TECHNISCHE UNIVERSITÄT
BERGAKADEMIE FREIBERG
The University of Resources. Since 1765.

Fakultät für Maschinenbau, Verfahrens- und Energietechnik
Institut für Mechanik und Fluiddynamik
Technische Mechanik - Festkörpermechanik

Assignment

Nonlinear finite element methods

Summer term 2020

Venkata Mukund Kashyap Yedunuthala

Computational Materials Science
Matriculation register: 65 163

07. July 2020

Contents

List of Figures

1 Overview

1.1 Introduction

The field of computer aided engineering is driven by numerical simulation tools for several kinds of investigations. Often, these investigations lead to equations that are of nonlinear nature. In this assignment, we focus on one such investigation, i.e. the creep of a thick-walled pipe under internal pressure p. The specifics of available data, and the subsequent investigation are detailed below.

1.2 Problem statement

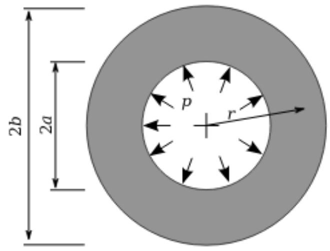

Fig. 1: Thick-walled pipe

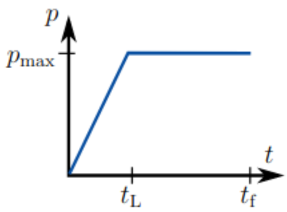

Fig. 2: Load sequence

Creep of thick-walled pipe, seen in Figure 1 under internal pressure p, that rises linearly until final value p_{max} which is held until t_f, as seen in Figure 2.

Plain strain conditions are considered, i.e. $\varepsilon_{zz} = 0$. The non-vanishing equilibrium condition, owing to axisymmetric conditions is as follows.

$$0 = \frac{\partial \left(r\sigma_{rr}\right)}{\partial r} - \sigma_{\phi\phi} \tag{1}$$

Here, σ_{rr} and $\sigma_{\phi\phi}$ are stress components with respect polar coordinate (r-ϕ) system. The weak form of the above equation is as follows.

$$0 \;=\; \partial W \;=\; \int_a^b \underline{\delta\varepsilon}^T . \underline{\sigma} r \mathrm{d}r - \left[r\sigma_{rr}\delta u_r\right]_{r=a}^b \tag{2}$$

The stresses and strains, written in Voigt notation are as follows.

$$\underline{\sigma} = \begin{bmatrix}\sigma_{rr}\\ \sigma_{\phi\phi}\end{bmatrix},\ \underline{\delta\varepsilon} = \begin{bmatrix}\delta\varepsilon_{rr} = \frac{\partial \delta u_r}{\partial r}\\ \delta\varepsilon_{\phi\phi} = \frac{\partial u_r}{r}\end{bmatrix},\ \text{analogously } \underline{\varepsilon} = \begin{bmatrix}\varepsilon_{rr} = \frac{\delta u_r}{\delta r}\\ \varepsilon_{\phi\phi} = \frac{u_r}{r}\end{bmatrix} \tag{3}$$

The boundary conditions for this problem are described as follows,

- At r = a, σ_{rr} = -p
- At r = b, σ_{rr} = 0

The linear visco-elastic behavior of the material given in this problem has been described by the following equations.

$$\underline{\sigma} = \underline{\underline{\mathbf{C}}} \cdot \underline{\varepsilon} + \underline{\sigma}^{ov} \tag{4a}$$

$$\underline{\dot{\sigma}}^{ov} \;=\; Q\mathrm{dev}\left(\underline{\dot{\varepsilon}}\right) - \frac{1}{T}\underline{\sigma}^{ov} \tag{4b}$$

Where, $\underline{\underline{\mathbf{C}}}$ is described as the isotropic (long-term) elastic stiffness matrix, expressed in terms of Young's modulus E and Poisson ration υ. The evolution of the overstress $\underline{\sigma}^{\mathrm{ov}}$ (as the internal stress variable) is described to be governed by Modulus Q and a characteristic time scale T.

1.3 Discretization of the weak form

The discretization of the weak form in space is highlighted as follows.

1.3.1 Strain displacement matrix

Using Galerkin method, u_r can be discretized as follows

$$u_r = [\mathbf{N}].\underline{\hat{u}} \tag{5}$$

Here, lagrangian polynomials, that are taken as the ansatz functions in this discretization, are given as follows,

$$[\mathbf{N}] = \left[\tfrac{1}{2}\left(1-\xi\right), \tfrac{1}{2}\left(1+\xi\right)\right] \text{ in } \Omega_B \; = \{\xi \in [-1, 1]\} \tag{6}$$

Then, the Jacobian for these values are calculated as follows,

$$J = \frac{\partial N}{\partial \xi}.\hat{\mathrm{r}} = \begin{bmatrix} -\frac{1}{2} & \frac{1}{2} \end{bmatrix} \begin{bmatrix} r1 \\ r2 \end{bmatrix}$$

The strain displacement matrix $\underline{\mathbf{B}}$ can then be calculated using the inverse of this Jacobian matrix, and $\frac{\partial N}{\partial \xi}$. The value of Jacobian in this case is a scalar, i.e.

$$J = \frac{h^{elem}}{2}$$

Upon calculating, the strain displacement matrix can be expressed as follows.

$$\underline{\mathbf{B}} = \begin{bmatrix} -\frac{1}{h^{elem}} & \frac{1}{h^{elem}} \\ \frac{(1-\xi)}{r_1^e(1-\xi)+r_2^e(1+\xi)} & \frac{(1+\xi)}{r_1^e(1-\xi)+r_2^e(1+\xi)} \end{bmatrix} \tag{7}$$

1.3.2 Discretization of weak form in space

Therefore, the discretization of the equation 3 can be performed as follows.

$$0 = \delta W = \int_a^b \delta \underline{\hat{u}}^T \underline{\mathbf{B}}^T \underline{\sigma} \mathrm{rdr} - \mathrm{pa}\delta \underline{u}$$

$$\delta W = \delta \underline{\hat{u}}^T \int_\Omega \underline{\mathbf{B}}^T \underline{\sigma}.\underline{\mathrm{N}}\mathrm{Jr}^e \mathrm{d}\xi - \mathrm{pa}\underline{\mathrm{x}} \tag{8}$$

Now, substituting equation 4a into equation 8 yields $\underline{\hat{\mathrm{F}}}_{int}$ as,

$$\underline{\hat{\mathrm{F}}}_{int} = \left(\int_\Omega \underline{\mathbf{B}}^T.\underline{\mathbf{C}}.\underline{\mathbf{B}}.\underline{\mathrm{N}}\mathrm{Jr}^e \mathrm{d}\xi \right) \underline{\hat{\mathrm{u}}}^e + \left(\int_\Omega \underline{\mathbf{B}}^T.\underline{\sigma}^{\mathrm{ov}}.\underline{\mathrm{N}}\mathrm{Jr}^e \mathrm{d}\xi \right) \underline{\hat{\mathrm{u}}}^e \tag{9}$$

This equation 9 is split into two parts and further descritized using Gauss quadrature and time-integration of evolution equation using Euler modified (EM) scheme as described below.

1.3.3 Gauss quadrature

The integral present in equation 9 is solved using Gauss quadrature scheme to further discretize. It is given that quadrature with 1 Gauss point per element must be used, and in this case we set the Gauss point at $\xi = 0$.
At ξ =0, we have weights $W_\alpha = 2$. Therefore $\underline{\mathrm{N}}$ resolves to $2\underline{\mathrm{N}}(\xi_o)$ at ξ_o=0.
It must be noted that at that point, $[\mathrm{N}] = \begin{bmatrix}\frac{1}{2} & \frac{1}{2}\end{bmatrix}$
Therefore, the element stiffness matrix can be rewritten as

$$\underline{\mathbf{K}}^e = 2\underline{\mathbf{B}}^T.\underline{\mathbf{C}}.\underline{\mathbf{B}}.\underline{\mathrm{N}}(\xi_o)\mathrm{Jr}_e \tag{10}$$

1.3.4 Material tangent matrix

The linear visco-elastic behavior of the material is described in the equations 4a, 4b. Using the *Euler Modified scheme*, the evolution of overstress is discretized temporally, in order to evaluate the internal nodal forces. The following part of the equation 9 is discretized using EM scheme.

$$\int_\Omega \underline{\mathbf{B}}^T.\underline{\sigma}^{\mathrm{ov}}.\underline{\mathrm{N}}\mathrm{Jr}^e\mathrm{d}\xi$$

Upon time-integration of evolution equation 4b, we obtain,

$$\underline{\sigma}^{\mathrm{ov}}_{m+1} = \underline{\sigma}^{\mathrm{ov}}_{m} + \mathrm{Q}\mathrm{dev}(\dot{\varepsilon})\Delta t - \frac{\Delta t}{2\mathrm{T}}\left[\underline{\sigma}^{\mathrm{ov}}_{m} + \underline{\sigma}^{\mathrm{ov}}_{m+1}\right]$$

Rearranging,

$$\underline{\sigma}^{\mathrm{ov}}_{m+1} = \left(\frac{1}{1+\frac{\Delta t}{2\mathrm{T}}}\right)\left[\left(-\frac{\Delta t}{2\mathrm{T}}\right)\underline{\sigma}^{\mathrm{ov}}_{m} + \mathrm{Q}\left(\Delta\underline{\varepsilon} - \frac{\mathrm{tr}(\Delta\underline{\varepsilon})}{3}\right)\right] \tag{11}$$

with the state law at the instance as,

$$\underline{\sigma}_{\mathrm{m+1}} = \underline{\mathbf{C}}.\underline{\varepsilon}_{\mathrm{m+1}} + \underline{\sigma}^{\mathrm{ov}}_{m+1}$$

The material tangent matrix is expressed as follows,

$$\underline{\mathbf{C}}_t = \underline{\mathbf{C}} + \frac{\mathrm{Q}}{1+\frac{\Delta t}{2T}}\frac{1}{3}\begin{bmatrix}2 & -1\\ -1 & 2\end{bmatrix} \tag{12}$$

1.4 Assembly

Further, the above descritized equation must be assembled in order to determine global $\underline{\hat{\mathrm{F}}_{int}}$ and stiffness matrix $\underline{\hat{\mathbf{K}}}$. Due to the non-linear nature of the equations present, a Newton-Raphson scheme is employed to obtain an incremental solution. The following set of equations provide an illustration of the method.

$$\underline{\mathbf{K}}_{\mathrm{m+1}} \approx \underline{\mathbf{K}}_{\mathrm{m}} + \frac{\partial\underline{\mathbf{K}}}{\partial\underline{\hat{\mathrm{u}}}}\Delta\underline{\hat{\mathrm{u}}} \tag{13}$$

$$\underline{\mathbf{K}}\Delta\underline{\hat{\mathrm{u}}} = \underline{\mathrm{F}}_{\mathrm{int}} - \underline{\mathrm{F}}_{\mathrm{ext}} \tag{14}$$

This ensures a load increment at equilibrium in that timestep.

2 Structure of the program

2.1 The codebase

The programming language used to employ the computational solvers is *Python*. The codebase consists of files with names that are easily identifiable.For example, the file *inputParams.py* consists of all the input parameters that are necessary for the functioning of the solver. The file named *meshGenerator.py* is used to generate the nodes and their positions based on the inputs given. This file also creates an image called *Generated_ mesh.png* that serves as a depiction of the positions of the nodes radially. The file *main.py* is the most important file in the codebase, as this file contains the solver along with the necessary element and material routines tailored for this specific case. The *analytical_ sol.py* file generates the exact solution of the considered boundary value problem, as described in Verification section of the problem statement. The file *convergenceStudy.py* specifically focuses upon the tasks performed during convergence study, this file generates plots that help study the distribution of displacement at last time step with respect to variation in number of elements and variation in time step.

2.2 User manual

The necessary prerequisite of this codebase is that all the files must be present in the *same directory*. There is however no specified order in which the files must be arranged.

2.2.1 Prerequisites

Owing to the open-source nature of Python programming language, several developers have released packages that aid the user in simplifying the implementation. Therefore, the user is expected to have the following libraries installed on their preferred development environment.

– NumPy - A library designed to support numerical computing using Python.

– Matplotlib - A library designed to aid generating different kinds of plots using Python.

– datetime - A built-in python library that handles manipulation of dates and times.

– JSON - A built-in python library that encodes and decodes .json files, required only for convergence study.

2.2.2 Workflow

The entire procedure can be summarized as follows:

– Update the initial values in the *inputParams.py* file as per requirement.

– Run the file *main.py*

The solver employed in *main.py* file is based on Newton-Raphson scheme with appropriate convergence criterion. Upon a successful execution of the program, the statement *"Convergence achieved in x seconds"* where x is the time taken to run the solver. The following files, that consist of plots, are generated upon completion of the program.

∗ `generatedMesh.png`: A PNG-file consisting of the positions of nodes taken in radial direction.

* `displacementDistribution.png`: A PNG-file consisting of the distribution of displacement u_r with respect to radii at the last time step taken in the solver.
* `stressDistribution.png`: A PNG-file consisting of the distribution of the stresses with respect to radii at the last time step taken in the solver.
* `displacementEvolution.png`: A PNG-file consisting of the evolution of displacement u_r at each time step at radius r = b.
* `comparison.png`: A PNG-file consisting of the comparison of the distribution of the analytically and computationally generated displacements with respect to radial positions.
* `convergenceStudy_ u_ nelem.png`: A PNG-file consisting of convergence study plot of displacements with respect to number of elements.
* `convergenceStudy_u_dt.png`: A PNG-file consisting of convergence study plot of displacements with respect to variation in time step.

JSON files consisting of variations in displacement are used to aid convergence study and have no bearing in the functioning of program. The files store the values every time the function is called, therefore user discretion is advised. It is recommended to delete these files every time a convergence study is performed.
The user is expected to have intermediate level of skillset specific to Python, in order to debug any issues that may arise due to improper formatting of the input variables. The user is also expected to understand the specifics of the theory behind this investigation in order to debug any symantical errors. The files generated upon the execution of the program are included in the sections below.

2.2.3 Description of variables

This section details the variables that are recommended to be altered by the user to suit their requirements. The specifics are as follows, with the values used in this investigation described in parantheses:

- `INNER_RADIUS` - Variable that describes the inner radius of the axisymmetric investigation in millimeters. (Taken as 60 mm)
- `OUTER_RADIUS` - Variable that describes the outer radius of the axisymmetric investigation in millimeters. (Taken as 120 mm)
- `E` - Variable that describes the young's modulus in MPa. (Taken as 70 000 MPa)
- `UPSILON` - Variable that describes the poisson ration (υ). (Taken as 0.30)
- `Q` - Variable that describes the modulus as used in equation 4b in MPa. (Taken as 35 000 MPa)
- `TIME_SCALE` - Variable that describes characteristic time scale as seen in equation 4b.(Taken as 3 s)
- `P_MAX` - Variable that describes final value of pressure p_{max} (Taken as 50 MPa)
- `START_TIME` - Variable used in setting initial point in time. (Taken as 0 s)

- `END_TIME` - Variable used in setting final point in time. (Taken as 30 s)
- `LOADING_TIME` - Variable used in setting point in time where load reaches maximum. (Taken as 6 s)
- `TIME_STEP` - Variable used in increment in time. (Taken as 0.1 s)
- `MAX_ITERATIONS` - Variable used to limit the number of iterations as a limit to the solver.
- `NUMBER_OF_ELEMENTS` - Variable used to set the number of elements the domain is divided into.

3 Verification

3.1 Linear-elastic case

The solution obtained by the solver can be verified by using the classical theory of elasticity. The exact solution of the boundary value problem for *linear-elastic* material is as follows.

$$u_r^{elast} = (1+\upsilon)\frac{p}{E}\frac{a^2}{b^2-a^2}\left[(1-2\upsilon)r + \frac{b^2}{r}\right] \tag{15}$$

Using the solver described above, while setting the modulus Q to zero, the following plot is obtained.

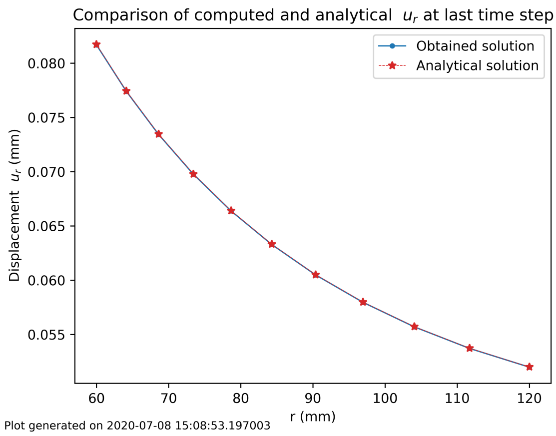

Fig. 3: Comparison of analytical and obtained solution of the displacement u_r

3.2 Convergence study

In this step, the convergence study with respect to number of elements and time increments/steps Δt for the visco-elastic model is performed. The respective plots are presented below.

It can be seen that any change in number of elements or any change in the increment time has little effect on the convergence of this particular investigation.

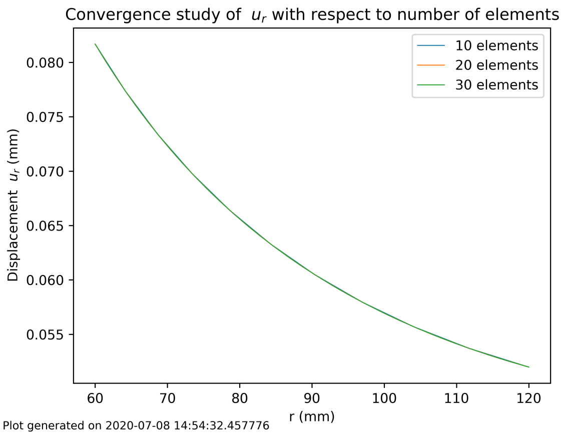

Fig. 4: Convergence study w.r.t. elements

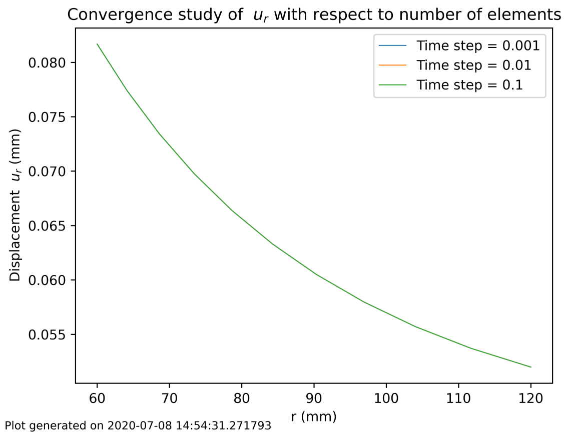

Fig. 5: Convergence study w.r.t. time increment Δt

4 Results

This section consists of the following distributions extracted from the investigation.

- Distribution of displacement u_r at $t = t_f$.
- Distribution of stresses σ_{rr} and $\sigma_{\phi\phi}$ at $t = t_f$.
- The time history of the widening of the pipe $u_r(r = b, t)$ for $t \in [0, t_f]$.

The plots can be seen below. In the figure 8, it can be clearly seen that the visco-elastic solution relaxes towards the elastic solution.

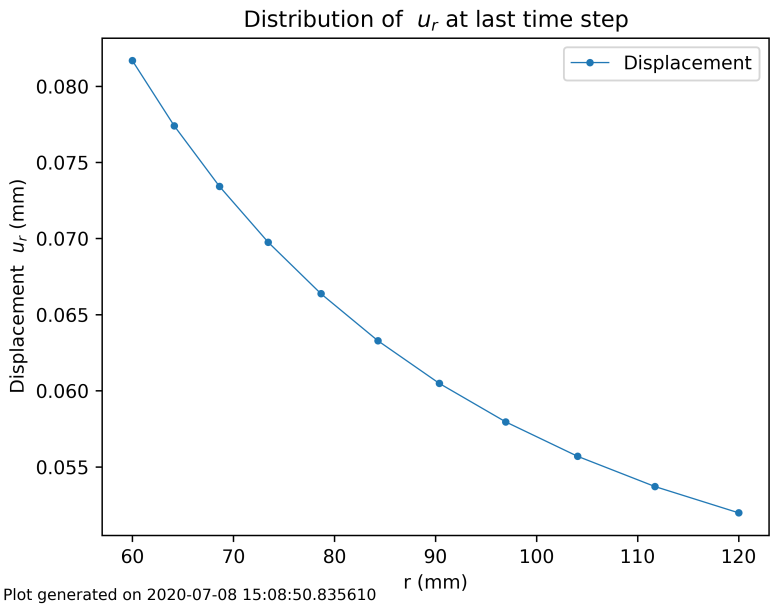

Fig. 6: Distribution of the displacement u_r at final time step.

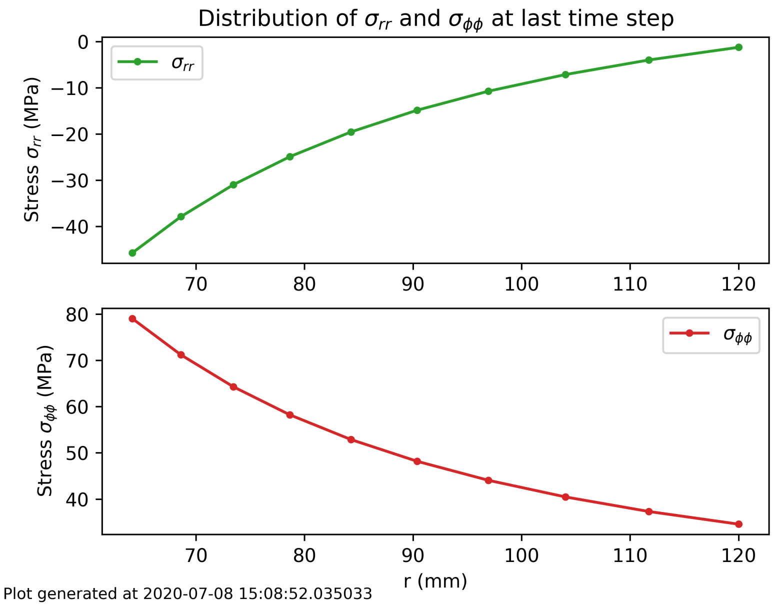

Fig. 7: Distribution of stresses $sigma_{rr}$ and $sigma_{\phi\phi}$ at last time step.

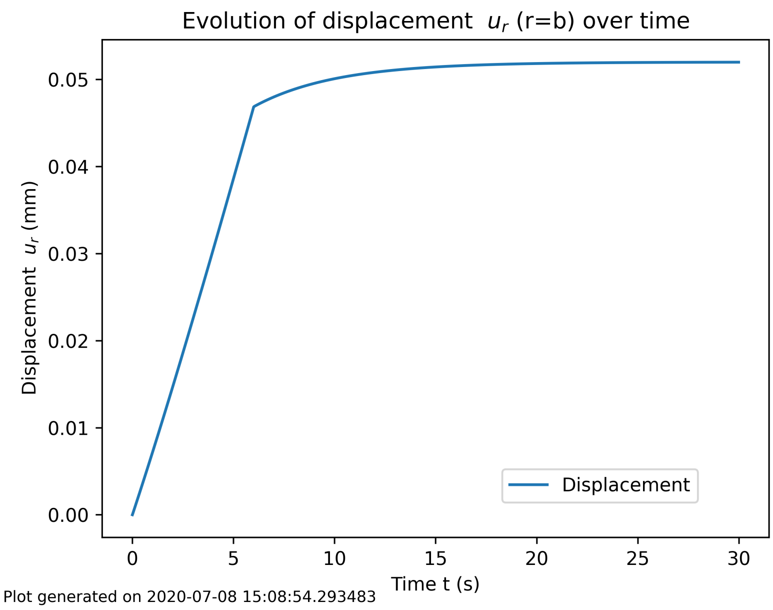

Fig. 8: Time history of displacement at specified nodes.