs ϵ_{11}

Appendix - MATLAB Code

File: main.m

```
clc
clear all

%% parameters
sig_y   = 200;   %yield stress
E       = 200e3;    %youngs modulus
nu      = 0.3; %poissions ratio

K       = 0;    %isotropic hardening modulus
H       = 50e3; %kinematic hardening modulus
nsteps  = 20;  %no of sub-steps between two consecutive stress values

%% nonlinearity parameters
isotropic   = 'nonlinear';  %linear or nonlinear
sig_infty   = 300;            %sigma infinity
delta       = 1e4;            %control parameter for q
cnvtol      = 1e-7;         %tolerance for convergence of NR scheme

%% viscosity parameters
viscous = 'no';
eta = 1e5;
time = 1*[0:2]; %length of sig and t should be same

%% undamaged stresses, array can be extended to any length
sig         = [0,0,0,0,0,0
               400,0,0,0,0,0
               800,0,0,0,0,0]               ;
sig = sig.';
strain      = get_strain(sig,nsteps,E,nu);

%% time history calculation
if length(time) ~= size(sig,1)
    time = 0:size(sig,1)-1;
    disp('Time array length not same, modifying the time array')
end
t           = get_time(time,nsteps);

%% internal variables
strain_pl       = zeros(6,size(strain,2));
int_2           = zeros(1,size(strain,2));
int_3           = zeros(6,size(strain,2));

%% initialize stresses to 0
stress  = zeros(6,size(strain,2));

%% constitutive law loop (two functions depending on linearity / non
% linearity
if strcmp(isotropic,'linear') && strcmp(viscous,'no')
    for i = 2:1:size(strain,2)
        [stress(:,i), strain_pl(:,i), int_2(:,i), int_3(:,i)] = ...
        constitutive_linear(strain(:,i),strain_pl(:,i-1),int_2(:,i-1),...
          int_3(:,i-1),E,nu,sig_y,K,H);
      end
  elseif strcmp(isotropic,'nonlinear') && strcmp(viscous,'no')
      for i = 2:1:size(strain,2)
          [stress(:,i), strain_pl(:,i), int_2(:,i), int_3(:,i)] = ...
          constitutive_nonlinear(strain(:,i),strain_pl(:,i-1),...
          int_2(:,i-1),int_3(:,i-1),E,nu,sig_y,K,H,sig_infty,delta,cnvtol);
      end
  elseif strcmp(isotropic,'linear') && strcmp(viscous,'yes')
     for i = 2:1:size(strain,2)
          [stress(:,i), strain_pl(:,i), int_2(:,i), int_3(:,i)] = ...
          constitutive_linear_visco(strain(:,i),strain_pl(:,i-1),int_2(:,i-1),...
          int_3(:,i-1),E,nu,K,H,sig_y,eta,t(i),t(i-1));
     end
  elseif strcmp(isotropic,'nonlinear') && strcmp(viscous,'yes')
      for i = 2:1:size(strain,2)
          [stress(:,i), strain_pl(:,i), int_2(:,i), int_3(:,i)] = ...
          constitutive_nonlinear_visco(strain(:,i),strain_pl(:,i-1),...
          int_2(:,i-1),int_3(:,i-1),E,nu,K,H,sig_y,sig_infty,delta,cnvtol,...
          eta,t(i),t(i-1));
      end
  else
      disp('invalid input')
  end

  %% get deviatoric stress components
  s_dev = zeros(6,size(stress,2));
  for i=1:1:size(stress,2)
      s_dev(:,i) = dev(stress(:,i));
  end

  %% Post Process
  % Stress-Strain Plot 11
  figure(1)
  hold on
  grid on
  box on
  index = (0:1:size(sig,2)-1)*nsteps+1;
  colormat = ['r','g','b','m','c','r','g','b','m','c'].';
  colormat = [colormat;colormat;colormat;colormat];
  set(gca,'fontsize',14);
  set(gcf,'color','white')

  counter = 1;
  for j = 1:1: length(index)-1
      startp = index(j);
      endp = index(j+1);
      plot(strain(1,startp:endp),stress(1,startp:endp),...
        strcat('.',bigcol,'-'),'LineWidth',3,'MarkerSize',20);
      %text(strain(1,startp),stress(1,startp),num2str(j),'FontSize',14);
      xlabel('Strain 11')
      ylabel('Stress (MPa) 11','FontSize',14)
      counter = counter + 1;
  end
  %text(strain(1,endp),stress(1,endp),num2str(j+1),'FontSize',14);

  figure(2) %Stress-Time Plot 11
  hold on
  grid on
  box on
  set(gca,'fontsize',14);
  set(gcf,'color','white')
  counter = 1;
  for j = 1:1: length(index)-1
      startp = index(j);
      endp = index(j+1);
      plot(t(startp:endp),stress(1,startp:endp),...
        strcat('.',bigcol,'-'),'LineWidth',3,'MarkerSize',20);
      %text(t(startp),stress(1,startp),num2str(j),'FontSize',14);
      xlabel('Time')
      ylabel('Stress (MPa) 11','FontSize',14)
      counter = counter + 1;
  end
  %text(t(endp),stress(1,endp),num2str(j+1),'FontSize',14);

  figure(3) %Dev Stress-Strain 11
  hold on
  grid on
  box on
  set(gca,'fontsize',14);
  set(gcf,'color','white')
  counter = 1;
  for j = 1:1: length(index)-1
      startp = index(j);
      endp = index(j+1);
      plot(strain(1,startp:endp),s_dev(1,startp:endp),...
        strcat('.',bigcol,'-'),'LineWidth',3,'MarkerSize',20);
      %text(strain(1,startp),s_dev(1,startp),num2str(j),'FontSize',14);
      xlabel('Strain 11')
      ylabel('Deviatoric Stress (MPa) 11','FontSize',14)
      counter = counter + 1;
  end
  %text(strain(1,endp),s_dev(1,endp),num2str(j+1),'FontSize',14);

  figure(4) %Dev Stress-time 11
  hold on
  grid on
  box on
  set(gca,'fontsize',14);
  set(gcf,'color','white')

  counter = 1;
```

```
    for j = 1:1: length(index)-1
        startp = index(j);
        endp = index(j+1);
        plot(t(1,startp:endp),s_dev(1,startp:endp),...
          strcat('.',bigcol,'-'),'LineWidth',3,'MarkerSize',20);
        %text(strain(1,startp),s_dev(1,startp),num2str(j),'FontSize',14);
        xlabel('Time')
        ylabel('Deviatoric Stress (MPa) 11','FontSize',14)
        counter = counter + 1;
    end
```

File: get_time.m

```
function t = get_time(time,nsteps)
% initialize lengths
len_t  = (length(time)-1)*nsteps+1;
t_short   = zeros(nsteps,length(time)-1);
%create a column vector for every substep in sigma
for i=1:1:length(time)-1
    temp            = linspace(time(i),time(i+1),nsteps+1);
    t_short(:,i)    = temp(1:end-1).';
end
%combine to create a full sigma
t = [reshape(t_short,1,len_t-1),time(end)];
```

File: get_strain.m

```
function strain = get_strain(sig,nsteps,E,nu)
len_strain  = (size(sig,2)-1)*nsteps+1;
sig_short   = zeros(nsteps,size(sig,2)-1);
sig_full = zeros(6,len_strain);
for row=1:1:6
    for i=1:1:size(sig,2)-1
        temp            = linspace(sig(row,i),sig(row,i+1),nsteps+1);
        sig_short(:,i)  = temp(1:end-1).';
    end
    sig_full(row,:) = [reshape(sig_short,1,len_strain-1),sig(row,end)];
end
strain= inv(get_c(E,nu))*sig_full;
```

File: new_raph_data.m

```
function [D,R] = new_raph_data(sig_infty,sig_y,E,nu,H,K,delta,int_2_n,gamma_k,f_trial)
% D = derivative at gamma_k
% R = residual at gamma_k
mu = E/2/(1+nu);
D = -(2*mu+2/3*H)-delta*2/3*(sig_infty-sig_y)*exp(-delta*(sqrt(2/3)*gamma_k+int_2_n));
f2 = f(sig_infty,sig_y,delta,sqrt(2/3)*gamma_k+int_2_n);
f1 = f(sig_infty,sig_y,delta,int_2_n);
R = f_trial - gamma_k*(2*mu+2/3*H) - sqrt(2/3)*(f2-f1);
end
```

File: f.m

```
function val = f(sig_infty,sig_y,delta,p)
%evaluated the value of the below function for given input parameters
val = (sig_infty-sig_y)*(1-exp(-delta*p));
```

File: constitutive_linear.m

```
function [s_n1, ep_n1, int_2_n1, int_3_n1] = ...
        constitutive_linear(et_n1,ep_n,int_2_n,int_3_n,E,nu,sig_y,K,H)

% variables naming convention

% input
% ep_n      = plastic strain old
% int_2_n   = internal variable 2 old
% int_3_n   = internal variable 3 old
% sig_y     = yield stress
% et_n1     = strain total new
% E,K,H     = Youngs, Isotropic, Kinematic hardening Modulus

% output
% s_n1      = stress new
% ep_n1     = plastic strain new
% int_2_n1  = internal variable 2 new
% int_3_n1  = internal variable 3 new
% E_tan     = Elasto-plastic tangent modulus

mu = E/2/(1+nu);

%trial state of internal variables
s_trial         = get_c(E,nu)*(et_n1-ep_n);
int_2_trial     = int_2_n;
int_3_trial     = int_3_n;


%trial state of dependent variables
q_trial         = -K*int_2_trial;
q_bar_trial     = -2*H/3*int_3_trial;

%yield function
f_trial         = norm_dist(dev(s_trial)-q_bar_trial) - sqrt(2/3)*(sig_y-q_trial);
disp(get_n(dev(s_trial)-q_bar_trial))

if f_trial <= 0     % elastic loading-unloading or neutral loading
    s_n1        = s_trial;
    ep_n1       = ep_n;
    int_2_n1    = int_2_trial;
    int_3_n1    = int_3_trial;
else                % plastic loading
    gamma       = f_trial/(2*mu+2/3*K+2/3*H);
    ep_n1       = ep_n + gamma*get_n(dev(s_trial)-q_bar_trial);
    int_2_n1    = int_2_n + gamma*sqrt(2/3);
    int_3_n1    = int_3_n - gamma*get_n(dev(s_trial)-q_bar_trial);
    s_n1        = get_c(E,nu)*(et_n1-ep_n1);
end
end
```

File: dev.m

```
function sol = dev(sig)
if size(sig,1) == 1;
    sig = sig.';
end
trace = sum(sig(1:3));
sol = sig-1/3*trace*[1;1;1;0;0;0];
```

File: constitutive_nonlinear.m

```
function [s_n1, ep_n1, int_2_n1, int_3_n1] = ...
         constitutive_nonlinear...
         (et_n1,ep_n,int_2_n,int_3_n,E,nu,sig_y,K,H,sig_infty,delta,cnvtol)

% variables naming convention

% input
% ep_n      = plastic strain old
% int_2_n   = internal variable 2 old
% int_3_n   = internal variable 3 old
% sig_y     = yield stress
% et_n1     = strain total new
% E,K,H     = Youngs, Isotropic, Kinematic hardening Modulus
% sig_infty = sigma infinity
% delta     = material property
% cnvtol    = convergence tolerance

% output
% s_n1      = stress new
% ep_n1     = plastic strain new
% int_2_n1  = internal variable 2 new
% int_3_n1  = internal variable 3 new
% E_tan     = Elasto-plastic tangent modulus

% temporary
% D         = derivative calculated at gamma_k
% R         = residual of nonlinear equation

mu = E/2/(1+nu);

%trial state of internal variables
s_trial          = get_c(E,nu)*(et_n1-ep_n);
int_2_trial      = int_2_n;
int_3_trial      = int_3_n;

%trial state of dependent variables
q_bar_trial      = -2*H/3*int_3_trial;
q_trial          = -f(sig_infty,sig_y,delta,int_2_trial);

%yield function
f_trial          = norm_dist(dev(s_trial)-q_bar_trial) - sqrt(2/3)*(sig_y-q_trial);

if f_trial <= 0      % elastic loading-unloading or neutral loading
        s_n1        = s_trial;
        ep_n1       = ep_n;
        int_2_n1    = int_2_trial;
        int_3_n1    = int_3_trial;
        return
else                 %plastic loading
```

```
            while true
                [D,R]           = new_raph_data(sig_infty,sig_y,E,nu,H,K,...
                                  delta,int_2_n,gamma_k,f_trial);
                %convergence check using the residual
                if abs(R) < cnvtol
                    break
                else
                    gamma_k     = gamma_k - R/D;
                end
            end
            gamma       = gamma_k;
            ep_n1       = ep_n + gamma*get_n(dev(s_trial)-q_bar_trial);
            int_2_n1    = int_2_n + gamma*sqrt(2/3);
            int_3_n1    = int_3_n - gamma*get_n(dev(s_trial)-q_bar_trial);
            s_n1        = get_c(E,nu)*(et_n1-ep_n1);
    end
    end
```

File: get_n.m

```
function sol = get_n(val)
num = val;
deno = norm_dist(val);
sol=num/deno;
```

File: get_c.m

```
function C = get_c(E,nu)
fact = E/(1+nu)/(1-2*nu);
C = [1-nu,      nu,         nu,         0,          0,          0
    nu,         1-nu,       nu,         0,          0,          0
    nu,         nu,         1-nu,       0,          0,          0
    0,          0,          0,          0.5-nu,     0,          0
    0,          0,          0,          0,          .5-nu,      0
    0,          0,          0,          0,          0,          .5-nu];
C=fact*C;
```

File: norm_dist.m

```
function sol = norm_dist(a)
sol = sqrt(sum(a.*a));
end
```

File: new_raph_data_visco.m

```
function [D,R] = new_raph_data_visco(sig_infty,sig_y,E,nu,H,K,delta,...
    int_2_n,gamma_k,f_trial,eta,dt)
% D = derivative at gamma_k
% R = residual at gamma_k
mu = E/2/(1+nu);
D = -(2*mu+2/3*H+eta/dt)-delta*2/3*(sig_infty-sig_y)*exp(-delta*(sqrt(2/3)↙
*gamma_k+int_2_n));
f2 = f(sig_infty,sig_y,delta,sqrt(2/3)*gamma_k+int_2_n);
f1 = f(sig_infty,sig_y,delta,int_2_n);
R = f_trial - gamma_k*(2*mu+2/3*H+eta/dt) - sqrt(2/3)*(f2-f1);
end
```

File: constitutive_linear_visco.m

```
function [s_n1, ep_n1, int_2_n1, int_3_n1] = ...
        constitutive_linear_visco(et_n1,ep_n,int_2_n,int_3_n,...
        E,nu,K,H,sig_y,eta,tend,tstart)

% variables naming convention

% input
% ep_n       = plastic strain old
% int_2_n    = internal variable 2 old
% int_3_n    = internal variable 3 old
% sig_y      = yield stress
% et_n1      = strain total new
% E,K,H      = Youngs, Isotropic, Kinematic hardening Modulus

% output
% s_n1       = stress new
% ep_n1      = plastic strain new
% int_2_n1   = internal variable 2 new
% int_3_n1   = internal variable 3 new
% E_tan      = Elasto-plastic tangent modulus

mu = E/2/(1+nu);
dt = tend-tstart;

%trial state of internal variables
s_trial         = get_c(E,nu)*(et_n1-ep_n);
int_2_trial     = int_2_n;
int_3_trial     = int_3_n;

%trial state of dependent variables
q_trial         = -K*int_2_trial;
q_bar_trial     = -2*H/3*int_3_trial;

%yield function
f_trial         = norm_dist(dev(s_trial)-q_bar_trial) - sqrt(2/3)*(sig_y-q_trial);

if f_trial <= 0     % elastic loading-unloading or neutral loading
    s_n1        = s_trial;
    ep_n1       = ep_n;
    int_2_n1    = int_2_trial;
    int_3_n1    = int_3_trial;
else                % plastic loading
    gamma       = f_trial/(2*mu+2/3*K+2/3*H+eta/dt);
    ep_n1       = ep_n + gamma*get_n(dev(s_trial)-q_bar_trial);
    int_2_n1    = int_2_n + gamma*sqrt(2/3);
    int_3_n1    = int_3_n - gamma*get_n(dev(s_trial)-q_bar_trial);
    s_n1        = get_c(E,nu)*(et_n1-ep_n1);
end
```

File: constitutive_nonlinear_visco.m

```
function [s_n1, ep_n1, int_2_n1, int_3_n1] = ...
         constitutive_nonlinear_visco...
         (et_n1,ep_n,int_2_n,int_3_n,E,nu,K,H,sig_y,sig_infty,delta,cnvtol,...
         eta,tend,tstart)

% variables naming convention

% input
% ep_n      = plastic strain old
% int_2_n   = internal variable 2 old
% int_3_n   = internal variable 3 old
% sig_y     = yield stress
% et_n1     = strain total new
% E,K,H     = Youngs, Isotropic, Kinematic hardening Modulus
% sig_infty = sigma infinity
% delta     = material property
% cnvtol    = convergence tolerance

% output
% s_n1      = stress new
% ep_n1     = plastic strain new
% int_2_n1  = internal variable 2 new
% int_3_n1  = internal variable 3 new
% E_tan     = Elasto-plastic tangent modulus

% temporary
% D         = derivative calculated at gamma_k
% R         = residual of nonlinear equation

dt = tend-tstart;
mu = E/2/(1+nu);

%trial state of internal variables
s_trial         = get_c(E,nu)*(et_n1-ep_n);
int_2_trial     = int_2_n;
int_3_trial     = int_3_n;

%trial state of dependent variables
q_bar_trial     = -2*H/3*int_3_trial;
q_trial         = -f(sig_infty,sig_y,delta,int_2_trial);

%yield function
f_trial         = norm_dist(dev(s_trial)-q_bar_trial) - sqrt(2/3)*(sig_y-q_trial);

if f_trial <= 0     % elastic loading-unloading or neutral loading
        s_n1        = s_trial;
        ep_n1       = ep_n;
        int_2_n1    = int_2_trial;
        int_3_n1    = int_3_trial;
        return
else                %plastic loading
```

```
        gamma_k = 0; %some starting value
        %newton raphson iterations
        while true
            [D,R]           = new_raph_data_visco(sig_infty,sig_y,E,nu,H,K,...
                              delta,int_2_n,gamma_k,f_trial,eta,dt);
                          %convergence check using the residual
            if abs(R) < cnvtol
                break
            else
                gamma_k     = gamma_k - R/D;
            end
        end
        gamma       = gamma_k;
        ep_n1       = ep_n + gamma*get_n(dev(s_trial)-q_bar_trial);
        int_2_n1    = int_2_n + gamma*sqrt(2/3);
        int_3_n1    = int_3_n - gamma*get_n(dev(s_trial)-q_bar_trial);
        s_n1        = get_c(E,nu)*(et_n1-ep_n1);
end
end
```