Universitat Politècnica De Catalunya, Barcelona

MSc. Computational Mechanics Erasmus Mundus

Assignment 2.1: 1D Plasticity

# Computational Solid Mechanics

*Author:*
Nikhil Dave

Date: May 2, 2018

# Contents

# List of Figures



# List of Tables

# 1 Introduction

The purpose of this work is to analyse the behaviour of the material considering various 1D plasticity models. To this effect, a MATLAB program is implemented to perform numerical simulations for these models and generate post-processed stress-strain and stress-time graphs to examine and validate our understanding of 1D plasticity.

## 1.1 Input data and Material parameters

The implementation requests the user to determine several parameters required for the analysis and also suggests default values that could be considered. These values define the model type the user desires to analyse. The default material parameters are specified as the properties of metal in order to simulate close to a real-world scenario. The default input data and material properties are given in Table 1. It is important to remark that for specific cases few parameters are not needed and are accordingly neglected by the MATLAB program presented in the Appendix.

| **Input data & material parameters** | **Value** |
|---|---|
| Young's modulus, $\boldsymbol{E}$ | 2.1e+11 $Pa$ |
| Yield stress, $\boldsymbol{\sigma_y}$ | 4.0e+8 $Pa$ |
| Isotropic hardening modulus, $\boldsymbol{K}$ | 2.0e+10 $Pa$ |
| Kinematic hardening modulus, $\boldsymbol{H}$ | 1.0e+10 $Pa$ |
| Asymptotic maximum stress, $\boldsymbol{\sigma_\infty}$ | 9.0e+8 $Pa$ |
| Exponential saturation parameter, $\boldsymbol{\delta}$ | 150 |
| Viscous coefficient, $\boldsymbol{\eta}$ | 3.0e+10 $Pa \cdot s$ |
| Total time of simulation, $\boldsymbol{t}$ | 5 $s$ |
| Step size, $\boldsymbol{\Delta t}$ | 0.025 $s$ |

**Table 1:** Default input data and material properties used in the analysis

## 1.2 Loading path

The loading path for this analysis is defined as a strain-time curve which starts with a uniaxial loading state till $\varepsilon = 0.01$, surpassing tensile yield stress and achieving plastic loading. Next, an uniaxial unloading is performed till $\varepsilon = -0.01$, to surpass the compressive yield stress. This is followed by a loading state again until $\varepsilon = 0.01$. This collection of loading and unloading steps could be expressed as a cyclic loading path as shown in Figure 1.

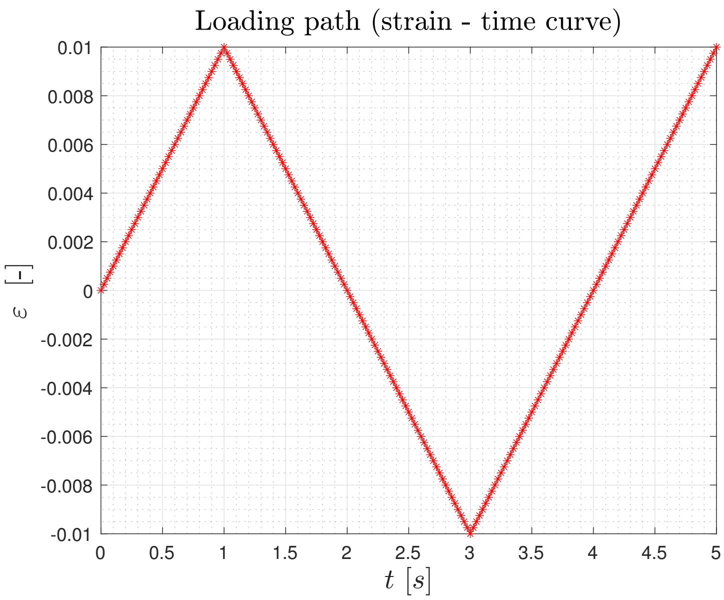



**Figure 1:** Strain-time curve: loading path considered in the analysis.

# 2 Perfect plasticity

## 2.1 Rate-independent model

In this section, we analyse the rate-independent perfect plasticity model for varying Young's modulus $E$. As associated with the perfectly plastic model, the stresses cannot exceed the yield stress and hence provide a constant value curve on the stress-strain graph. The same effect is also evident during the unloading phase, wherein the stresses decrease to negative yield stress value and become constant thereafter. It can also be seen in the stress-strain graph shown in Figure 2 that the value of Young's modulus or the stiffness of the material determines the slope of the curve during both the loading and unloading phase affecting the rate of increase of the stresses to reach the yield value faster.

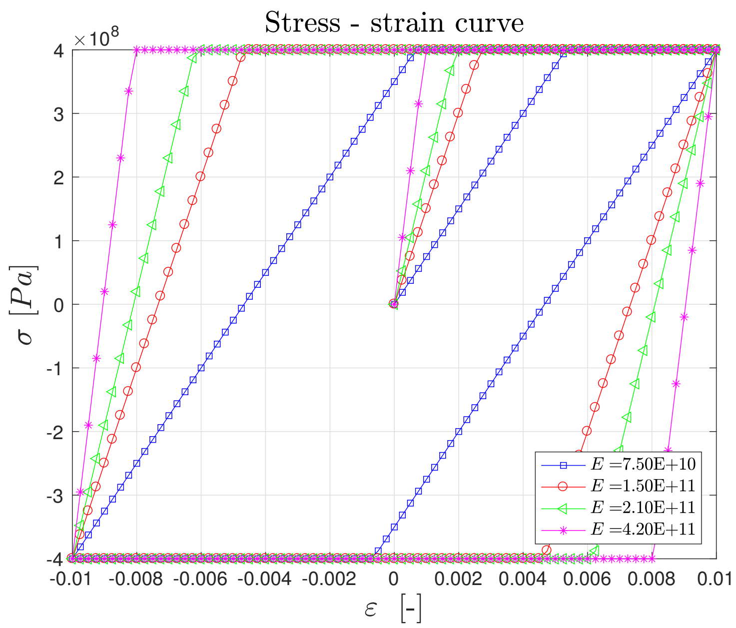
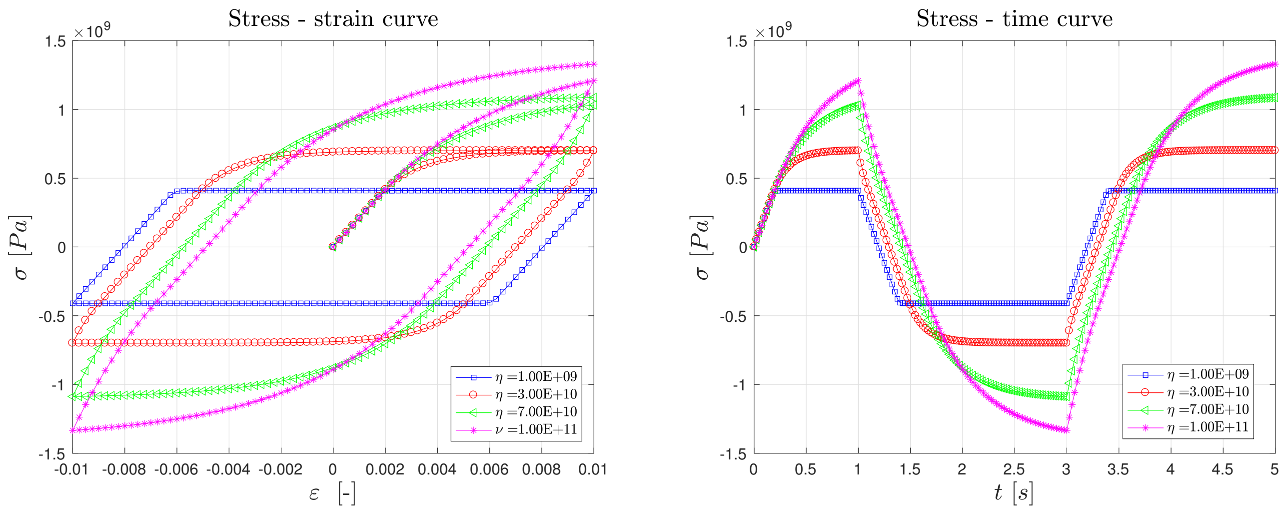



**Figure 2:** Rate-independent perfect plasticity: Stress-strain curve for varying Young's modulus.

## 2.2 Rate-dependent model

Now, we analyse the effect of viscosity of the material, $\eta$ in the rate-dependent model. In this model, since the elastoplastic tangent operator changes due to viscosity, we notice the increase in stresses above the yield value during both loading and unloading phases. Figure 3(a) shows the stress-strain graph for this model as a function of the viscosity of the material wherein a higher slope is observed with increasing value of viscosity without any effect on the yield surface. For this model, we also study the behaviour of stresses with time. In the stress-time curve shown in Figure 3(b), we observe the symmetry of the response in both tension and compression with constant stress value in the elastoplastic region.




(a) Stress-strain curve (b) Stress-time curve

**Figure 3:** Rate-dependent perfect plasticity model with different values of viscous coefficient.

# 3 Linear isotropic hardening plasticity

## 3.1 Rate-independent model

In this section, we analyse the behaviour of the linear isotropic hardening model with varying isotropic hardening modulus, $K$. In this model, an expansion of the elastic region could be noticed in the stress-strain graph shown in Figure 4. With increasing hardening modulus, the slope of the curve increases and the elastic region expands. This is in order to keep the rate-independent model a reasonable process as per the internal variable, $q$. On overcoming the yield stress, the relation between stress and strain depends on the elastoplastic tangent operator.

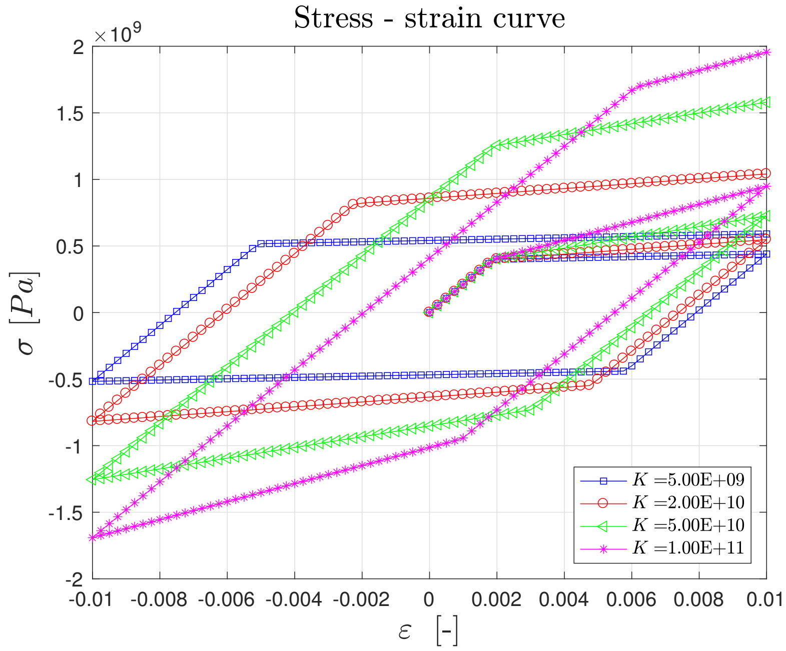
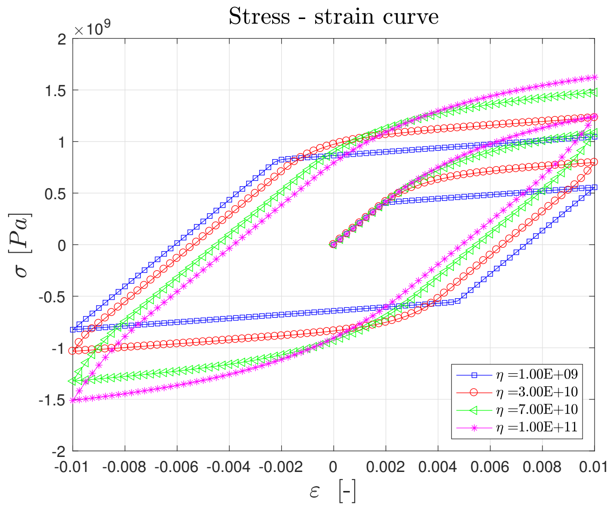


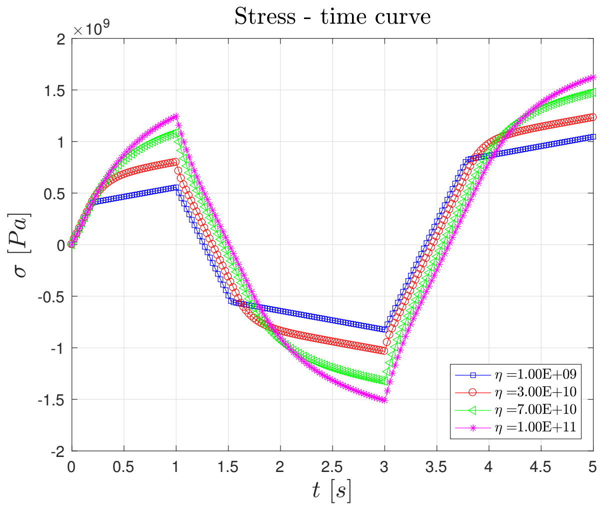

**Figure 4:** Rate-independent linear isotropic hardening plasticity model: Stress-strain curve for different values of isotropic hardening modulus.

## 3.2 Rate-dependent model

Now, we analyse the effect of viscosity of the material, $\eta$ in the rate-dependent model. Figure 5(a) shows the stress-strain graph for this model as a function of the viscosity of the material wherein a higher slope and also higher stresses are observed with increasing value of viscosity without any effect on the yield surface. This is because the hardening modulus is kept constant for all the cases performed in the analysis. For this model, we also consider the behaviour of stresses with time. In the stress-time curve shown in Figure 5(b), we observe that compared to the perfect plasticity model the stresses do not remain constant and increase with the expansion of the domain.




(a) Stress-strain curve




(b) Stress-time curve

**Figure 5:** Rate-dependent linear isotropic hardening plasticity model with different values of viscous coefficient.

# 4 Nonlinear isotropic hardening plasticity considering an exponential saturation law

## 4.1 Rate-independent model

In this section, we analyse the behaviour of the nonlinear isotropic hardening model with varying exponential saturation parameter, $\delta$. The exponential part can be seen in Figure 6 when the material surpasses the yield stress. The curve also tends to be asymptotic independent of the exponential saturation parameter value. Although higher stresses cannot be achieved once the asymptotic value is reached, increase in the exponential saturation parameter makes this process faster as it controls the expansion of the yield surface.

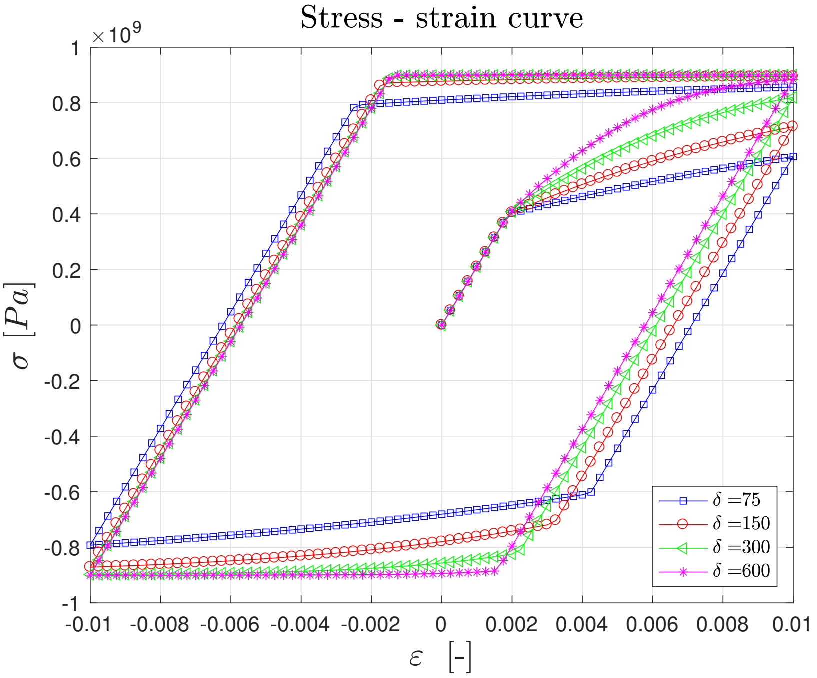
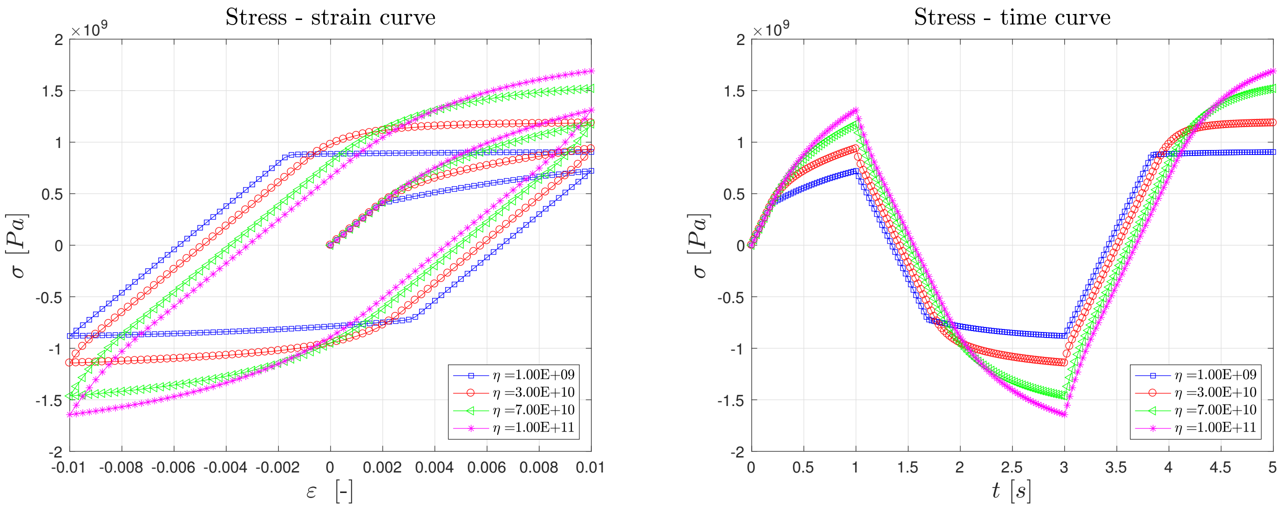



**Figure 6:** Rate-independent nonlinear isotropic hardening plasticity model: Stress-strain curve for different values of exponential saturation parameter.

## 4.2 Rate-dependent model

Now, we analyse the effect of viscosity of the material, $\eta$ in the rate-dependent model. Figure 7(a) shows the stress-strain graph for this model as a function of the viscosity of the material wherein we observe that the material is able to overcome the asymptotic value since the rate-dependent model enables the material to be present outside the elastic domain. The effect of increasing the viscous coefficient is seen as higher stresses are observed in the analysis. For this model, we now examine the behaviour of stresses with time. In the stress-time curve shown in Figure 7(b), it is noted that the yield surface expands exponentially compared to the linear isotropic hardening model.




(a) Stress-strain curve (b) Stress-time curve

**Figure 7:** Rate-dependent nonlinear isotropic hardening plasticity model with different values of viscous coefficient.

# 5 Linear kinematic hardening plasticity

## 5.1 Rate-independent model

In this section, we analyse the behaviour of the linear kinematic hardening model with varying kinematic hardening modulus, $H$. In this model, there is no expansion of the elastic region as seen in the stress-strain graph shown in Figure 8. With increasing hardening modulus, the slope of the curve increases and the elastic region translates as per the internal variable, $q$. It is interesting to note that with this translation, with higher kinematic hardening the compressive plastic loading occurs ahead of the other cases.

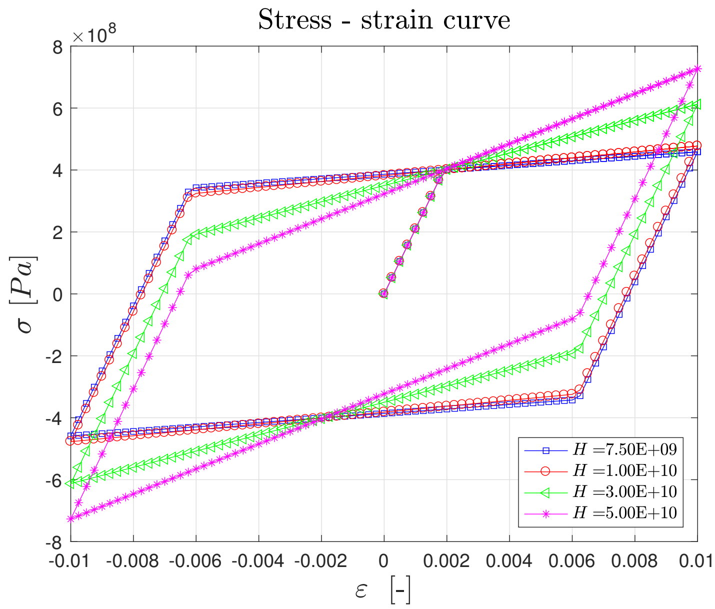
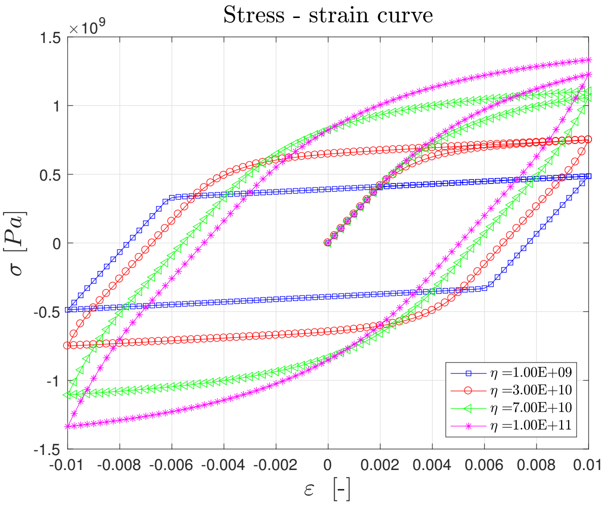


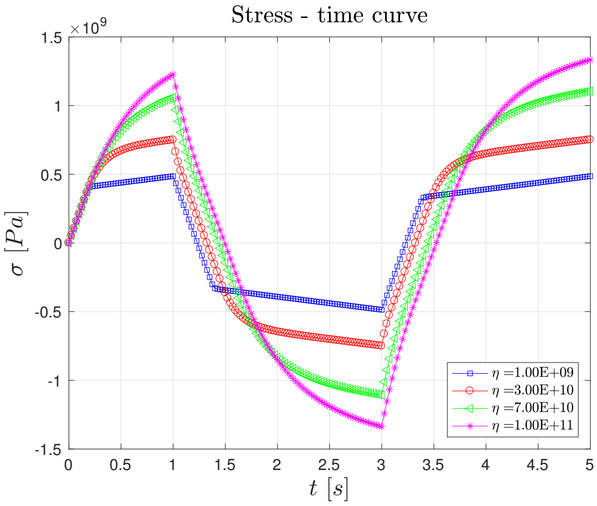

**Figure 8:** Rate-independent linear kinematic hardening plasticity model: Stress-strain curve for different values of kinematic hardening modulus.

## 5.2 Rate-dependent model

Now, we analyse the effect of viscosity of the material, $\eta$ in the rate-dependent model. Figure 9(a) shows the stress-strain graph for this model as a function of the viscosity of the material wherein we observe a closed curve since there is no expansion of the yield surface. Also, with increasing viscous coefficient, higher stresses are observed in the analysis. For this model, now we analyse the behaviour of stresses with time. In the stress-time curve shown in Figure 9(b), we observe the linear increment due to the translation effect discussed above.




(a) Stress-strain curve




(b) Stress-time curve

**Figure 9:** Rate-dependent linear kinematic hardening plasticity model with different values of viscous coefficient.

# 6 Nonlinear isotropic and linear kinematic hardening plasticity

## 6.1 Rate-independent model

In this section, an important and interesting analysis is performed to understand the behaviour of the material by combining the two models discussed in the previous sections i.e. the nonlinear isotropic and linear kinematic hardening models. To this end, Figure 10 shows the stress-strain graph for the rate-independent model wherein the effect of including both the models could be observed clearly. Firstly, due to the inclusion of isotropic hardening, expansion of the yield surface is possible. Secondly, the insertion of kinematic hardening results in losing the symmetry and therefore the asymptotic value would not be achieved in this case.

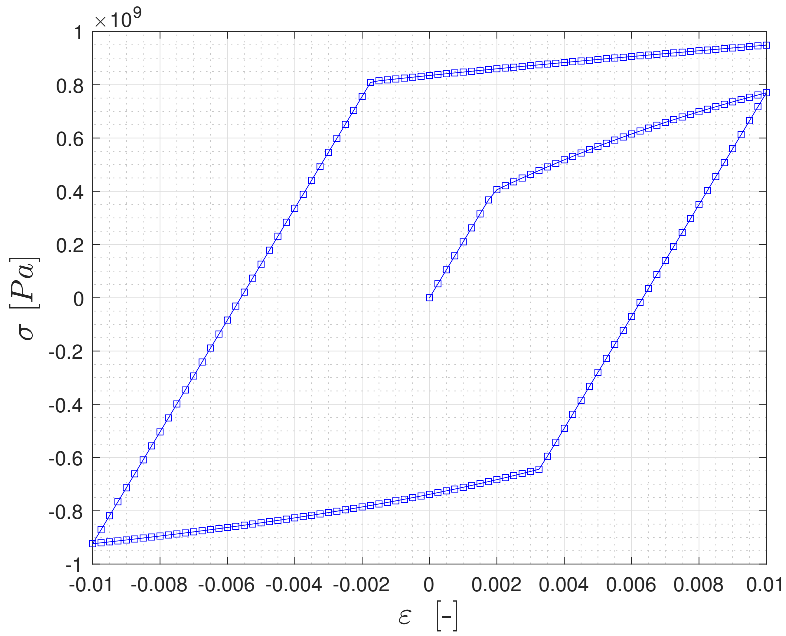
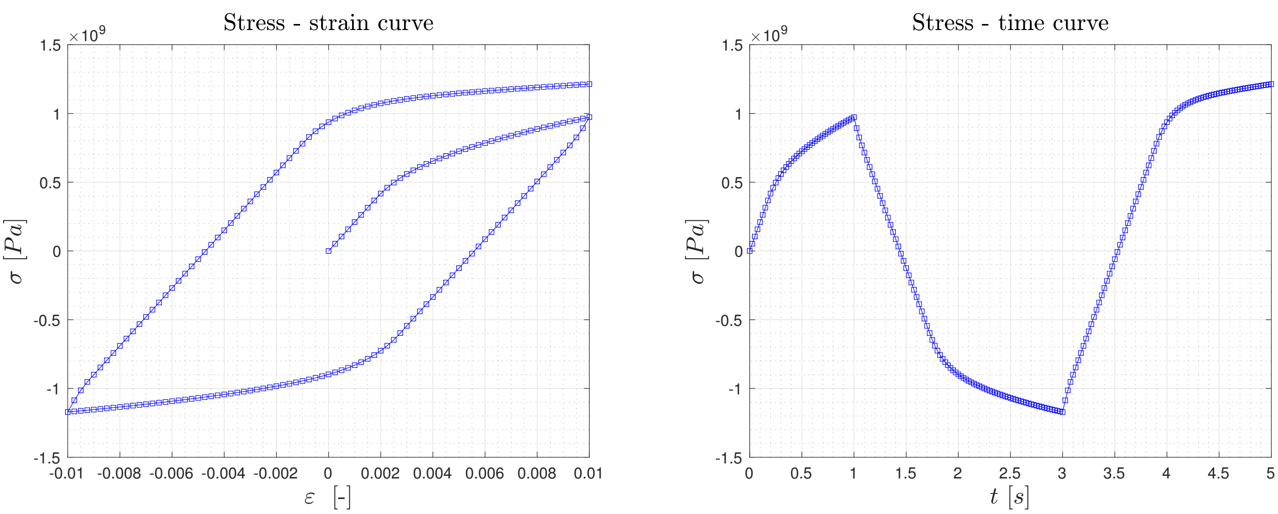



**Figure 10:** Rate-independent nonlinear isotropic and linear kinematic hardening plasticity model: Stress-strain curve.

## 6.2 Rate-dependent model

In case of the rate-dependent model, Figure 11(a) shows the stress-strain graph wherein the viscous coefficient just adds a regular shift between the two regions as also observed in all the earlier cases and exhibits identical effects as discussed in the rate-independent model. For this model, we also looked at the behaviour of stress with time. In the stress-time curve shown in Figure 11(b), we observe the effect of including both isotropic and kinematic hardening models as noticed above.



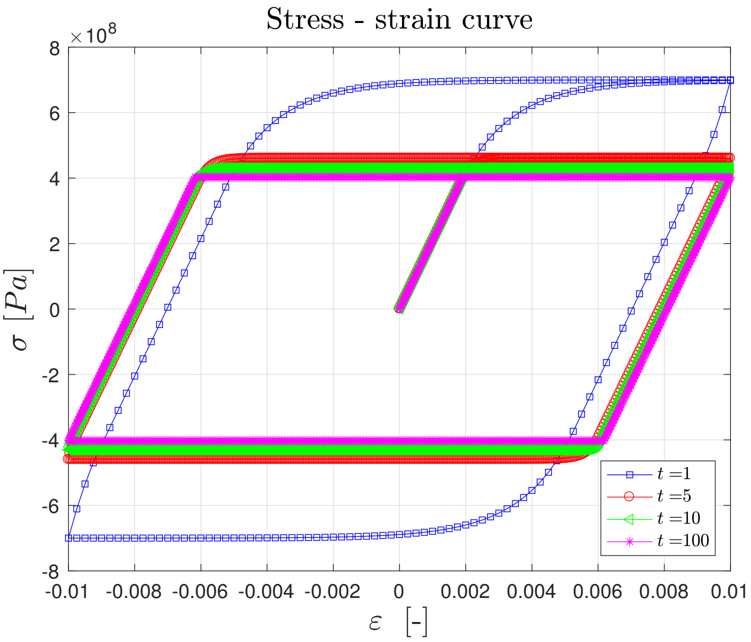

(a) Stress-strain curve

(b) Stress-time curve

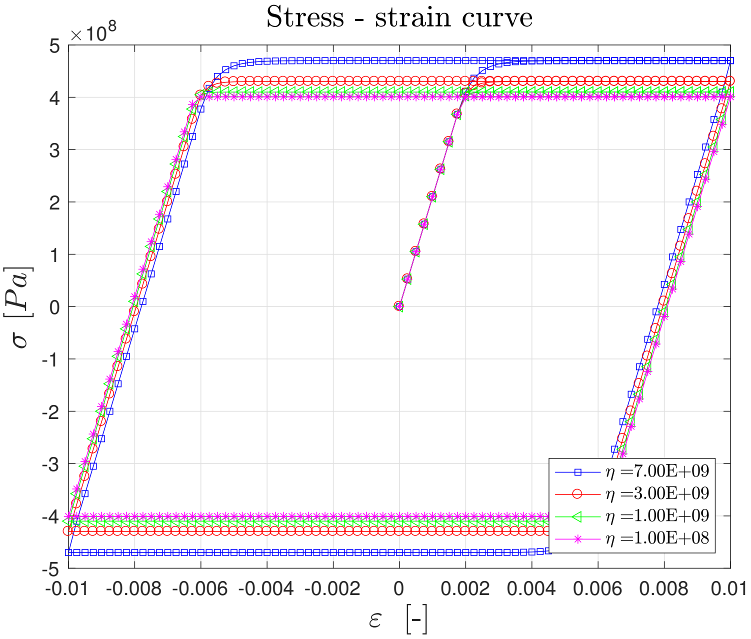
**Figure 11:** Rate-dependent nonlinear isotropic and linear kinematic hardening plasticity model with different values of viscous coefficient.

# 7 Restoration of the rate-independent behaviour from rate dependent model

In this section, we aim to restore the rate-independent behaviour of the model from a rate-dependent case. This could be achieved by two simple approaches. Firstly, increasing the total time of the simulation would essentially decrease the loading rate and therefore result in recovering a rate-independent model. Another approach is to decrease the viscous coefficient in our rate-dependent analysis, which would ideally mean, that we simulate the rate-independent case. Both these approaches are used to validate our understanding and are shown in Figure 12 where




(a) Plot with varying total simulation time




(b) Plot with varying viscous coefficient

**Figure 12:** Rate-dependent perfect plasticity model with varying parameters: Stress-time curve

the perfect plasticity model is used to demonstrate this effect. Figure 12(a) shows the effect of increasing the total time of the simulation till the results become independent of the loading rate whereas, in Figure 12(b), the viscosity of the material is reduced till the rate-independent effect is observed. Both the results provide the same effect and validate our understanding and the implementation in MATLAB.

# 8 Conclusion

In this work, the BE time-stepping algorithm for 1D rate-independent/dependent hardening plasticity models, including linear & nonlinear isotropic hardening and linear kinematic hardening is implement in MATLAB. Multiple numerical simulations are performed with the presented material properties and data of the cyclic loading. The post-processed results i.e. stress-strain graphs and the stress-time graphs for the rate-dependent plasticity models are presented to analyse the behaviour of the material with varying material parameters. The implementation is finally validated by showing that the rate-independent behaviour could be recovered from the rate-dependent model under certain circumstances.

# 9 Appendix

## main_1D_plasticity.m

```

%============================================================================
%============================================================================
% Program for 1D Plasticity - By: Nikhil Dave
% Computational Solid Mechanics - MSc. Computational Mechanics
% Universitat Politecnica de Catalunya (Barcelona Tech)
%============================================================================
%============================================================================
% Clear screen, workspace, close open figures
clear;
close all;
clc;
%============================================================================
% Input parameters
%============================================================================
   % Material properties
   Mat_Prop.E = suggest_para('Specify youngs modulus, E [Pa]:',2.1e11);
   fprintf(' \n ')
   Mat_Prop.sigma_y = suggest_para('Specify yield stress, \sigma_y [Pa]:',4e8);

   % Various models to be analysed
   fprintf(' \n ')
   fprintf(' \n ')
   disp('(1): Analyse perfect plasticity.')
   disp(' (2): Analyse isotropic hardening plasticity.')
   disp(' (3): Analyse kinematic hardening plasticity.')
   disp(' (4): Analyse isotropic and Kinematic hardening plasticity.')
   plastic_mod = suggest_para('Which model to be analysed?:',1);

   % Specify rate dependency
   fprintf(' \n ')
   Rate = input('Include rate-dependency? [Y/N]:','s');
   if Rate == 'Y'
   fprintf(' \n ')
   Mat_Prop.visc = suggest_para('Specify the viscous coefficient [Pa*s]:',3e10);
   else
   Mat_Prop.visc=0; % zero for rate-independent case
   end

   % Models with hardening
   switch plastic_mod
   case 2 % Isotropic hardening
       fprintf(' \n ')
       disp('You are analysing the isotropic hardening plasticity model.');
       Hardening = 'Y';
       fprintf(' \n ')
```

```
        Mat_Prop.K = suggest_para('Specify isotropic hardening modulus, K [Pa]:',2e10);
        Mat_Prop.H = 0;
    case 3 % Kinematic hardening
        fprintf(' \n ')
        disp('You are analysing the kinematic hardening plasticity model.');
        Hardening = 'Y';
        Isotropic_Hardening = 'None';
        fprintf(' \n ')
        Mat_Prop.H = suggest_para('Specify kinematic hardening modulus ,H [Pa]:',1e10);
        Mat_Prop.K = 0;
    case 4 % Isotropic and Kinematic hardening
        fprintf(' \n ')
        disp('You are analysing the isotropic and kinematic hardening plasticity model.');
        Hardening = 'Y';
        fprintf(' \n ')
        Mat_Prop.K = suggest_para('Specify isotropic hardening modulus, K [Pa]:',2e10);
        fprintf(' \n ')
        Mat_Prop.H = suggest_para('Specify kinematic hardening modulus, H [Pa]:',1e10);
    otherwise % Perfect plasticity
        fprintf(' \n ')
        disp('You are analysing the perfect plasticity model');
        Hardening = 'N';
        Isotropic_Hardening = 'None';
        Mat_Prop.K = 0;
        Mat_Prop.H = 0;
    end

    % Including isotropic hardening type
    if plastic_mod == 2 || plastic_mod == 4
     fprintf(' \n ')
     disp('(1): Analyse linear isotropic hardening plasticity.')
     disp(' (2): Analyse nonlinear isotropic hardening plasticity considering exponential saturation law.')
     isotropic_hardening = suggest_para('Specify isotropic hardening type:',1);
    if isotropic_hardening == 2
        Isotropic_Hardening = 'Exp';
        fprintf(' \n ')
        Mat_Prop.sigma_inf = suggest_para('Specify asymptotic maximum stress, sigma_inf [Pa]:',9e8);
        fprintf(' \n ')
        Mat_Prop.delta = suggest_para('Specify exponential saturation parameter, delta:',150);
    else
           Isotropic_Hardening = 'Linear';
    end
    end

```

```
    % Total simulation time and step size
    fprintf(' \n ')
    tot_time = suggest_para('Specify total simulation time for each loadstate [s]:',1)
        ;
    fprintf(' \n ')
    time_step = suggest_para('Specify time step size [s]:',0.025);

%==========================================================================
% Processing
%==========================================================================
    no_of_loadstates = 5;
    eps_vector = zeros(no_of_loadstates,1);
    eps_vector(1) = 0.0;
    eps_vector(2) = 0.01;
    eps_vector(3) = 0.0;
    eps_vector(4) = -0.01;
    eps_vector(5) = 0.0;
    eps_vector(6) = 0.01;
    strain = zeros(no_of_loadstates*tot_time/time_step,1);
    for ii = 2:(tot_time/time_step)+1
     strain(ii) = (eps_vector(2)/(tot_time/time_step))*(ii-1);
     strain(ii+tot_time/time_step) = eps_vector(2)+((eps_vector(3)...
                            -eps_vector(2))/(tot_time/time_step))*(ii-1);
     strain(ii+2*(tot_time/time_step)) = eps_vector(3)+((eps_vector(4)...
                            -eps_vector(3))/(tot_time/time_step))*(ii-1);
     strain(ii+3*(tot_time/time_step)) = eps_vector(4)+((eps_vector(5)...
                            -eps_vector(4))/(tot_time/time_step))*(ii-1);
     strain(ii+4*(tot_time/time_step)) = eps_vector(5)+((eps_vector(6)...
                            -eps_vector(5))/(tot_time/time_step))*(ii-1);
    end
    time = zeros((no_of_loadstates)*(tot_time/time_step),1);
    for k = 1:(no_of_loadstates*tot_time/time_step)
     time(k) = k*time_step;
    end

     % Initialising
    chi = zeros(length(strain),1); % isotropic strain variable
    chi_dash = zeros(length(strain),1); % kinematic strain variable
    eps_pl = zeros(length(strain),1); % plastic strain
    gamma = zeros(length(strain),1); % plastic multiplier
    stress = zeros(length(strain),1);
    q = zeros(length(strain),1);
    q_dash = zeros(length(strain),1);

    % get yield function and trial state values
    for i = 2:length(strain)
     [try_f,trystate] = trystatefn(Mat_Prop,chi(i-1),chi_dash(i-1),...
                              eps_pl(i-1),strain(i),Isotropic_Hardening);
     if try_f <= 0
         eps_pl(i) = trystate.eps_pl;
         chi(i) = trystate.chi;
         chi_dash(i) = trystate.chi_dash;
```

```
        stress(i) = trystate.stress;
        q(i) = trystate.q ;
        q_dash(i) = trystate.q_dash;
        E_epl = Mat_Prop.E; % elastoplastic tangent modulus
    else

        % linear or no isotropic hardening
        if strcmp(Isotropic_Hardening ,'Linear') == 1 || strcmp(Isotropic_Hardening ,
            'None')==1
            gamma = try_f/((Mat_Prop.E+Mat_Prop.K+Mat_Prop.H+Mat_Prop.visc/time_step)
                *time_step);
            [E_epl,Upd] = Plastic_upd_fn_linear (gamma,Mat_Prop.E,Mat_Prop.H,Mat_Prop
                .K,...
                                                            trystate,Mat_Prop.visc,
                                                                time_step);

        % nonlinear isotropic hardening
        else
            gamma = NRmethod (try_f,Mat_Prop.visc,Mat_Prop.E,Mat_Prop.H,Mat_Prop.
                sigma_y,...
                                    Mat_Prop.sigma_inf,Mat_Prop.delta,trystate.chi,
                                        time_step);
            [E_epl, Upd] = Plastic_upd_fn_nonlinear (gamma,Mat_Prop.E,Mat_Prop.H,...
            Mat_Prop.sigma_inf,Mat_Prop.sigma_y,trystate,time_step,Mat_Prop.visc,
                Mat_Prop.delta);
        end

        % Update
        eps_pl(i) = Upd.eps_pl;
        chi(i) = Upd.chi;
        chi_dash(i) = Upd.chi_dash;
        stress(i) = Upd.stress;
        gamma(i) = gamma;
        q(i) = Upd.q;
        q_dash (i) = Upd.q_dash;
    end
    end

%===========================================================================
% Post-processing
%===========================================================================

% stress-strain graph
figure(1)
plot(strain,stress,'bs-');
hold on
grid on
grid minor
set(gca,'FontSize',12)
xlabel('$\varepsilon$ \ [-]','Interpreter','LaTex','FontSize',20)
ylabel('$\sigma \ [Pa]$','Interpreter','LaTex','FontSize',20)
legend(['E = ' num2str(Mat_Prop.E,'%1.2E')],'Location','southeast')
```

```
title('Stress - strain curve','Interpreter','LaTex','FontSize',20)

% stress-time graph
if Mat_Prop.visc ~= 0
figure (2)
plot([0;time],stress,'bs-')
hold on
grid on
grid minor
set(gca,'FontSize',12)
xlabel('$t \ [s]$','Interpreter','LaTex','FontSize',20)
ylabel('$\sigma \ [Pa]$','Interpreter','LaTex','FontSize',20)
legend(['\eta = ' num2str(Mat_Prop.visc,'%1.2E')],'Location','southeast')
title('Stress - time curve','Interpreter','LaTex','FontSize',20)
end
```

## suggest_para.m

```

function Result = suggest_para(text,default)
%==========================================================================
% para_in suggests an input parameter to the user
%==========================================================================
prompt = [text '(suggested value ' num2str(default) ') = '];
Result = input(prompt);
if isempty(Result)
    Result = default;
end
```

## trystatefn.m

```

function [try_f , trystate] = trystatefn(Mat_Prop,chi,chi_dash,eps_pl,...
                                            strain,Isotropic_Hardening)
%==========================================================================
% trystate1 computes variables for trial state
%==========================================================================

% Input
chi_try = chi;
chi_dash_try = chi_dash;
eps_pl_try = eps_pl;
eps_i = strain;
stress_try = Mat_Prop.E*(eps_i - eps_pl_try);

% Linear or no isotropic hardening
if strcmp(Isotropic_Hardening ,'Linear') == 1 || strcmp(Isotropic_Hardening ,'None')
     == 1
```

```
q_try = - Mat_Prop.K * chi_try ;
% Nonlinear isotropic hardening
elseif strcmp(Isotropic_Hardening ,'Exp') == 1
q_try = (Mat_Prop.sigma_y - Mat_Prop.sigma_inf)*(1-exp(-Mat_Prop.delta*chi_try));
end

% Output
q_dash_try = -Mat_Prop.H*chi_dash_try;
try_f = abs(stress_try - q_dash_try) - Mat_Prop.sigma_y +q_try;
trystate.eps_pl = eps_pl_try;
trystate.chi = chi_try;
trystate.chi_dash = chi_dash_try;
trystate.stress = stress_try;
trystate.q = q_try;
trystate.q_dash = q_dash_try;
end
```

## plastic_upd_fn_linear.m

```

function [E_epl, Upd] = Plastic_upd_fn_linear (gamma,E,H,K,trystate,visc,time_step)
%==========================================================================
% Plastic_upd_fn_linear finds elastoplastic tangent modulus and updated
% plastic values for linear case
%==========================================================================

% Input
eps_pl_try = trystate.eps_pl;
chi_try = trystate.chi;
chi_dash_try = trystate.chi_dash;
stress_try = trystate.stress;
q_try = trystate.q;
q_dash_try = trystate.q_dash;

% Output
Upd.eps_pl = eps_pl_try + gamma*time_step*sign(stress_try-q_dash_try);
Upd.chi = chi_try + gamma*time_step;
Upd.chi_dash = chi_dash_try-gamma*time_step*sign(stress_try-q_dash_try);
Upd.stress = stress_try - gamma*time_step*E*sign(stress_try-q_dash_try);
Upd.q = q_try - gamma*time_step*K;
Upd.q_dash = q_dash_try + gamma*time_step*H*sign(stress_try-q_dash_try);
E_epl = E*(1-E/(E+K+H+visc/time_step));
end
```

### plastic_upd_fn_nonlinear.m

```
function [E_epl, Upd] = Plastic_upd_fn_nonlinear (gamma,E, H,sigma_inf,...
                              sigma_y, trystate, time_step,visc, delta)
%==========================================================================
% Plastic_upd_fn_nonlinear finds elastoplastic tangent modulus and updated
% plastic values for nonlinear case
%==========================================================================

% Input
eps_pl_try = trystate.eps_pl;
chi_try = trystate.chi;
chi_dash_try = trystate.chi_dash;
stress_try = trystate.stress;
q_dash_try = trystate.q_dash;

% Output
Upd.eps_pl = eps_pl_try + gamma*time_step*sign(stress_try-q_dash_try);
Upd.chi = chi_try + gamma*time_step;
Upd.chi_dash = chi_dash_try - gamma*time_step*sign(stress_try-q_dash_try);
Upd.stress = stress_try - gamma*time_step*E*sign(stress_try-q_dash_try);
Upd.q = (sigma_y - sigma_inf)*(1-exp(-delta*(chi_try + gamma*time_step)));
Upd.q_dash = q_dash_try + gamma*time_step*H*sign(stress_try-q_dash_try);
E_epl = E*(1-E/(E+(sigma_inf-sigma_y)*delta*exp(-delta*(chi_try+gamma*...
                                    time_step))+ H + visc/time_step));
end
```

### NRmethod.m

```
function gamma = NRmethod (try_f,visc,E,H,sigma_y,sigma_inf,delta,chi,time_step)
%==========================================================================
% NRmethod is the Newton-Raphson method for solving nonlinear problems
%==========================================================================

tol = 1e-6; % convergence tolerance
maxit = 10; % maximum iterations
jj = 0; % initialise counter
gamma = 0; % initialise gamma

% calculate residual
residual = try_f - gamma*time_step*(E+H+visc/time_step)-...
    (sigma_inf - sigma_y)*(1 - exp(-delta*(chi + gamma*time_step)))+...
                          (sigma_inf - sigma_y)*(1 - exp(-delta * chi));
% while loop with tolerance
while abs(residual) > tol && jj < maxit
  dgamma = -time_step*(E+H+visc/time_step)-(sigma_inf-sigma_y)*delta*...
                            time_step*exp(-delta*(chi+gamma*time_step));
  del_gamma = -(1/dgamma)*residual;
```

```
  % update gamma and residual for next loop
  gamma = gamma + del_gamma;
  residual = try_f-gamma*time_step*(E+H+visc/time_step)-(sigma_inf-...
      sigma_y)*(1-exp(-delta*(chi+gamma*time_step)))+(sigma_inf-sigma_y)...
                                                      *(1-exp(-delta*chi));
jj=jj+1; % counter update
end
end
```