Finite elastic-plastic deformations (BMEGEMMDKPL)
II. Homework

Szász Zsolt
KRCH5Q

May 3, 2025

Loading

We have an L mm $\times$ L mm $\times$ L mm brick element ($L = 1$ mm), which node numbering is shown on Figure 1.

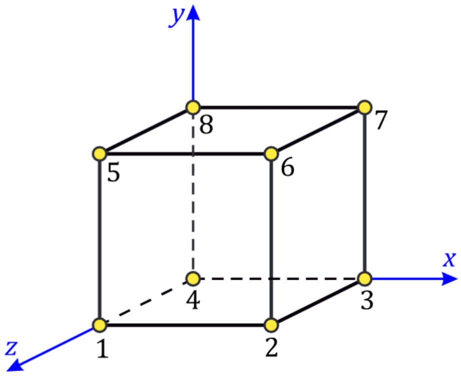

Figure 1: Nodal layout of the brick element.

We have a prescribed displacements on the upper nodes, defined by displacement vector as

$$[\boldsymbol{U_1}] = [\boldsymbol{U_2}] = [\boldsymbol{U_3}] = [\boldsymbol{U_4}] = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T, \tag{1}$$

$$[\boldsymbol{U_5}] = [\boldsymbol{U_6}] = [\boldsymbol{U_7}] = [\boldsymbol{U_8}] = \begin{bmatrix} u_x & u_y & 0 \end{bmatrix}^T, \tag{2}$$

where the time evolution of u_x and u_y and their time derivatives are shown on Figure 2.

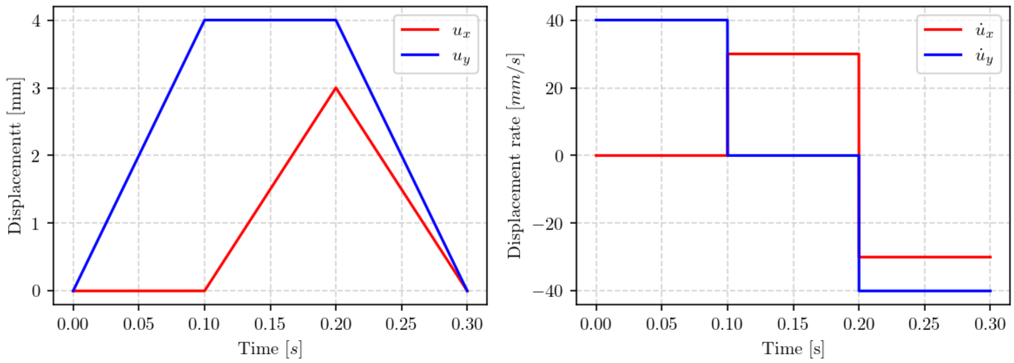

Figure 2: The parameters of the loading

From the displacement vectors we can determine the deformations gradient regarding to that motion.

$$[F] = \begin{bmatrix} 1 & u_x/L & 0 \\ 0 & 1 + u_y/L & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{3}$$

For the further calculations we will need the velocity gradiend, which can be calculated from the deformation gradient as

$$[\boldsymbol{l}] = [\dot{\boldsymbol{F}}\boldsymbol{F}^{-1}] = \begin{bmatrix} 0 & \dot{u}_x & 0 \\ 0 & \dot{u}_y & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & -\dfrac{u_x}{1+u_y} & 0 \\ 0 & \dfrac{1}{1+u_y} & 0 \\ 0 & 0 & 1 \end{bmatrix} = \frac{1}{1+u_y} \begin{bmatrix} 0 & \dot{u}_x & 0 \\ 0 & \dot{u}_y & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{4}$$

The rate of deformation ($\boldsymbol{d}$) is the symmetric and the spin tensor ($\boldsymbol{w}$) is the skew symmetric part of the velocity gradient:

$$\boldsymbol{d} = \frac{1}{2}(\boldsymbol{l} + \boldsymbol{l}^T) \quad \& \quad \boldsymbol{w} = \frac{1}{2}(\boldsymbol{l} - \boldsymbol{l}^T). \tag{5}$$

Material behaviour

This element has an elastic-plastic behaviour. We have finite deformations, therefore we need to use hypoelastic-plastic modelling approach, which is based on the additive decomposition of the rate of deformation tensor into elastic and plastic part

$$\boldsymbol{d} = \boldsymbol{d}^e + \boldsymbol{d}^p. \tag{6}$$

Following the ABAQUS formulation, the elastic behaviour can be defined by the Hooke's law as follows

$$\overset{\circ}{\boldsymbol{\sigma}} = \frac{E}{1+\nu}\left(\boldsymbol{d}^e + \frac{\nu}{1-2\nu}(\mathrm{tr}\boldsymbol{d}^e)\boldsymbol{I}\right) := \mathbb{D} : \boldsymbol{d}^e, \tag{7}$$

where the $\overset{\circ}{\boldsymbol{\sigma}}$ denotes the Jaumann rate of the Cauchy stress tensor. The discrete form of the Hooke's law in the rotated coordinate system looks like

$$\tilde{\boldsymbol{\sigma}}_{n+1} = \tilde{\boldsymbol{\sigma}}_n + \mathbb{D}\Delta\tilde{\boldsymbol{\varepsilon}}^e \tag{8}$$

To be able to rotate the quentities back and forth, we need to determine the $\boldsymbol{Q}$ rotation tensor in each step, which can be calculated from the $\boldsymbol{w}$ spin tensor as

$$\dot{\boldsymbol{Q}} = \boldsymbol{w}\boldsymbol{Q}, \tag{9}$$

where we can conclude that we will need time integration to determine the values. We will use the Hughes-Winget algorithm as it is required, which is defined as

$$\boldsymbol{Q}_{n+1} = \left(\boldsymbol{I} - \frac{1}{2}\boldsymbol{w}_{n+1/2}\Delta t\right)^{-1}\left(\boldsymbol{I} + \frac{1}{2}\boldsymbol{w}_{n+1/2}\Delta t\right)\boldsymbol{Q}_n. \tag{10}$$

The values of the elastic material parameters are the followings

$$E = 200 \text{ GPa} \quad \& \quad \nu = 0.3 \tag{11}$$

Furthermore the characteristic of the Yield stress is defined by the following Swift model:

$$Y(\lambda) = a(\varepsilon_0 + \lambda)^n \quad \text{,where} \quad a = 400 \text{ MPa}, \quad \varepsilon_0 = 0.05, \quad n = 0.25. \tag{12}$$

Solution

My implementation of the radial-retirn algorithm for this hypoelastic-plastic model in Python language is the following:

```
Q = np.tile(I, (N, 1, 1))
sig_tilde = np.tile(O, (N, 1, 1))
sig = np.tile(O, (N, 1, 1))
Y = np.ones(N) * Y_F(0)
_lambda = np.zeros(N)

for n in range(N-1):
    dt = t[n+1] - t[n]
    w_mid = (w[n+1] + w[n])/2
    d_mid = (d[n+1] + d[n])/2

    # Hughes-Winget
    Q[n+1] = inv(I - 1/2*w_mid*dt) @ (I + 1/2*w_mid*dt) @ Q[n]
    Q_mid = Q[n+1]

    deps_tilde = Q_mid.T @ d_mid* dt @ Q_mid

    # Step 3: trial stress
    sig_trial_tilde = sig_tilde[n] + D(deps_tilde)
    s_trial_tilde = dev(sig_trial_tilde)

    # Step 4: trial yield function
    F_trial = (3/2)**(1/2) * norm(s_trial_tilde) - Y[n]

    if F_trial <= 0:
        # Step 6: Elastic increment
        sig_tilde[n+1] = sig_trial_tilde
        Y[n+1] = Y[n]
        _lambda[n+1] = _lambda[n]
    else:
        # Step 7: Accumulated plastic strain increment
        def eq(d_lambda):
            ret  = (3/2)**(1/2) * norm(s_trial_tilde)
            ret -= 3*G*d_lambda
            ret -= Y_F(_lambda[n]+d_lambda)
            return ret
        d_lambda = (root(eq,1e-8).x)[0]

        # Step 8: Plastic strain increment and yield stress update
        deps_p_tilde = (3/2)**(1/2) * d_lambda * s_trial_tilde /
           norm(s_trial_tilde)
        _lambda[n+1] = _lambda[n] + d_lambda
        Y[n+1] = Y_F(_lambda[n+1])

        # Step 9: Update the rotated stress
        sig_tilde[n+1] = sig_trial_tilde - D(deps_p_tilde)

    # Step 10: Compute the stress tensor
    sig[n+1] = Q[n+1] @ sig_tilde[n+1] @ Q[n+1].T
```

5 increments per step

The values of the plotted quentities are also attached as a CSV file.

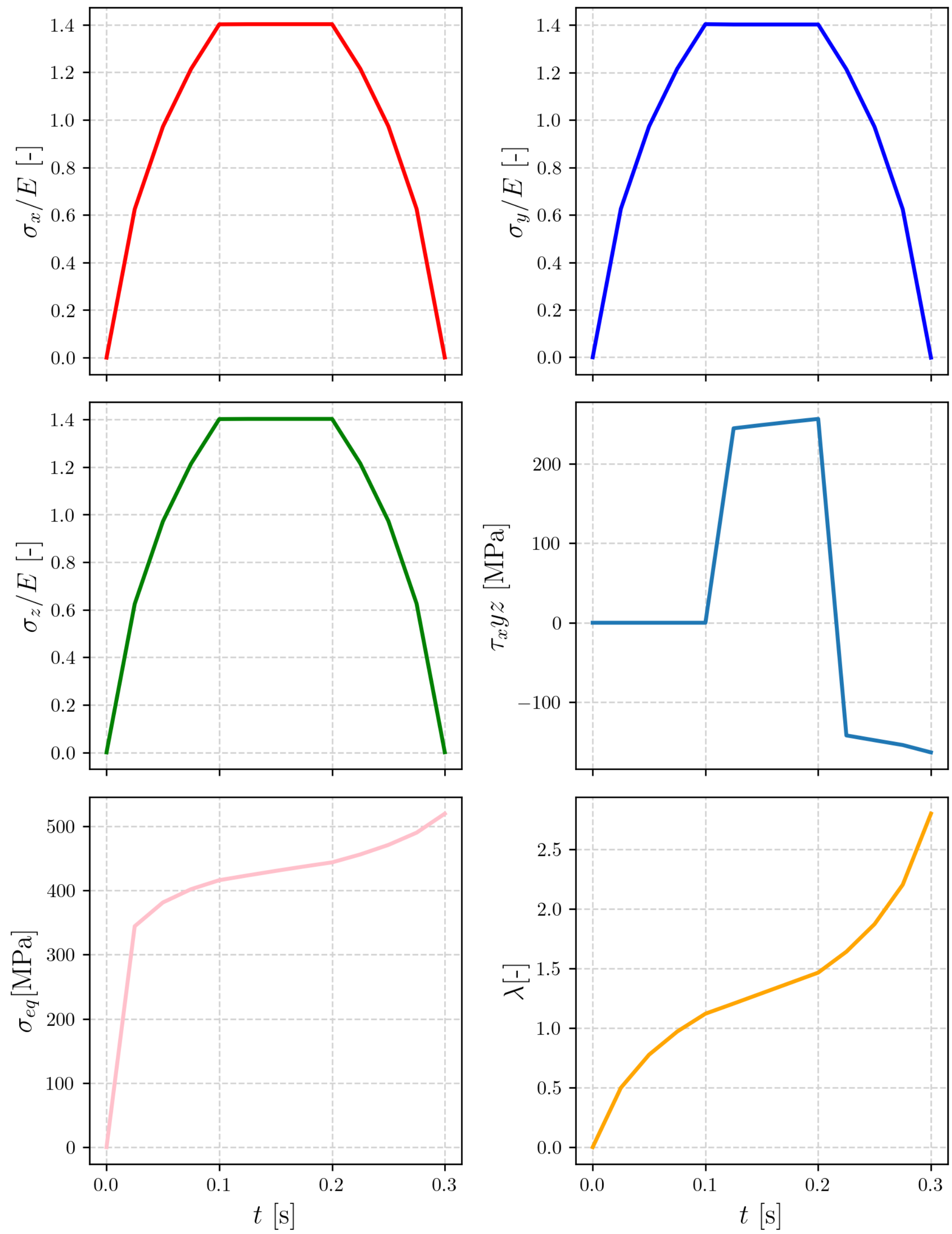

Figure 3: The solution of the problem, by applying 5 increments in each step.

1000 increments per step

The values of the plotted quentities are also attached as a CSV file.

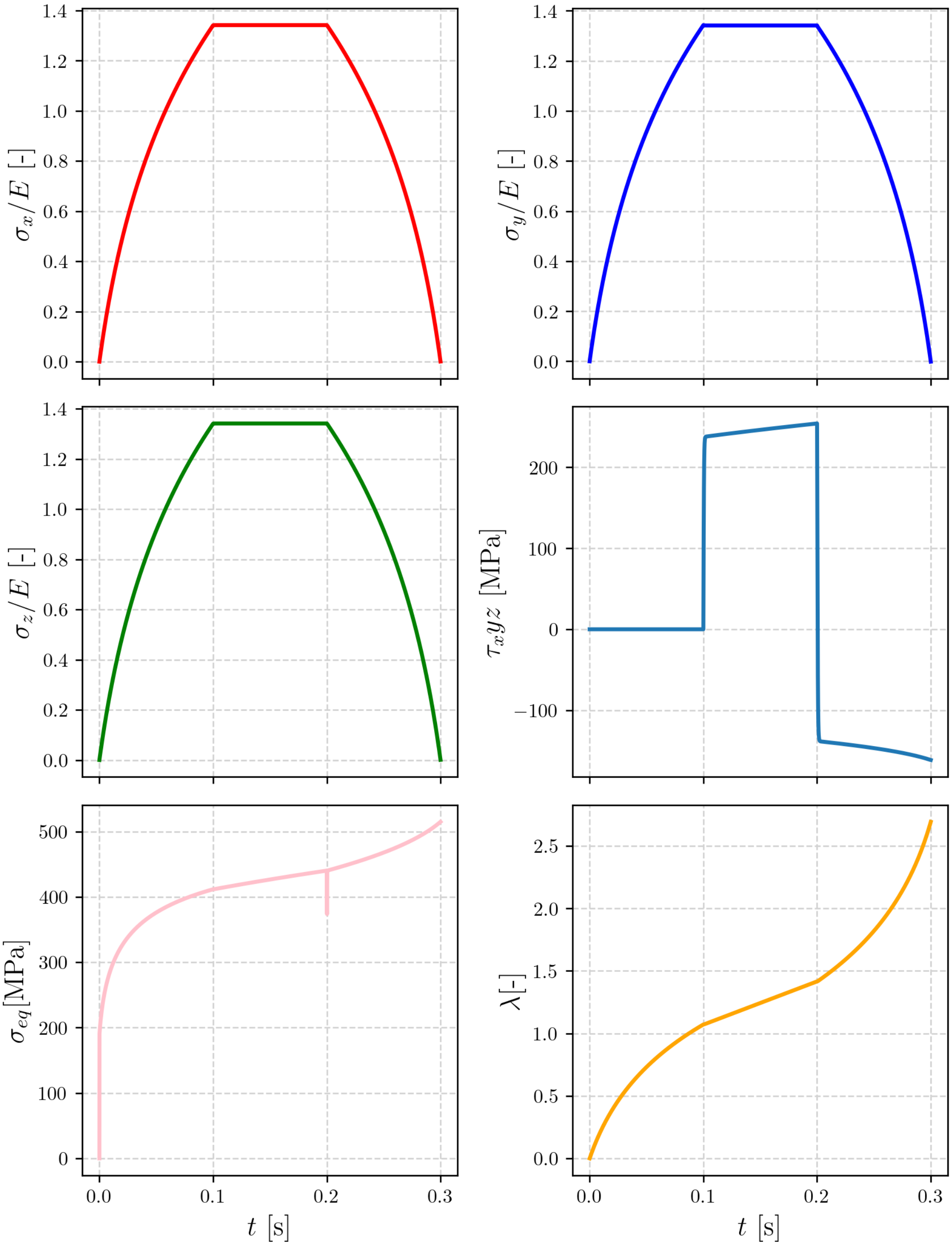

Figure 4: The solution of the problem, by applying 1000 increments in each step.

References

You can find the detailed code on GitHub.