ME40064 System Modelling and Simulation - Coursework 2

922 Words, Candidate No. 11973, 2nd December 2025
Department of Mechanical Engineering, University of Bath

1. Introduction

Finite Element Method (FEM) is a powerful numerical technique for solving equations over a discrete domain. The simulated system is split into small regions called **elements**, connected by **nodes** which represent discrete points in the domain, together making up a **mesh**. Elements are evaluated using **basis functions** which approximate the solution within each element based on node values [1]. This approach allows for practical solutions to problems that may be difficult or impossible to solve analytically. In addition to this, the size and shape of elements can be adjusted to improve accuracy or reduce computational cost, making FEM a powerful and flexible tool for modelling (Figure 1).

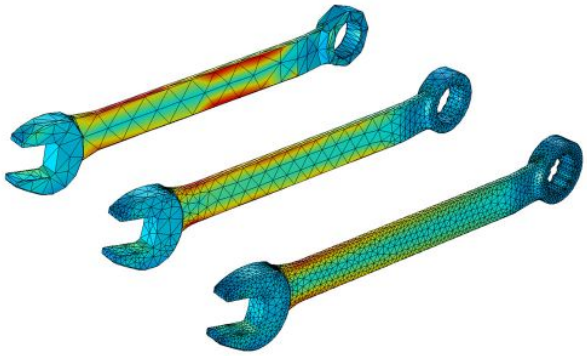

Figure 1: Finite Element Modelling of a Wrench under a Test Load Scenario [2]

This coursework focuses on the implementation and verification of a FEM solver for the transient diffusion-reaction equation, given by [3]:

$$\frac{\delta c}{\delta t} = D\frac{\delta^2 c}{\delta x^2} + \lambda c + f \tag{1}$$

Where:
- c is the concentration level
- D is the diffusion coefficient
- λ is the reaction rate
- f is a source term

The transient diffusion-reaction equation models processes where substances diffuse through a medium while undergoing reactions or being influenced by boundary interactions. Examples of situations modelled by this equation include the transfer of heat through a material or (as explored in Part 3 of this report) the diffusion of a drug through biological tissue.

This coursework describes the development and validation of a FEM solver for the transient diffusion-reaction equation. To keep the scope manageable, the solver was implemented in 1D, using MATLAB as the scripting language [4]

2. Part 1: Software Verification

2.1. Background

A static FEM solver was implemented in a previous coursework for the steady-state diffusion-reaction equation. This solver was subsequently adapted to solve the transient form of the equation (Equation 1).

For the initial case, the values of $D = 1$ and $\lambda = 0$ were used, representing a pure diffusion scenario with linear behaviour. The **Crank-Nicolson** finite difference method was used for time integration. It has unconditional stability but no damping of oscillations, providing a good compromise between accuracy and stability at this stage [5].

The problem space was further defined with the following conditions:

Problem Space	$0 \leq x \leq 1$
Left Boundary Condition	Dirichlet: $c(0,t) = 0$
Right Boundary Condition	Dirichlet: $c(1,t) = 1$
Initial Condition	$c(x,0) = 0$

Table 1: Initial Case Conditions

These conditions have a known analytical solution, given by Equation 2:

$$c(x,t) = x + \frac{2}{\pi}\sum_{n=1}^{\infty}\frac{(-1)^n}{n}e^{-n^2\pi^2 t}\sin(n\pi x) \tag{2}$$

The analytical solution allows for direct comparison of results between the FEM solver and expected values, providing a quantitative measure of accuracy.

2.2. Software Architecture

The solver was implemented with a modular, object-oriented software architecture to improve readability and control flow. Classes were created to encapsulate well-defined functions of the solver, such as mesh generation or plotting (Figure 2).

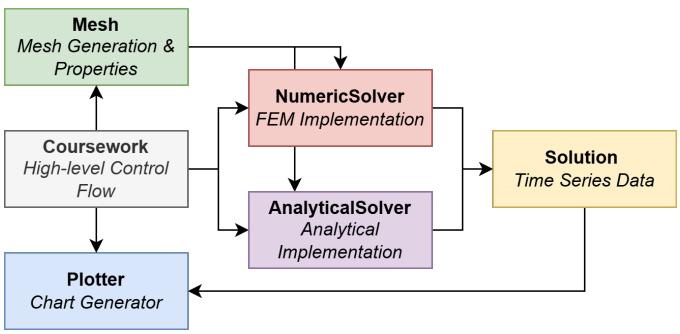

Figure 2: High-Level Software Architecture of the FEM Solver

2.3. Results

Having implemented the FEM solver as described above, a simulation was run using a mesh size of 50 elements and a time step of 0.01s, over the time period $0 < t \leq 1s$.

After this, the results were plotted on a series of charts for a visual comparison of the two solutions. The first of these were heatmaps which are an effective method for visualising the 1D diffusion over time (Figure 4, Figure 3).

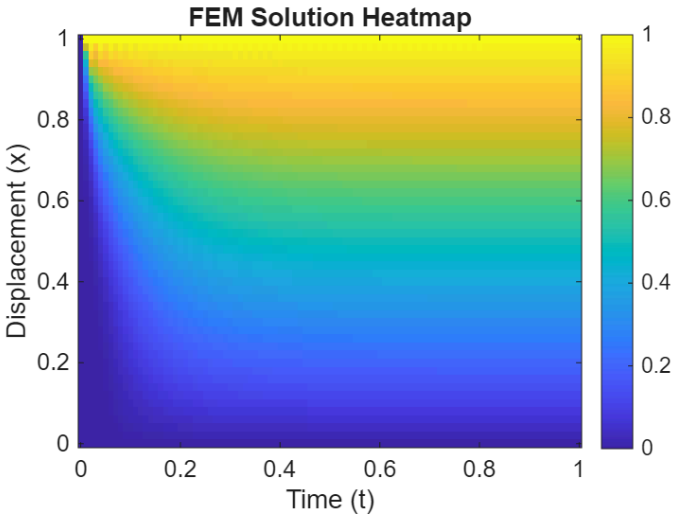

Figure 3: FEM Solution of Diffusion Equation over using the Crank-Nicolson method over $0 \leq x \leq 1$ and $0 \leq t \leq 1s$

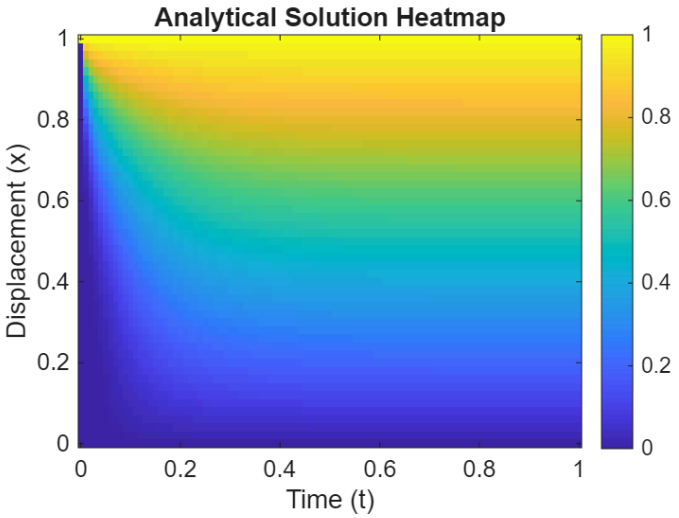

Figure 4: Analytical Solution of Diffusion Equation over $0 \leq x \leq 1$ and $0 \leq t \leq 1s$

The data was also represented in a 2D plot, showing the concentration through the mesh at sample times of $t = 0.05s, 0.1s, 0.3s, 1.0s$, shown in Figure 5 and Figure 6.

Additionally, a chart was created for both solutions at a single point in the mesh ($x = 0.8$), shown in Figure 7. Unlike previous plots, this shows both methods on the same axes for direct comparison, demonstrating the agreement between the two solutions.

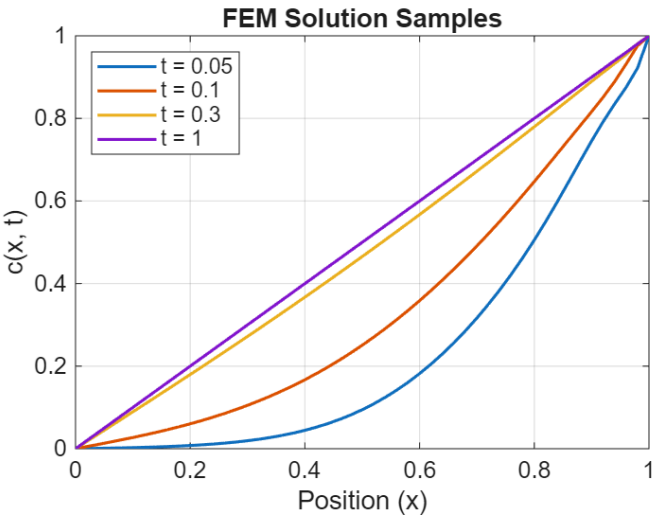

Figure 5: FEM Solution of Diffusion Equation over using the Crank-Nicolson method over $0 \leq x \leq 1$ and at $t = 0.05s, 0.1s, 0.3s, 1.0s$

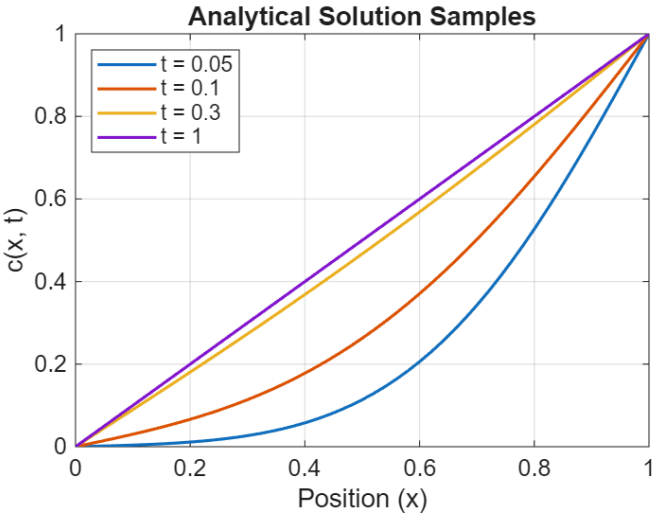

Figure 6: Analytical Solution of Diffusion Equation over $0 \leq x \leq 1$ and at $t = 0.05s, 0.1s, 0.3s, 1.0s$

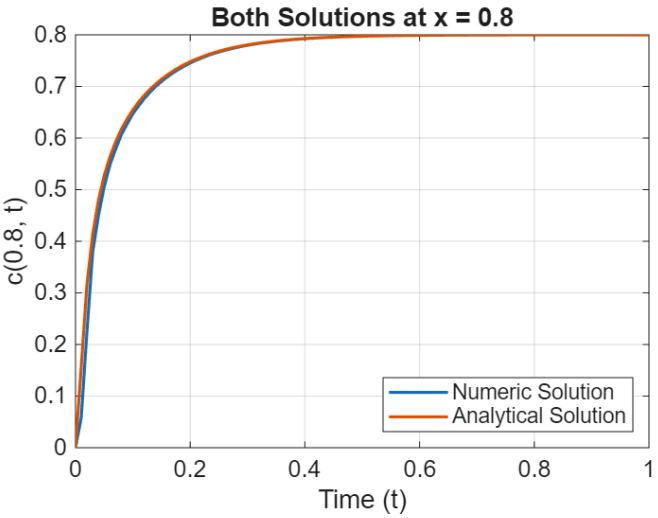

Figure 7: Comparison of Analytical and FEM Solutions at $x = 0.8$ over $0 \leq t \leq 1s$

2.4. Spacial and Temporal Convergence

To quantitatively assess the accuracy of the FEM solver, the **Root Mean Square (RMS)** error between numerical and analytical solutions was evaluated over a range of element and time step sizes. As shown in Figure 8 and Figure 9, the RMS error decreases with both smaller element sizes and smaller time steps, demonstrating convergence of the numerical solution towards the analytical solution with increasing resolution.

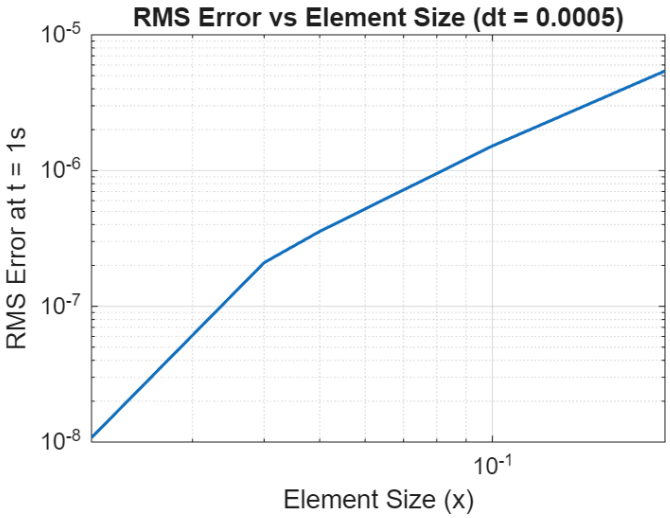

Figure 8: Comparison of RMS errors at $t = 1s$ for Varying Element Sizes

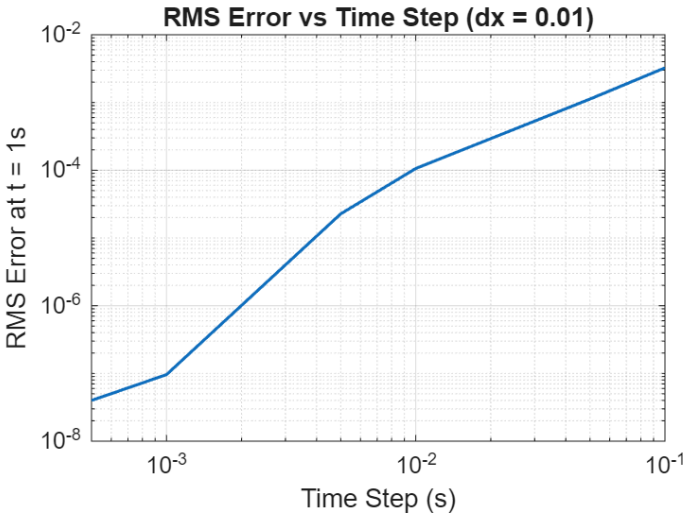

Figure 9: Comparison of RMS errors at $t = 1s$ for Varying Time Steps

2.5. Testing and Validation

A set of unit tests were created alongside the FEM solver, to verify the functionality of individual components such as mesh generation, element assembly, and time integration. As part of the development process, the project was continuously tested to ensure it passed all scenarios.

In particular, a unit test was created to validate the solver against a manufactured solution of the transient diffusion-reaction equation. This involved selecting specific values for D, λ, and f such that the solution could be expressed in a simple analytical form.

3. Part 2: Software features

3.1. Error Evaluation

In Part 1 of the coursework, the RMS error term was used to evaluate the accuracy of the FEM solver. While RMS is a useful metric, it can be sensitive to outliers and therefore may not always provide a complete picture of the solution accuracy. L2 norm doesn't suffer as much from this, and is more widely used in literature as a result [3] To address this, a dedicated L2 error evaluation class was added to the solver, allowing for more robust error analysis.

3.2. Integration Methods

Using the L2 norm error evalutation class, the performance of three different time integration methods was compared: Forward (Explicit) Euler, Backward (Implicit) Euler, and Crank-Nicolson.

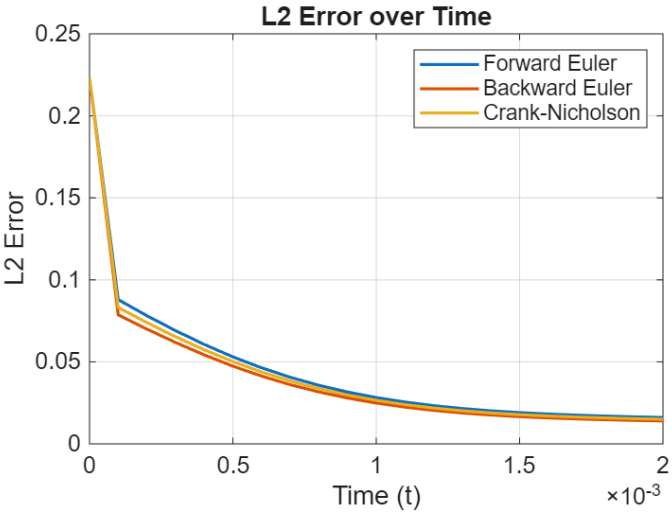

Figure 10: Comparison of RMS errors at $t = 1s$ for Varying Time Steps

3.3. Quadratic Basis Functions

3.4. Gaussian Quadrature

- The **Mesh** and **MeshElement** classes were modified to support higher-order elements.

3.5. Comparing Solver Methods

- Forward Euler - conditionally stable, requires 0.0002s dt for stability
  - needs to be dt <= (dx^2) / (2D)
- other two methods - unconditionally stable

Next, the FEM solver was extended to account for the following advanced features:

- Different solver methods (Explicit Euler, Implicit Euler, Crank-Nicolson)
- Gaussian quadrature for numerical integration
- Quadratic basis functions
- Using L2 norm to evaluate solution accuracy

NEED TO REFINE MESH!

4. Part 3: Modelling & Simulation Results

5. Conclusion

6. References

[1] J. L. G. Dhatt G. Touzot, *Finite Element Method*. Wiley.

[2] "Finite Element Mesh Refinement." [Online]. Available: https://www.comsol.com/multiphysics/mesh-refinement

[3] W. Hundsdorfer, "Numerical Solution of Advection-Diffusion-Reaction Equations," 2000. [Online]. Available: https://bpb-us-e1.wpmucdn.com/blogs.gwu.edu/dist/9/297/files/2018/01/66bdd115ac105ea17af303e73d4fec449754-v448bk.pdf

[4] "MATLAB." [Online]. Available: https://mathworks.com/products/matlab.html

[5] C. W. T. C. Sun, "Unconditionally stable Crank-Nicolson scheme for solving two-dimensional Maxwell's equations," 2003. [Online]. Available: https://doi.org/10.1049/el:20030416

7. Use of Generative AI

This coursework was completed in Visual Studio Code (with the MATLAB Extension), using Typst for report writing. The GitHub Copilot AI tool was enabled, providing generative suggestions for report phrasing and code snippets.