# 1D 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, $\sigma_y = 200, E = 200e3, K = 0, H = 0$. The applied undamaged stress loading is [0, 800, 0, -800, 0, 800] and 20 sub steps were used for each step. Figure 1 shows the result. Note that all results are based on strain driven models.

Material reaches yield limit and starts perfectly plastic behavior at point P1 (0.001,200). As expected, the Y coordinate equals the yield stress and slope of this point is 200e3, the Youngs Modulus of the material. The unloading slope was also found to match with Youngs Modulus.

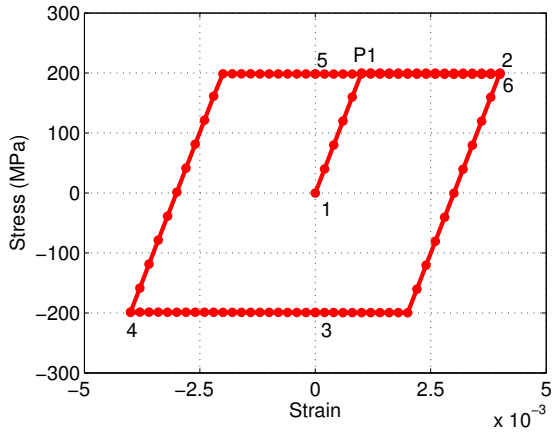


Figure 1: Rate Independent Perfect Plastic Model

### 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 kept as 50e3. The applied undamaged stress loading is same as before i.e. [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 2. This slope was found to be 40000, which equals $\frac{EK}{(E+K)}$. 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 (0.004,320) and point P1 (-0.0008,-320).

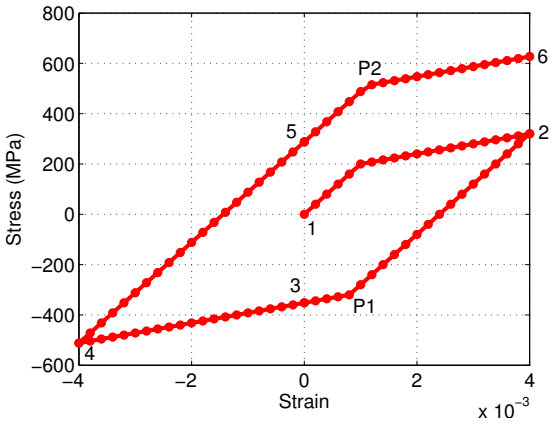


Figure 2: Rate Independent Linear Isotropic Model

To study the dependence on hardening, three cases were simulated (Figure 3) keeping loading same as before, K=50e3, 100e3 and 200e3. It is expected, that as hardening modulus increases, the slope after yield would increase. A trivial case, where K=0 is perfectly plastic.

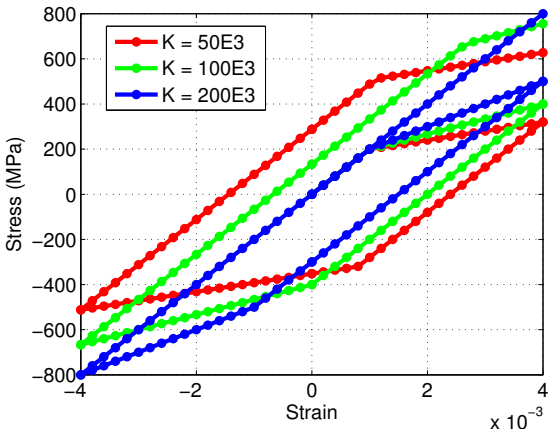


Figure 3: Rate Independent Linear Isotropic Model, Different K

### 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 kept as 50e3. The applied undamaged stress loading is same as before i.e. [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. This slope was found to be 40000, which equals $\frac{EH}{(E+H)}$. Also, kinematic nature of hardening implies that difference of elastic limit in tension and compression should be same and equal to $2 * \sigma_y$ (400). This is observed to be correct based on Y coordinates of point 2 (0.004,320) and point P1 (-0.002,-80).

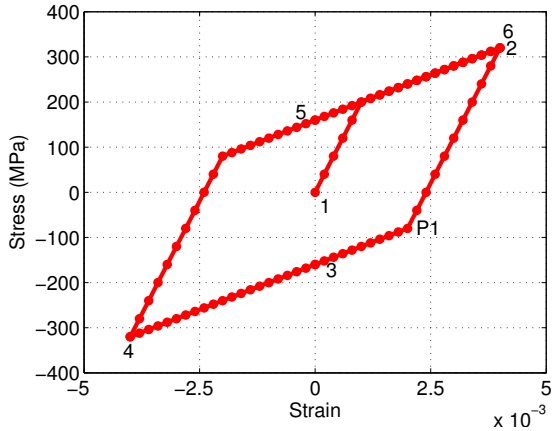


Figure 4: Rate Independent Linear Kinematic Model

To study the dependence on hardening, three cases were simulated (Figure 5) keeping loading same as before, H=50e3, 100e3 and 200e3. It is expected, that as hardening modulus increases, the slope after yield would increase. A trivial case, where H=0 is perfectly plastic.

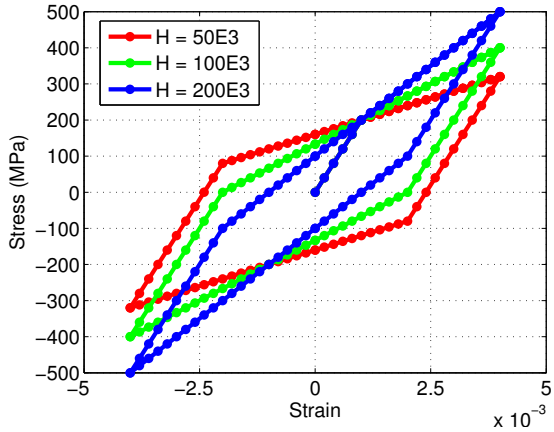


Figure 5: Rate Independent Linear Kinematic Model, Different H

### 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 undamaged stress loading is [0, 600, 0, -600, 0, 600] and 20 sub steps were used for each step.

The stress approaches $\sigma_\infty$ (300) nonlinearly when loaded in compression or tension. 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 (279.8) and $\sigma_\infty$ should be same as difference between point 3 (-280.8) and -$\sigma_\infty$, since $\sigma_\infty$ is the total margin for the yield surface to expand.

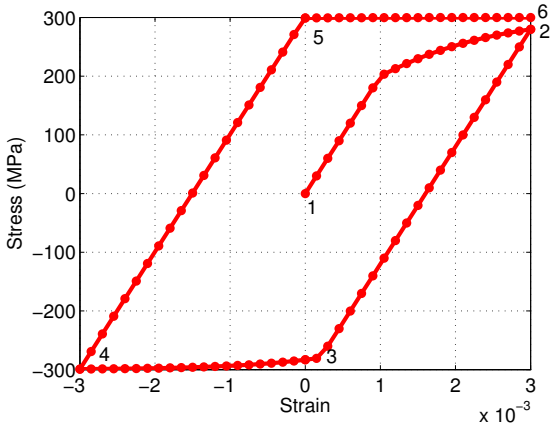


Figure 6: Rate Independent Nonlinear Isotropic Model

To study variation with parameter $\delta$, 4 simulations with $\delta$=0, 2e2, 1e3 and 1e4 were performed. A higher value would imply that $\sigma_\infty$ is reached at a faster rate (Figure 7). A trivial case, $\delta = 0$, would imply, that $\Pi'(\xi) = 0$, which implies no isotropic hardening. Thus, the curve resembles perfect plasticity.

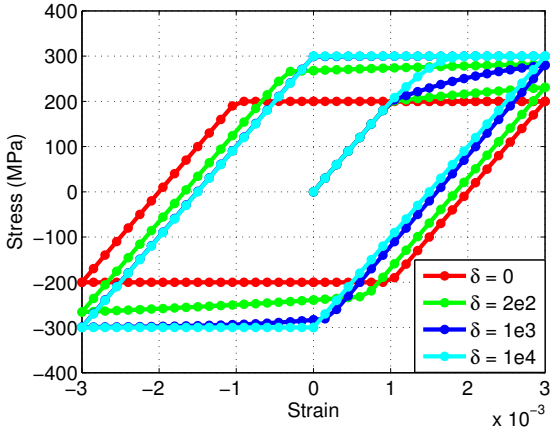


Figure 7: Rate Independent Nonlinear Isotropic Model, Different $\delta$

### 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 8.

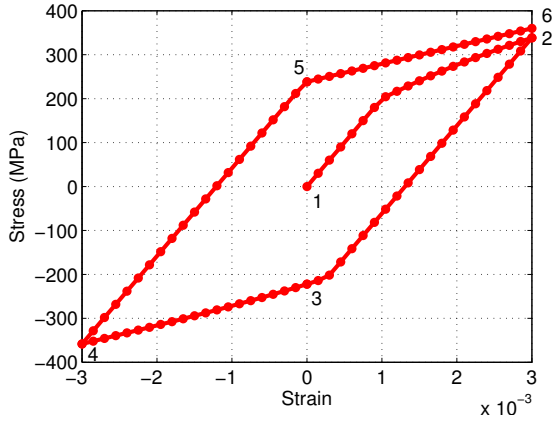


Figure 8: Rate Independent Nonlinear Isotropic Linear Kinematic Model

### Rate Dependent Perfect Plasticity Model

The parameters used for this model are, $\sigma_y = 200, E = 200e3, K = 0, H = 0, \eta = 1e4$. 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 9)allow stress to exceed $\sigma_y$ by an amount which is determined by strain rate and viscosity parameter $\eta$. The dependence of $\eta$ indicates (10) that at very low $\eta$, rate dependent model becomes equivalent to rate independent model (Figure 10), where three simulations were run for $\eta$= 0, 1e4 and 3e4. As $\eta$ 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 $\sigma_\infty$. But in case of this model, the nonlinearity is shown every time, the stress exceeds $\sigma_y$.

The effect of changing strain rate was observed with variation in dt= 0.025, 0.05, 0.1 and 5. It is seen that the as strain rate decreases (dt increases), the curve approaches rate independent perfect plasticity model (Figure 11)

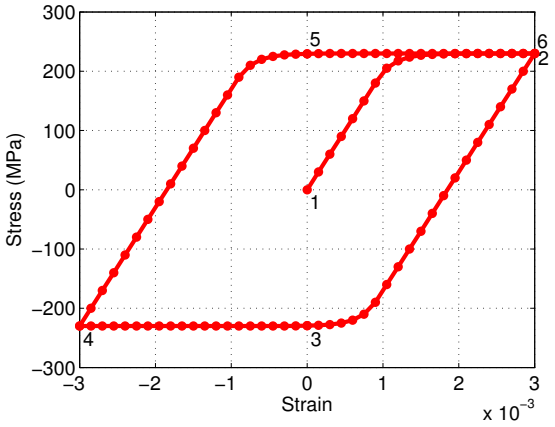


Figure 9: Rate Dependent Perfect Plasticity Model

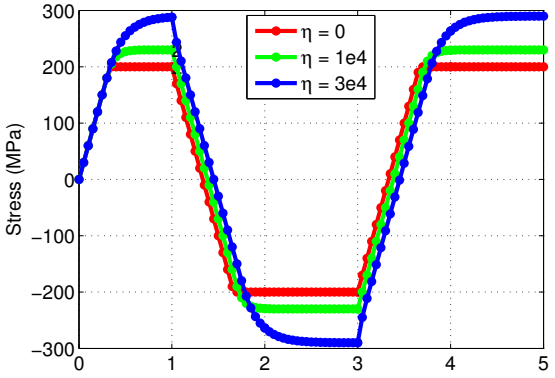


Figure 10: Rate Dependent Perfect Plasticity Model, Different $\eta$

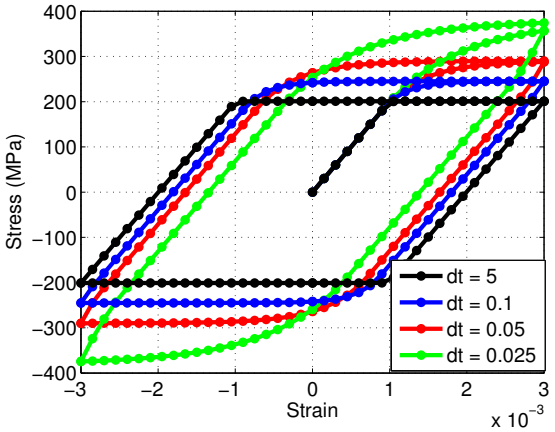


Figure 11: Rate Dependent Perfect Plasticity Model, Different dt

### Rate Dependent Linear Isotropic Model

The parameters used for this model are, $\sigma_y = 200, E = 200e3, K = 50e3, H = 0$. $\eta$ 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 12 and Figure 13 show stress-strain and stress-time plots with different $\eta$. Similar observations can be made as that of perfect plasticity case. The model approached to time independent linear isotropic model as $\eta$ becomes lesser. A very high value of $\eta$ would make the material perfectly elastic.

Stress-Time graph indicates that materials with high viscous parameter can resist more load compared to one with less viscous parameter.

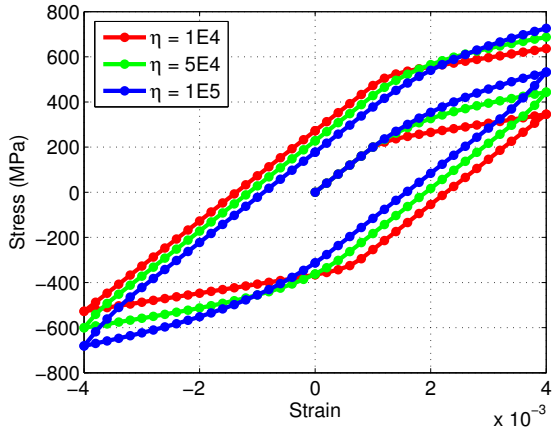


Figure 12: Rate Dependent Linear Isotropic Model, Stress-Strain

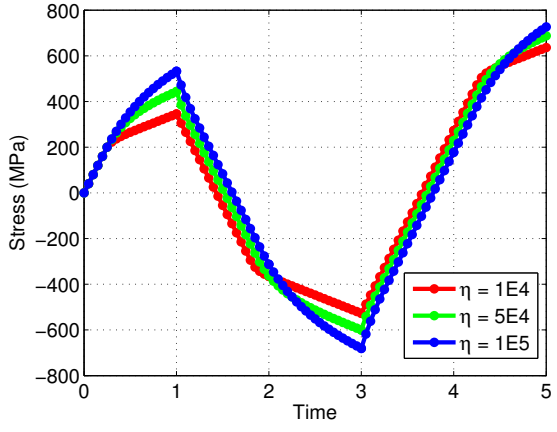


Figure 13: Rate Dependent Linear Isotropic Model, Stress-Time

### Rate Dependent Linear Kinematic Model

The parameters used for this model are, $\sigma_y = 200, E = 200e3, K = 0, H = 50e3$. $\eta$ 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 14 and Figure 15 show stress-strain and stress-time plots with different $\eta$. Similar observations can be made as that of rate dependent linear isotropic case. The model approached to time independent linear kinematic model as $\eta$ becomes lesser. A very high value of $\eta$ would make the material perfectly elastic. Also, it is seen from plots that the variation of$\eta$ 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.

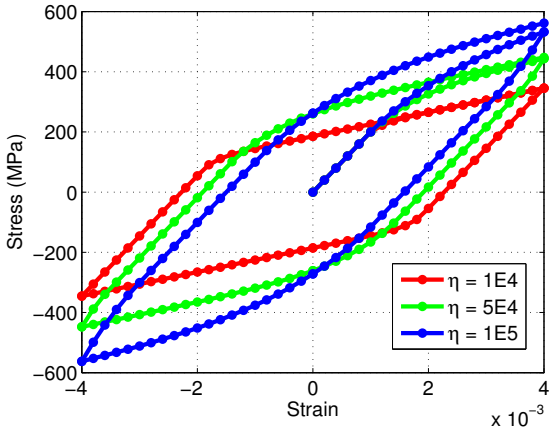


Figure 14: Rate Dependent Linear Kinematic Model, Stress-Strain

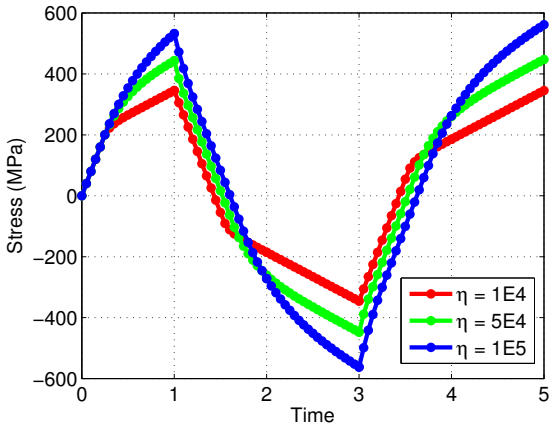


Figure 15: Rate Dependent Linear Kinematic Model, Stress-Time

### Rate Dependent Nonlinear Isotropic Model

The parameters used for this model are, $\sigma_y = 200, E = 200e3, K = 0, H = 0, \delta = 1e3, cnvtol = 1e-7, \eta = 1e4$. 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 16 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 is 339.9 and in compression zone it is -338.7. These values are equal in magnitude, which is a typical scenario for isotropic models.

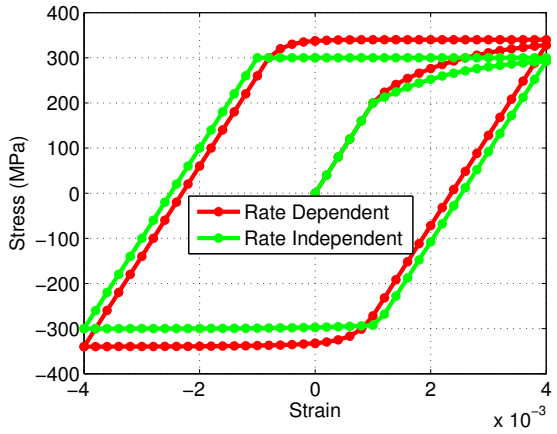


Figure 16: Rate Dependent Nonlinear Isotropic Model, Comparison

Figure 17 shows that $\delta$ helps to raise the maximum stress that a material can take for a fixed value of viscous parameter. $\delta$ was varied from 1e3, 1e2 to 1.

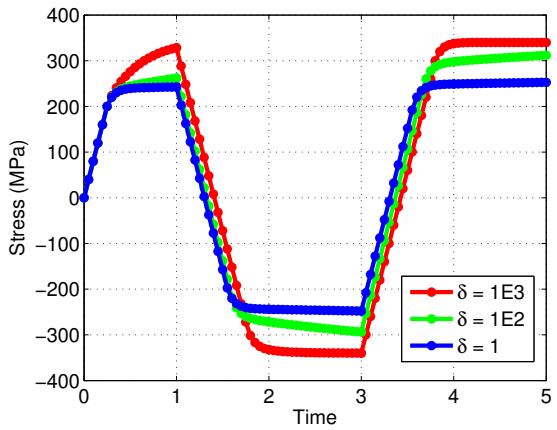


Figure 17: Rate Dependent Nonlinear Isotropic Model, Stress-Time

### 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 18 and Figure 19 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.

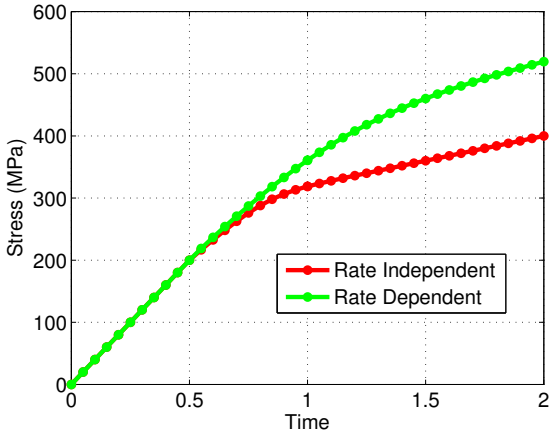


Figure 18: Nonlinear Isotropic Linear Kinematic Model, Comparison, Stress-Strain

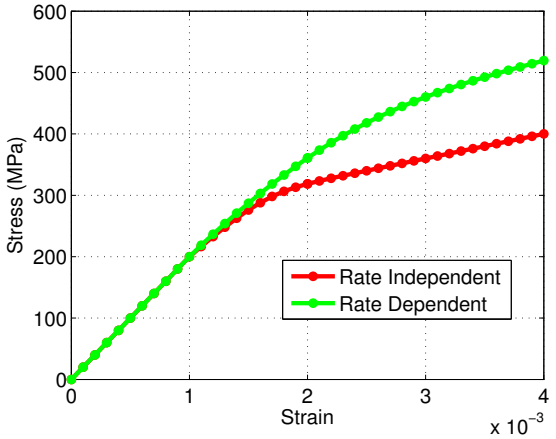


Figure 19: Nonlinear Isotropic Linear Kinematic Model, Comparison, Stress-Time

# Appendix - MATLAB Code

File: main.m

```
clc; clear all

%% parameters
sig_y    = 200;       %yield stress
E        = 200e3;     %youngs modulus
K        = 0;         %isotropic hardening modulus
H        = 0;         %kinematic hardening modulus
nsteps   = 20;        %no of sub-steps

%% nonlinearity parameters
isotropic    = 'linear';      %linear or nonlinear
sig_infty    = 300;           %sigma infinity
delta        = 1e3;           %control parameter for q
cnvtol       = 1e-7;          %tolerance NR scheme

%% viscosity parameters
viscous = 'no';
eta = 1e4;
time =1*[0:3]; %length of sig and t should be same

%% time history calculation
t            = get_time(time,nsteps);

%% undamaged stresses, array can be extended
sig          = [0,800,0,-800,0,800];
strain       = get_strain(sig,nsteps,E);

%% internal variables
strain_pl        = zeros(1,length(strain));
int_2            = zeros(1,length(strain));
int_3            = zeros(1,length(strain));

%% initialize stresses to 0
stress   = zeros(1,length(strain));

%% constitutive law loop
if strcmp(isotropic,'linear') && strcmp(viscous,'no')
    for i = 2:1:length(strain)
        [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,K,H,sig_y);
    end
elseif strcmp(isotropic,'nonlinear') && strcmp(viscous,'no')
    for i = 2:1:length(strain)
        [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,K,H,sig_y,sig_infty,delta,cnvtol);
    end
elseif strcmp(isotropic,'linear') && strcmp(viscous,'yes')
   for i = 2:1:length(strain)
        [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,K,H,sig_y,eta,t(i),t(i-1));
    end
elseif strcmp(isotropic,'nonlinear') && strcmp(viscous,'yes')
    for i = 2:1:length(strain)
        [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,K,H,sig_y,sig_infty,delta,cnvtol,...
        eta,t(i),t(i-1));
    end
else
    disp('invalid input')
end

%% Post Process

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

figure(2) %Stress-Time Plot
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(startp:endp),...
          strcat('.','g','-'),'LineWidth',3,'MarkerSize',20);
        text(t(startp),stress(startp),num2str(j),'FontSize',14);
        xlabel('Time')
        ylabel('Stress (MPa)','FontSize',14)
        counter = counter + 1;
    end
    text(t(endp),stress(endp),num2str(j+1),'FontSize',14);
```

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));
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)

% initialize lengths
len_strain  = (length(sig)-1)*nsteps+1;
sig_short   = zeros(nsteps,length(sig)-1);

%create a column vector for every substep in sigma
for i=1:1:length(sig)-1
    temp            = linspace(sig(i),sig(i+1),nsteps+1);
    sig_short(:,i)  = temp(1:end-1).';
end

%combine to create a full sigma
sig_full = [reshape(sig_short,1,len_strain-1),sig(end)];

%get strain from stress
strain = sig_full/E;
```

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,K,H,sig_y)

% 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

%trial state of internal variables
s_trial         = E*(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     = -H*int_3_trial;

%yield function
f_trial         = abs(s_trial - q_bar_trial) - 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/(E+K+H);
    ep_n1       = ep_n + gamma*sign(s_trial - q_bar_trial);
    int_2_n1    = int_2_n + gamma;
    int_3_n1    = int_3_n - gamma*sign(s_trial - q_bar_trial);
    s_n1        = E*(et_n1-ep_n1);
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,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

dt = tend-tstart;

%trial state of internal variables
s_trial         = E*(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     = -H*int_3_trial;

%yield function
f_trial         = abs(s_trial - q_bar_trial) - 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
    gammadt       = f_trial/(E+K+H+eta/dt);
    ep_n1       = ep_n + gammadt*sign(s_trial - q_bar_trial);
    int_2_n1    = int_2_n + gammadt;
    int_3_n1    = int_3_n - gammadt*sign(s_trial - q_bar_trial);
    s_n1        = E*(et_n1-ep_n1);
end
```

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,K,H,sig_y,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

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

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

%yield function
f_trial         = abs(s_trial - q_bar_trial) - 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 = f_trial/(E+K+H); % some starting value
        %newton raphson iterations
        while true
            [D,R]           = new_raph_data(sig_infty,sig_y,E,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;
        %update internal variables
        ep_n1       = ep_n + gamma*sign(s_trial - q_bar_trial);
        int_2_n1    = int_2_n + gamma;
        int_3_n1    = int_3_n - gamma*sign(s_trial - q_bar_trial);
        %update stress
        s_n1        = E*(et_n1-ep_n1);
    end
    end
```

File: new_raph_data.m

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

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

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

%yield function
f_trial         = abs(s_trial - q_bar_trial) - 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,H,K,...
                              delta,int_2_n,gamma_k,f_trial,eta,dt);

            if abs(R) < cnvtol
                break
            else
                gamma_k     = gamma_k - R/D;
            end
        end
        gamma       = gamma_k;
        ep_n1       = ep_n + gamma*dt*sign(s_trial - q_bar_trial);
        int_2_n1    = int_2_n + gamma*dt;
        int_3_n1    = int_3_n - gamma*dt*sign(s_trial - q_bar_trial);
        s_n1        = E*(et_n1-ep_n1);
    end
    end
```

File: new_raph_data_visco.m

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

end
```