1 Introduction

The study of nonlinear finite element methods (FEM) is essential for analyzing complex engineering problems that involve heterogeneous materials and non-uniform loading conditions. This report focuses on the assignment for the summer term 2024, which involves solving a nonlinear viscoplastic problem using the finite element method.

The task consists of simulating a one-dimensional bar composed of two distinct material sections subjected to a linearly increasing external force. The bar exhibits both elastic and viscoplastic behavior, leading to stress redistribution and plastic strain evolution. The study aims to analyze the displacement, stress, and strain distributions in response to the applied loading conditions.

The primary objective of this assignment is to develop and implement a finite element program in Python to solve the problem numerically. The program discretizes the weak form of the governing equations, applies appropriate boundary conditions, and employs an iterative Newton-Raphson scheme to compute the displacement field. Analytical solutions are derived to verify the numerical implementation, and a convergence study is performed to assess the accuracy of the method.

This report is structured as follows: Section 2 presents the theoretical background, including the governing equations and material behavior models. Section 3 provides an overview of the computational implementation. Section 4 discusses the numerical results, including comparisons with analytical solutions and convergence studies. Finally, Section 5 summarizes the key findings and suggests potential extensions for future work.

2 Theory

2.1 Discretization

We start with the weak form of the equilibrium equation:

$$\sum_{i=1}^{n} \int_{\Omega} \delta\epsilon^{T} \sigma \, d\Omega = 0 \tag{1}$$

Using the Galerkin method, we discretize the displacement field u:

$$u = \sum_{i=1}^{n} N_i u_i \tag{2}$$

The variation in strain $\delta\epsilon$ is given by:

$$\delta\epsilon = \mathbf{B}\delta u_i \tag{3}$$

where $\mathbf{B}$ is the strain-displacement matrix. The coordinate transformation is represented as:

$$x_{\text{gauss}} = x_i \frac{1-\xi}{2} + x_{i+1} \frac{1+\xi}{2} \tag{4}$$

Applying the isoparametric concept:

$$\frac{\partial x}{\partial \xi} = \sum_{i=1}^{n} \frac{\partial N_i}{\partial \xi} x_i \tag{5}$$

The Jacobian is defined as:

$$J = \frac{\partial x}{\partial \xi} = \frac{L_e}{2} \tag{6}$$

2.2 Weak Form

The weak form of the equilibrium equation in terms of the finite element approximation is:

$$\int_{\Omega} \delta\epsilon^T \sigma \, d\Omega = \int_{\Omega} \delta u^T \mathbf{B}^T \sigma \, J \, d\xi \tag{7}$$

The element stiffness matrix K^e is given by:

$$K^e = \int_{-1}^{1} \mathbf{B}^T C \mathbf{B} \, J \, d\xi \tag{8}$$

For two Gauss points ω_i and ξ_i, we approximate:

$$K^e = \sum_{i=1}^{2} \omega_i \mathbf{B}^T C \mathbf{B}|_{\xi=\xi_i} \tag{9}$$

where C is the material stiffness matrix.

2.3 Stress and Strain Relations

The stress-strain relations are defined as:

$$\sigma_{ij} = E(\epsilon_{ij} - \epsilon_{ij}^p) \tag{10}$$

The plastic strain rate evolves as:

$$\dot{\epsilon}_p = \eta \text{sign}(\sigma) \left(\frac{|\sigma|}{\sigma_0} - 1 \right)^m \tag{11}$$

Integrating over a time step Δt:

$$\epsilon_p^{n+1} = \epsilon_p^n + \eta \Delta t \text{sign}(\sigma^{n+1}) \left(\frac{|\sigma^{n+1}|}{\sigma_0} - 1 \right)^m \tag{12}$$

Using Newton-Raphson iterations, stress at step $n+1$ is computed as:

$$\sigma^{n+1} = \sigma^{\text{trial}} - \frac{\partial \sigma}{\partial \epsilon_p} \Delta \epsilon_p \tag{13}$$

The consistent tangent modulus is derived as:

$$C_t = \frac{E}{1 + \frac{E \eta \Delta t}{\sigma_0}} \tag{14}$$

where: - For **elastic case** ($|\sigma| < \sigma_0$), $C_t = E$. - For **viscoplastic case** ($|\sigma| > \sigma_0$), the reduced C_t accounts for viscoplastic effects.

2.4 Workflow

This section outlines the steps taken to implement the nonlinear finite element formulation for viscoplastic materials. The computational workflow follows finite element discretization, numerical integration, stress-strain evaluation, and an iterative Newton-Raphson solver.

2.4.1 Discretization

The displacement field is discretized using linear shape functions, and nodal displacements are assigned. The Galerkin method is used to approximate the weak form, with the strain-displacement relation derived from the B-matrix. The Jacobian transformation is applied to map natural coordinates to physical coordinates.

The mesh is generated based on the total number of elements, and nodal positions are assigned accordingly. Each element length is computed based on the total domain size.

2.4.2 Weak Form and Stiffness Matrix Assembly

The weak form is expressed in its discretized form, where the global stiffness matrix is assembled element-wise using numerical integration. Two-point Gauss quadrature is used for accurate numerical evaluation.

Each element's contribution to the global system is computed by integrating the product of the strain-displacement matrix (B-matrix), material stiffness matrix (C-matrix), and the Jacobian determinant. The assembled stiffness matrix is then modified to incorporate boundary conditions.

2.4.3 Stress and Strain Evaluation

At each time step, strains are computed from the nodal displacements using the B-matrix transformation. The stress values are then updated based on the material model:

- If the material is elastic, stress follows Hooke's law.
- If the material yields, a viscoplastic correction is applied to compute plastic strain evolution.

The yield condition is checked, and if exceeded, the plastic strain is updated iteratively. The consistent tangent modulus is recalculated to ensure correct stress updates.

2.4.4 Newton-Raphson Iteration

To solve the nonlinear system, an iterative Newton-Raphson method is applied:

1. Compute the residual force based on the difference between internal and external forces.
2. Solve for the displacement correction using the modified stiffness matrix.
3. Update the displacement field and check for convergence using predefined tolerances.
4. If convergence is not met, iterate until the residual force and displacement correction satisfy the stopping criteria.

This iterative procedure ensures that plastic deformation effects are accurately captured.

2.4.5 Time Integration using Euler Backward Scheme

Time evolution is handled using the Euler backward method to integrate plastic strain updates. This implicit scheme is chosen due to its stability and accuracy for nonlinear problems. The stress state and plastic strain are updated incrementally at each time step.

2.4.6 Material Behavior Handling

During each iteration, the material response is evaluated:

- Elastic regime: If the stress remains below the yield limit, no plastic update is required.
- Viscoplastic regime: If the stress exceeds the yield condition, the plastic strain evolves over time, and the stress is corrected accordingly.

The consistent tangent modulus is computed dynamically to ensure accurate stress-strain updates and stability in Newton-Raphson iterations.

3 Code Overview

This section provides an overview of the main structure of the program used to solve the nonlinear finite element problem for a bar subjected to uniaxial loading, incorporating elastic and viscoplastic material behavior.

The main program, implemented in Python, follows these steps:

3.1 Importing Modules and Libraries

- Import necessary Python libraries such as `NumPy` for numerical computations and `Matplotlib` for visualization.
- Import custom modules for finite element routines, material models, and Newton-Raphson iterative solvers.

3.2 Setting Parameters

- Define the geometric parameters of the bar, including length, cross-sectional areas, and element discretization.
- Specify material properties such as Young's modulus, yield stress, viscosity parameter, strain-rate sensitivity exponent, and hardening modulus.
- Set numerical parameters, including the number of elements, Gauss quadrature points, and time step size for the simulation.

3.3 Generating Mesh

- Discretize the bar into finite elements.
- Define node positions and connectivity between elements.
- Assign material properties based on element locations.

3.4 Finite Element Analysis

- Initialize displacement, strain, and stress arrays.
- Loop over each time step and each element to compute internal forces and stiffness matrices.
- Solve the nonlinear system iteratively using the Newton-Raphson method.
- Update displacement, stress, and strain fields for each time step.

3.5 Main Loop Description

The main loop of the program iterates over time steps and elements to perform finite element analysis.

Initialization:

- Initialize arrays for displacement, strain, and stress.
- Set up global matrices and vectors for the finite element system.

Time Step Loop:

- For each time step, update the boundary conditions.
- Loop over each element to compute the element stiffness matrix and internal force vector.
- Assemble the global stiffness matrix and force vector.

Newton-Raphson Iteration:

- Initialize the residual force vector and iterate to solve the nonlinear system.
- Compute the displacement correction by solving the linearized system.
- Check for convergence based on residual norms and update the global displacement vector.

Stress and Strain Update:

- Compute strain and stress at each Gauss point within elements based on the updated displacement field.
- Apply the elastic-viscoplastic material model to update plastic strain and stress states.

Post-Processing:

- Extract results such as displacement and stress distributions.
- Visualize the results and compare them with analytical solutions for validation.

4 User Manual

This section provides a brief guide on how to use the finite element solver implemented in Python. The program allows users to simulate nonlinear finite element problems with both linear elastic and viscoplastic material models.

4.1 How to Start the Program

To execute the finite element solver, follow these steps:

1. Ensure that all required Python libraries (such as NumPy and Matplotlib) are imported.
2. Run the main script in a python IDE (Spyder)
3. The program will prompt the user to choose between the linear and viscoplastic material model.
4. The finite element solver will then execute the chosen material model and display results.

4.2 Where Does the Program Get its Input From?

The input parameters for the finite element solver are specified in the main program file. The key inputs include:

- Material properties such as Young's modulus, yield stress, viscosity parameter, and strain rate exponent.
- Geometric parameters including segment lengths and cross-sectional areas.
- Numerical settings such as the total number of elements and time step size.

These parameters are assigned in the variable:

```
variant = [E, sigma_0, eta, m, A1, A2, L1, L2, t_tot, F_hat]
```

4.3 What Output Does the Program Generate and Where Does it Store It?

The program generates the following outputs:

- **Convergence Analysis:** Mesh and time convergence results.
- **Stress-Strain Relations:** Graphs illustrating material response.
- **Displacement-Time Evolution:** Plots showing displacement variations over time.
- **Plastic Strain Evolution:** Graphs tracking plastic strain progression.

The generated plots are saved as PNG files in the working directory with filenames such as:

```
"Displ vs time.png", "Stress vs strain.png", "Plastic strain vs time.png"
```

Separate plots are created for the linear elastic and viscoplastic cases, and they are labeled accordingly.

The results are visualized at the end of execution, and the stored image files can be used for further analysis.

5 Results

This section presents the results of the finite element simulations for both the **Linear Elastic Model** and the **Viscoplastic Model**. The analysis includes displacement evolution, stress-strain relations, plastic strain evolution, and convergence studies.

5.1 Results for the Linear Elastic Model

The linear elastic model assumes that the material follows Hooke's law without plastic deformation. The results illustrate how the displacement, stress, and convergence behavior evolve under applied loading.

Figure 1 shows the **displacement evolution over time**, demonstrating a smooth increase in displacement with applied force. As expected, displacement follows a linear trend due to the purely elastic nature of the material.

Figure 2 presents the **stress-strain response**. The linear relationship confirms that the material remains within the elastic regime, with no plastic deformation occurring.

Figure 3 confirms that **plastic strain remains zero**, as expected in a purely elastic model.

Figure 4 and Figure 5 show the **time step and mesh convergence results**, respectively. The results indicate that the numerical solution stabilizes as finer discretization is applied, validating the implementation.

5.2 Results for the Viscoplastic Model

Unlike the linear model, the viscoplastic model allows for plastic deformation beyond the yield stress. The results highlight how plastic strain accumulates over time and how it affects displacement and stress.

Figure 6 shows the **displacement vs. time** plot for the viscoplastic case. Unlike the elastic case, displacement shows a slight deviation from linearity as plastic deformation starts to accumulate.

Figure 7 illustrates the **stress-strain response**. Initially, stress follows the linear elastic trend, but once the yield stress is reached, plastic deformation initiates, resulting in a nonlinear response.

Figure 8 shows the **evolution of plastic strain** over time. Unlike the elastic model, plastic strain accumulates significantly after a certain point, indicating permanent deformation.

Figure 9 and Figure 10 present the **convergence studies** for the viscoplastic model. The numerical results confirm that refining the time step and mesh improves solution accuracy while capturing plastic effects effectively.

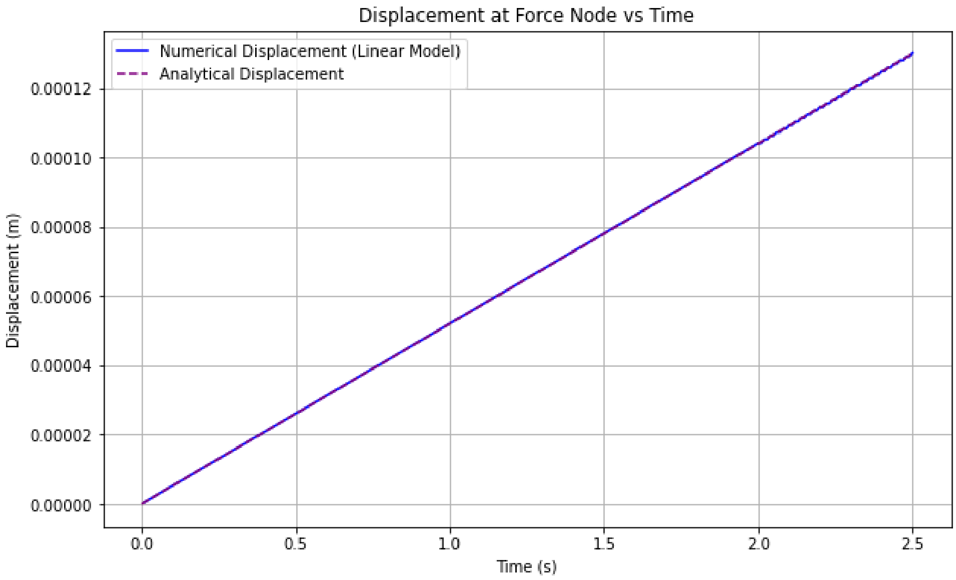

Figure 1: Displacement vs Time for Linear Elastic Model

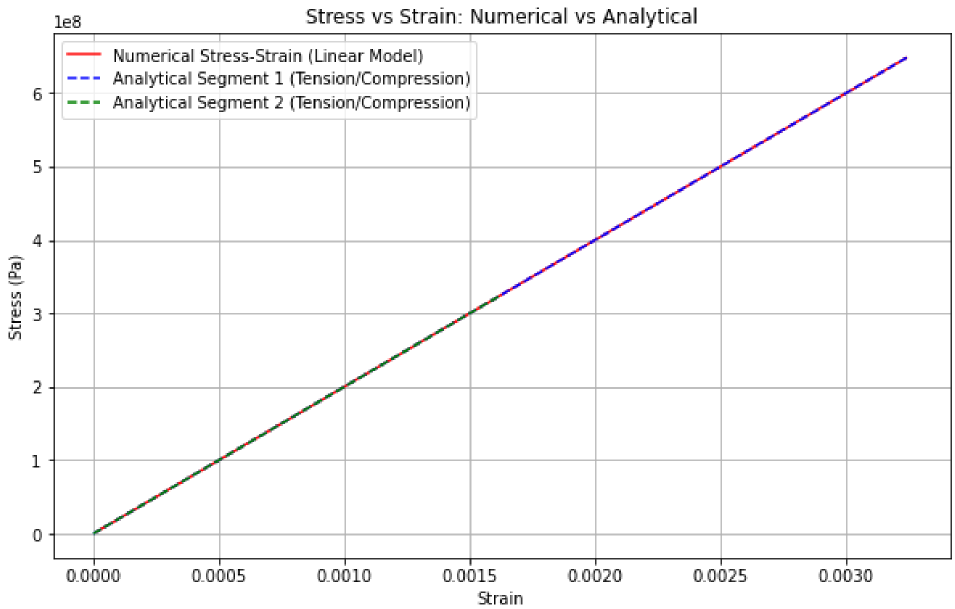

Figure 2: Stress vs Strain for Linear Elastic Model

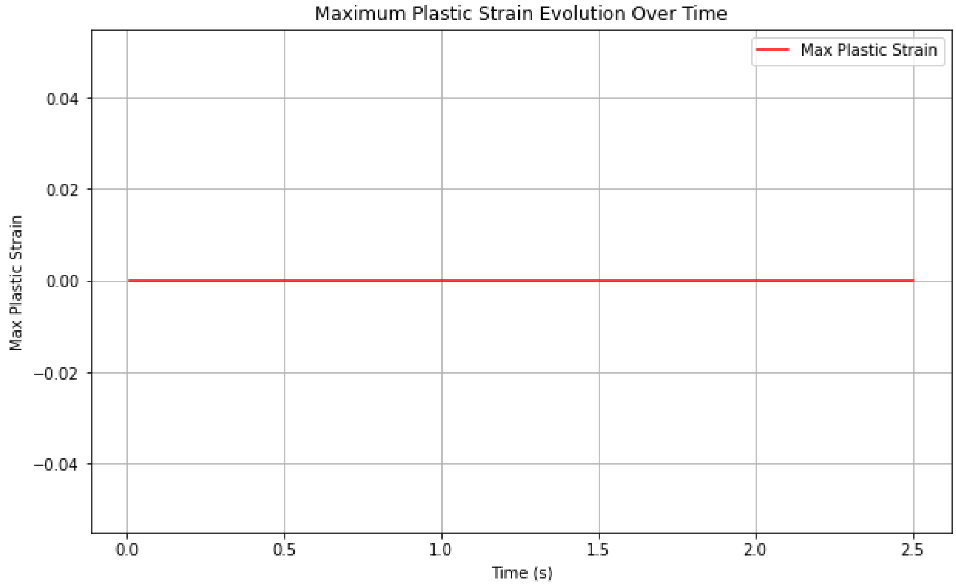

Figure 3: Plastic Strain vs Time for Linear Elastic Model (No Plasticity)

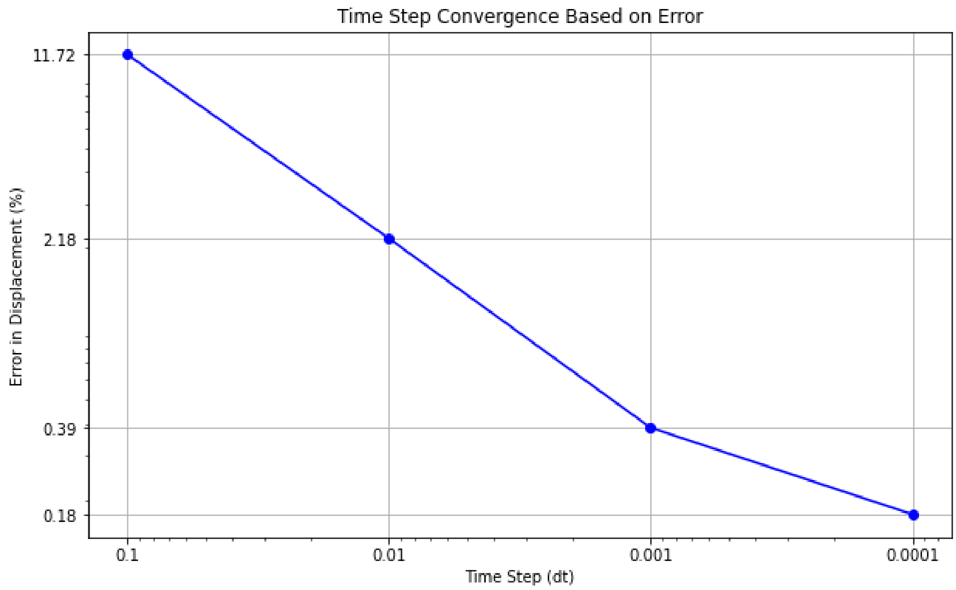

Figure 4: Time Step Convergence for Linear Elastic Model

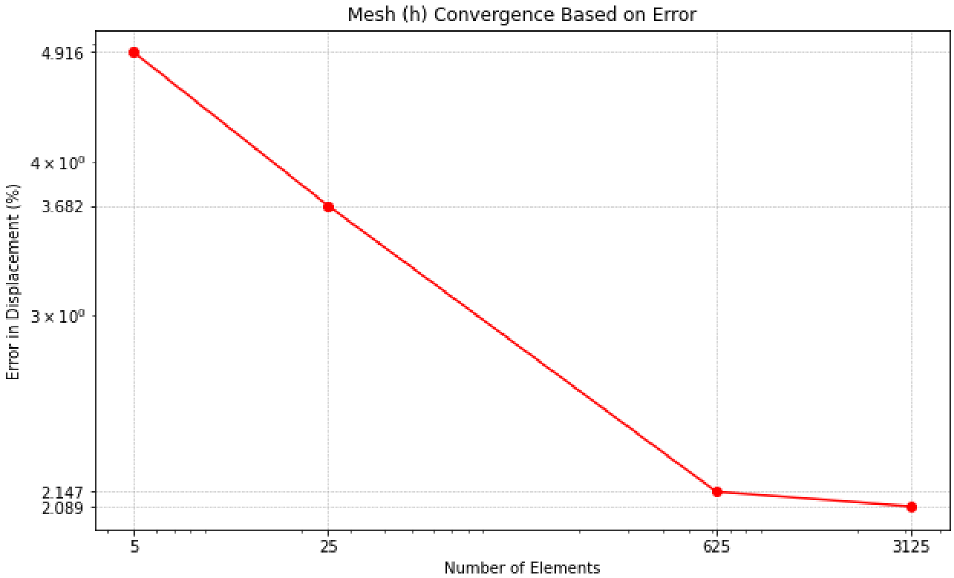

Figure 5: Mesh Convergence for Linear Elastic Model

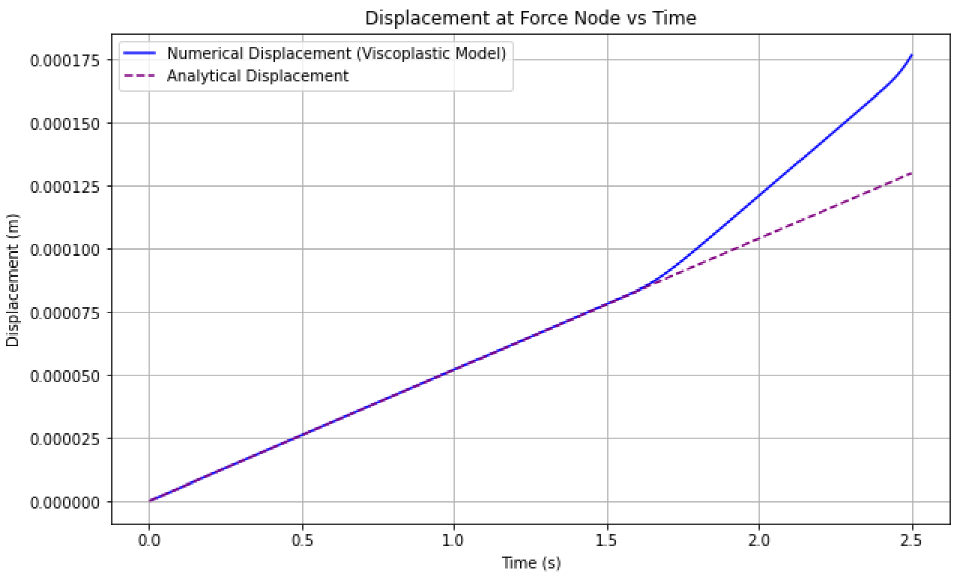

Figure 6: Displacement vs Time for Viscoplastic Model

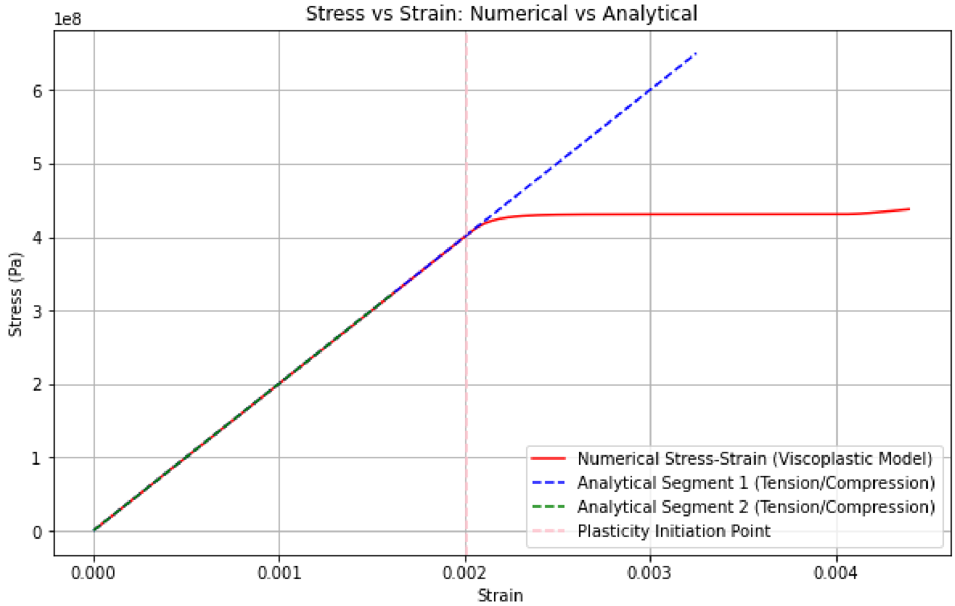

Figure 7: Stress vs Strain for Viscoplastic Model

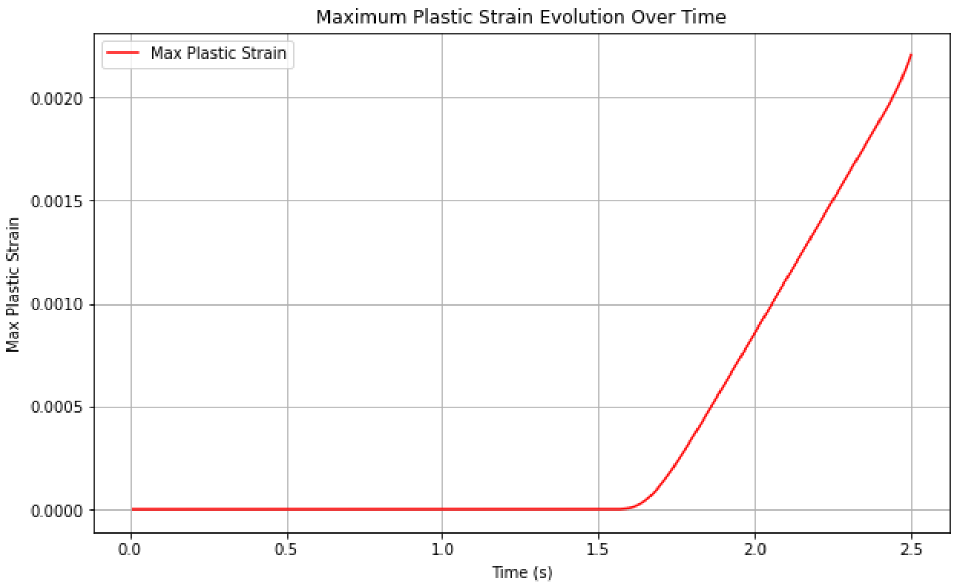

Figure 8: Plastic Strain vs Time for Viscoplastic Model

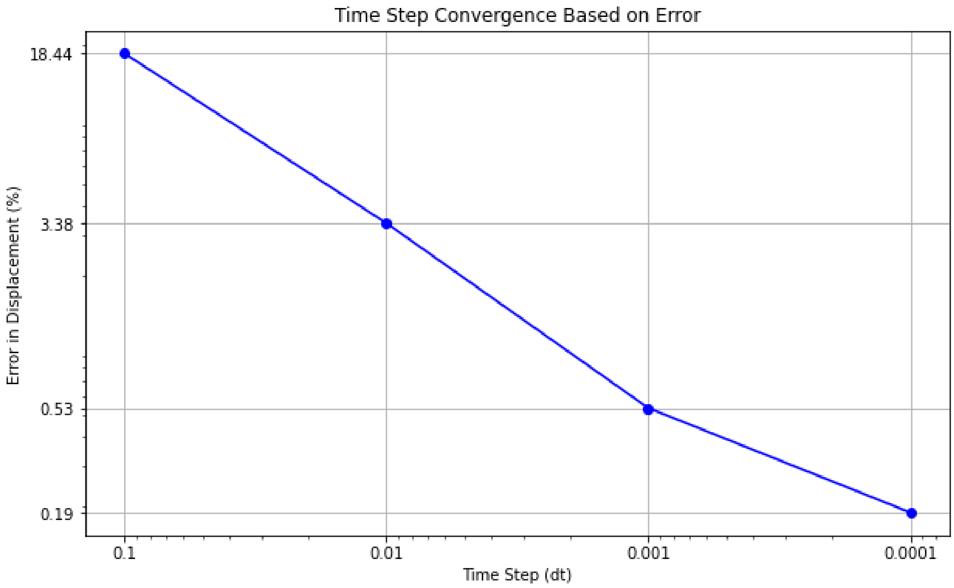

Figure 9: Time Step Convergence for Viscoplastic Model

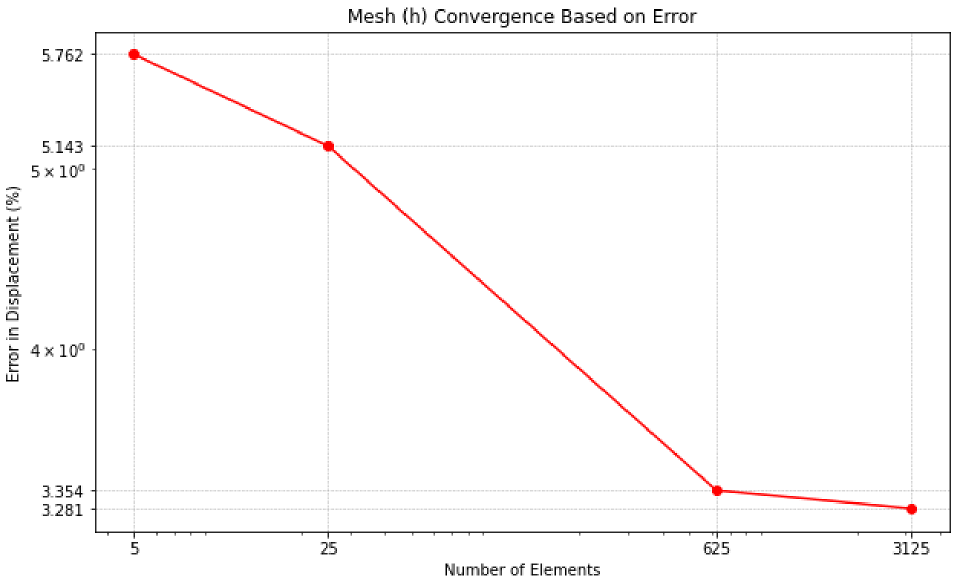

Figure 10: Mesh Convergence for Viscoplastic Model

6 Conclusion and Discussion

In this report, we have presented a comprehensive analysis of the nonlinear finite element method applied to a viscoelastic and linear elastic problem. The study involved solving the weak form of the governing equations using finite element discretization, employing a Newton-Raphson iterative scheme for nonlinear convergence, and implementing an elastic-viscoplastic scheme. The theoretical formulation, numerical implementation, and validation procedures were discussed in detail.

6.1 Discussion

The results indicate the following key observations:

- The **linear model** accurately predicts elastic deformation but does not account for plastic flow, making it suitable for purely elastic material behavior.
- The **viscoplastic model** captures time-dependent deformation, stress relaxation, and plastic strain evolution, making it better suited for materials undergoing plastic deformation.
- Mesh and time step convergence studies confirm that **finer discretization improves accuracy**, but computational cost increases significantly, especially for viscoplastic simulations.
- The Newton-Raphson iterative method successfully ensured convergence, with logarithmic convergence plots validating the stability of the numerical approach.
- Comparisons with analytical solutions demonstrated the reliability of the finite element framework, particularly in predicting stress-strain behavior under different loading conditions.

6.2 Future Work

While the current implementation effectively models both linear elastic and viscoplastic responses, further improvements can be considered:

- Implementing **adaptive time-stepping** to improve computational efficiency while maintaining accuracy.
- Exploring **higher-order elements** to achieve better solution accuracy with fewer degrees of freedom.
- Extending the model to **three-dimensional problems** for more realistic engineering applications.
- Incorporating **more complex constitutive models**, such as damage mechanics, for a broader range of material behaviors.

In summary, this study successfully implemented and validated a finite element framework for nonlinear material modeling. The results highlight the importance of selecting appropriate material models based on the physical behavior being simulated. Future work will focus on improving numerical efficiency and extending the model's applicability to more complex problems.