AE332 – Modeling and Analysis Lab

A Report submitted

For internal assessment for Coarse

Modeling and Analysis Lab

in

Aerospace Engineering

by

Shingala Vaidik

(SC21B054)

pursued in

Department of Aerospace Engineering

Indian Institute of Space Science and Technology

to

Dr. Manu

INDIAN INSTITUTE OF SPACE SCIENCE AND TECHNOLOGY

THIRUVANANTHAPURAM

November 26, 2023

Contents

Abstract

This report presents a numerical analysis of parabolic equations using the finite element method. The focus is on solving transient heat conduction problems with various boundary conditions and initial conditions. The numerical simulations are implemented using the FreeFEM software, and the results are compared with analytical solutions where applicable.

In Problem 1, a simple 1D transient heat conduction scenario is simulated with Dirichlet boundary conditions. The obtained numerical solution is compared with the analytical solution, demonstrating the accuracy of the numerical approach.

In Problem 2, a more complex 1D transient heat conduction problem is considered, involving non-homogeneous boundary conditions. The numerical solution is again compared with the analytical solution, showing good agreement.

Problem 3 introduces a Neumann boundary condition, providing insight into the effect of an insulated boundary on the temperature distribution. The numerical solution is compared with the analytical solution, demonstrating the capability of the finite element method to handle different boundary conditions.

Finally, Problem 4 extends the analysis to a 2D domain with a Neumann condition of constant gradient at the left boundary. The numerical solution is compared with an analytical solution, showcasing the versatility of the method in handling multi-dimensional problems.

The report concludes with a grid convergence and time step convergence study, highlighting the importance of choosing appropriate grid sizes and time steps for accurate numerical solutions.

Overall, the numerical analysis presented in this report serves as a valuable tool for understanding and solving parabolic equations in the context of transient heat conduction.

Chapter 1

Numerical Analysis of Parabolic Equations

Weak Form

One-dimensional transient heat conduction equation can be represented as in Equation 1.1.

$$\frac{du}{dt} = \alpha \frac{d^2u}{dx^2} \tag{1.1}$$

This equation is multiplied with a weight function (v) and is integrated over the domain.

$$\frac{du}{dt} \cdot v = \alpha \frac{d^2u}{dx^2} \cdot v \tag{1.2}$$

Equation 1.2 incorporates the temporal term, which is discretized using Euler's forward difference method. Simultaneously, the diffusive term is simplified through the application of the chain rule, selectively eliminating terms that become zero at the boundaries. The resulting expression is presented in Equation 1.3.

$$v\left(\frac{u^{n+1} - u^n}{dt}\right) = -\alpha \int_0^L \frac{du}{dx} \cdot \frac{dv}{dx}\, dx \tag{1.3}$$

1.1 Problem 1

A simple 1D transient heat conduction is simulated. The governing equations and boundary conditions are provided as follows.

$$\frac{du}{dx} = \frac{d^2u}{dx^2}, \quad \forall 0 < x < 1,\, t > 0 \tag{1.4}$$

$$u(x,0) = x - x^2, \quad u(0,t) = u(1,t) = 0 \tag{1.5}$$

1.1.1 Code

```
load "msh3";
real dt = 0.002;
int te = 1;
real m = 100;
meshL Th = segment(m, [x*1]);
```

```
real [ int ] xaxis ( m +1) , uline ( m +1) ;
ofstream q1 (" Resulty.csv") ;
func u0 = x - x^2;
fespace Vh(Th, P1);
Vh u = u0,  v, uold;

problem heatconduction(u, v)=
int1d(Th)(dx(u)*dx(v) + v*u/dt) - int1d(Th)(v*uold/dt)+on(1,2,u=0);
for(real t=0; t<te; t+=dt)
{
    uold = u;
    heatconduction;
    for(int i=0;i<=m;i++)
    {
        if(i==m)
        {
            q1 << uold[][i];
        }
        else
        {
            q1 << uold[][i] << " , ";
        }
    }
    q1 << endl;
}
plot (u , value = true ) ;
```

1.1.2 Output of code

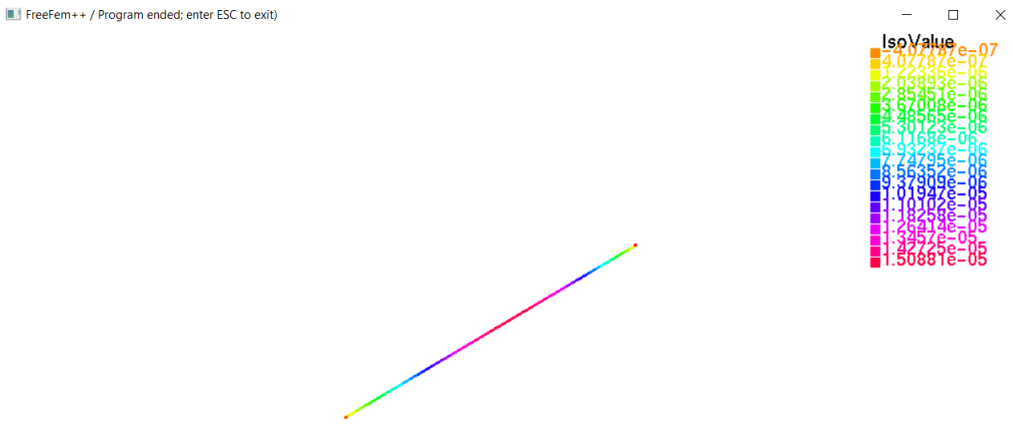


Figure 1.1: Freefam Plot

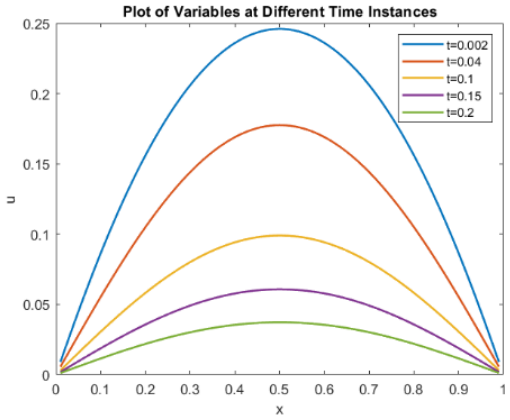


Figure 1.2: Matlab Plot

1.2 Compare with Analytical Solution

Compare the obtained solution with the analytical solution given by Equation 1.6.

$$u(x,t) = \frac{4}{\pi^3} \sum_{n=1}^{\infty} \frac{1-(-1)^n}{n^3} e^{-n^2\pi^2 t} \sin(n\pi x) \tag{1.6}$$

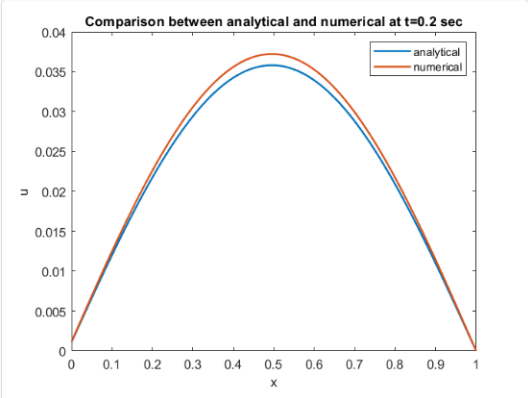


Figure 1.3: Comparison between analytical and numerical solution

1.3 Investigate Neumann Boundary Condition

In the current problem, the solution is obtained considering a Dirichlet boundary condition. Investigate the effect of a Neumann boundary condition (i.e., when the domain is insulated at both ends), by modifying the boundary conditions.

1.3.1 Code

```
load "msh3";
real dt = 0.002;
int te = 1;
real m = 100;
meshL Th = segment(m, [x*1]);
real [ int ] xaxis ( m +1) , uline ( m +1) ;
ofstream q1 (" Resulty.csv") ;
func u0 = x - x^2;
fespace Vh(Th, P1);
Vh u = u0,  v, uold;

problem heatconduction(u, v)=
int1d(Th)(dx(u)*dx(v) + v*u/dt) - int1d(Th)(v*uold/dt)//+on(1,2,u=0); insulation
    at both ends
for(real t=0; t<te; t+=dt)
{
    uold = u;
    heatconduction;
    for(int i=0;i<=m;i++)
    {
        if(i==m)
        {
            q1 << uold[][i];
        }
        else
        {
            q1 << uold[][i] << " , ";
        }
    }
    q1 << endl;
}
plot (u , value = true ) ;
```

1.3.2 Plot

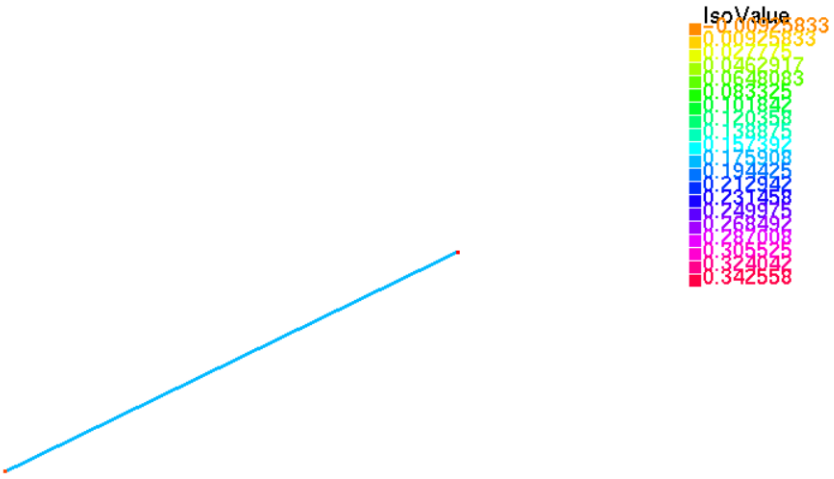


Figure 1.4: Freefem plot

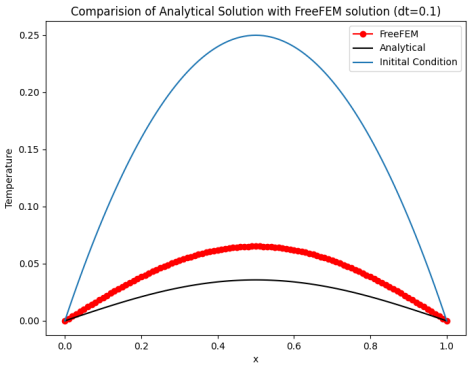
1.4 Grid Convergence and Time Step Convergence Study

1.4.1 Time Convergence Study

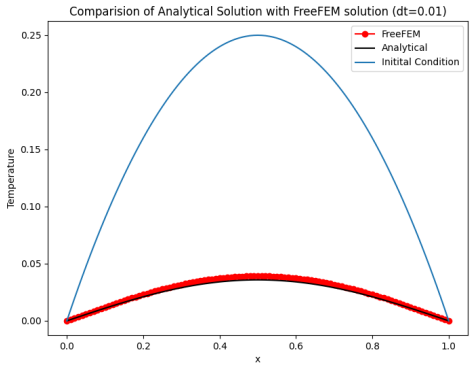
In this study, the grid size is chosen as $dx = 0.01$, and the time step dt is varied from 0.1 to 0.01 to 0.001. The numerical solution approaches the analytical solution as dt is reduced, leading to a decrease in error. The results are presented in Figure 1.5.

1.4.2 Grid Convergence Study

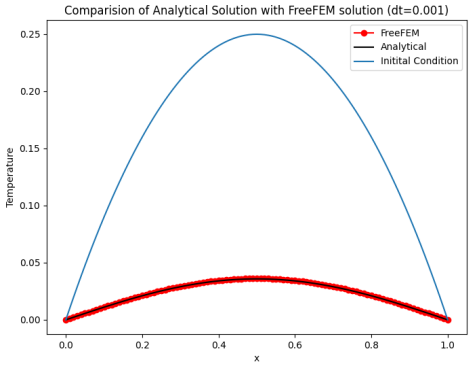
For the grid convergence study, the time step is kept constant at $dt = 0.001$, while the grid size dx is varied from 0.1 to 0.01 to 0.001. Minimal changes are observed as the grid size is altered. The results are illustrated in Figure 1.6.



(a) $dt = 0.1$

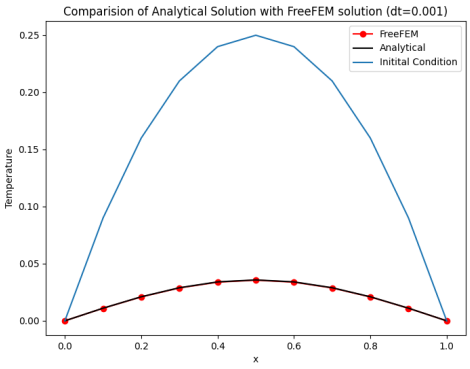


(b) $dt = 0.01$

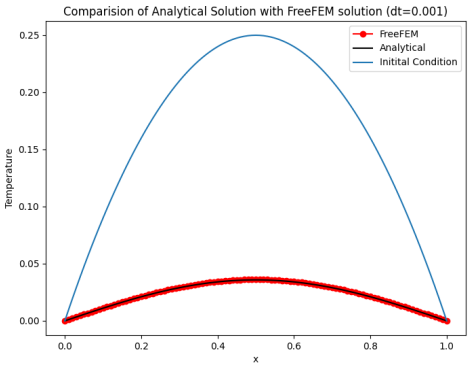


(c) $dt = 0.001$

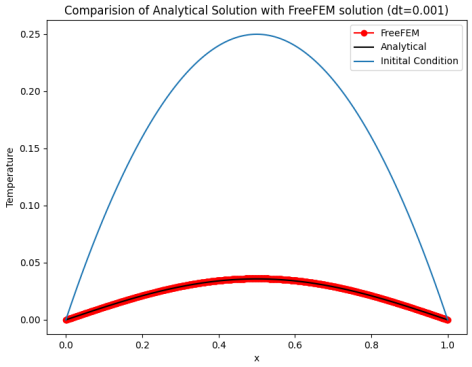
Figure 1.5: Time Step Convergence Study for Problem 1



(a) $dx = 0.1$



(b) $dx = 0.01$



(c) $dx = 0.001$

Figure 1.6: Grid Convergence Study for Problem 1

Chapter 2

Problem 2

2.1 Problem 2.1

The governing equation, boundary conditions, and initial conditions for the problem are stated below.

$$u_t = u_{xx} \quad \text{for } 0 < x < 2,\ t > 0 \tag{2.1}$$

$$u(x,0) = \begin{cases} x & \text{for } 0 < x < 1 \\ 2 - x & \text{for } 1 < x < 2 \end{cases} \tag{2.2}$$

$$u(0,t) = u_x(2,t) = 0 \tag{2.3}$$

2.1.1 Code

```
load "msh3";
real dt = 0.002;
int te = 1;
real m = 100;
meshL Th = segment(m, [x*2]);
real [ int ] xaxis ( m +1) , uline ( m +1) ;
ofstream q1 (" aditya2.dat") ;
func real u0(){
    if(x<=1){ return x ;}
    else{return 2 -x ;}
}
fespace Vh(Th, P1);
Vh u = u0(),  v, uold;

problem heat(u, v)=
int1d(Th)(dx(u)*dx(v) + v*u/dt) - int1d(Th)(v*uold/dt)
+ on(1, u=0)+int1d(Th, 1)(0*u*v);
for(real t=0; t<te; t+=dt)
{
```

```
    uold = u;
    heat;
    for(int i=0;i<=m;i++)
    {
        if(i==m)
        {
            q1 << uold[][i];
        }
        else
        {
            q1 << uold[][i] << " , ";
        }
    }
    q1 << endl;
}
plot (u , value = true ) ;
```

2.1.2 Plot

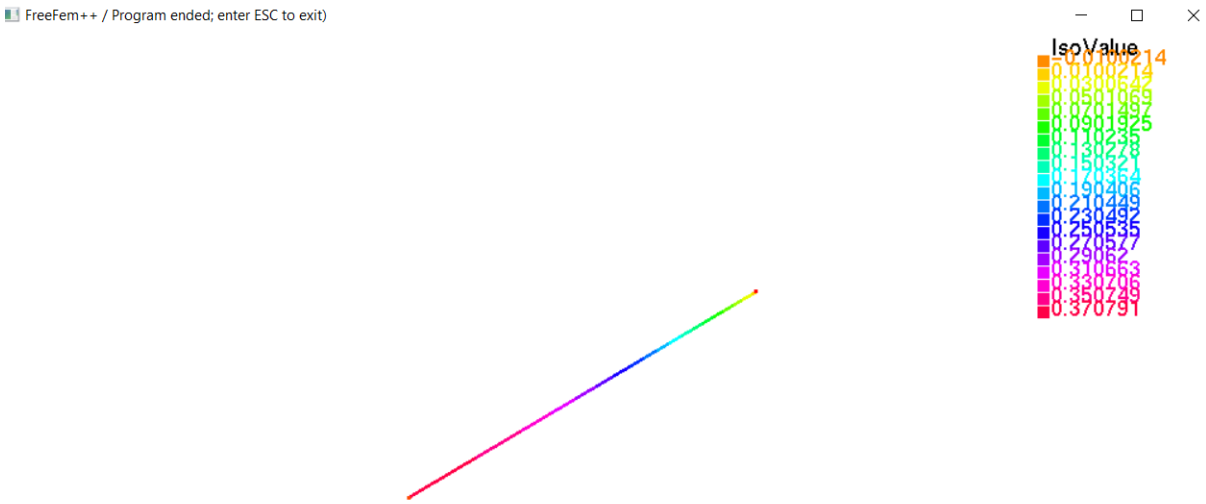


Figure 2.1: Freefem plot

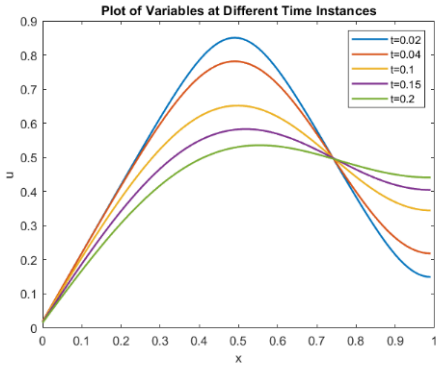


Figure 2.2: Matlab plot

2.2 Analytical solution

Compare the obtained solution with analytical solution

$$u(x,t) = \frac{32}{\pi^3}\sum_{n=1}^{\infty} sin\left(\frac{(2n-1)\pi}{4}\right)\frac{1-cos\left(\frac{(2n-1)\pi}{4}\right)}{(2n-1)^2}e^{\frac{(2n-1)^2}{16}\pi^2 t}sin\left(\frac{(2n-1)\pi}{4}x\right) \tag{2.4}$$

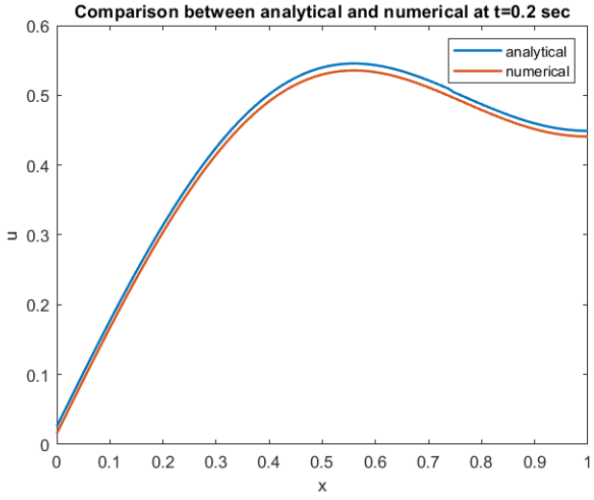


Figure 2.3: Comparison between analytical and numerical solution

We can see that the free fem solution is almost matching with the analytical solution for the given time step and grid size.

Chapter 3

Problem 3

3.1 Problem 3.1

In practical problems, there can be instances when we encounter non-standard or non-homogeneous boundaries. Here, a non-homogeneous boundary condition problem is discussed.

$$\frac{du}{dt} = \frac{d^2u}{dx^2},\ 0 < x < 3,\ t > 0 \tag{3.1}$$

$$u(x = 0) = 4x - x^2, u(0, t) = 0, u(3, t) = 3 \tag{3.2}$$

3.1.1 Code

```
load "msh3";
real dt = 0.002;
int te = 1;
real m = 100;
meshL Th = segment(m, [x*3]);
real [ int ] xaxis ( m +1) , uline ( m +1) ;
ofstream q1 (" Results2.csv") ;
func u0 = 4.*x - x^2;
fespace Vh(Th, P1);
Vh u = u0,  v, uold;

problem heatconduction(u, v)=int1d(Th)(dx(u)*dx(v) + v*u/dt) - int1d(Th)(v*uold/dt
    )+on(1,u=0)+on(2,u=3);
for(real t=0; t<te; t+=dt)
{
    uold = u;
    heatconduction;
    for(int i=0;i<=m;i++)
    {
        if(i==m)
        {
```

```
            q1 << uold[][i];
        }
        else
        {
            q1 << uold[][i] << " , ";
        }
    }
    q1 << endl;
}
plot (uold , value = true ) ;
for (int i =0; i <= m ; i ++)
{
xaxis [ i ]= i / m ;
uline [ i ] = u [][ i ];
q1 << xaxis [ i ] <<"\t" <<"\t" << uline [ i ] << endl ;
cout << xaxis [ i ] <<"\t" << uline [ i ] << endl ;
}
```

3.1.2 Plot

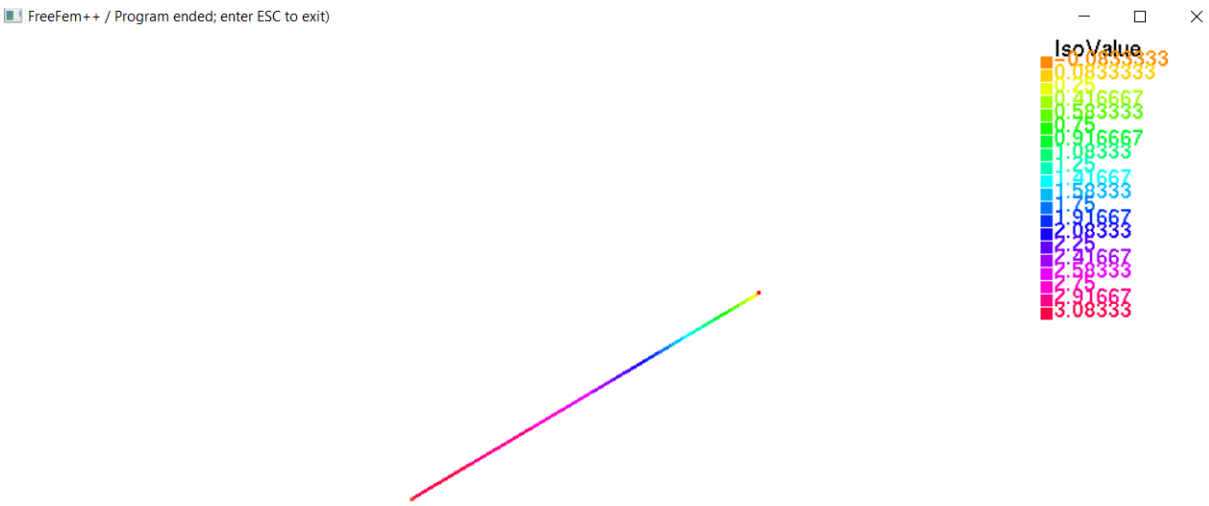


Figure 3.1: Freefem plot for problem 3

As the time passes the temperature distribution in the domain linearize and a linear distribution exists at steady state.This can be seen in figure 3.2.

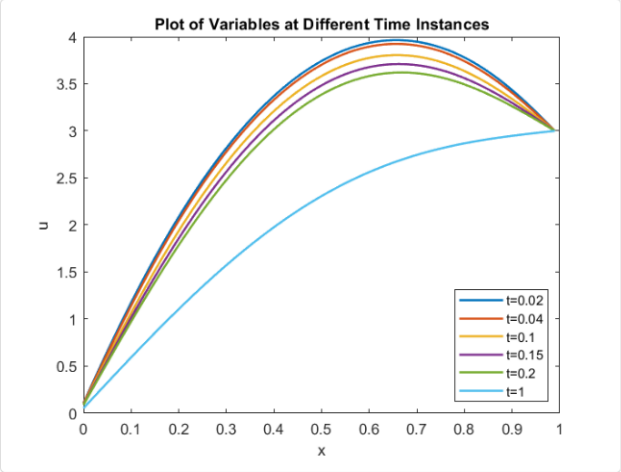


Figure 3.2: Matlab plot for problem 3

3.2 Analytical solution

Compare the obtained solution with analytical solution.

$$u(x,t) = x + \frac{32}{\pi^3}\sum_{n=1}^{\infty}\frac{(1-(-1)^n)}{n^3}e^{-\frac{n^2\pi^2}{9}t}\sin\frac{n\pi}{3}x.$$

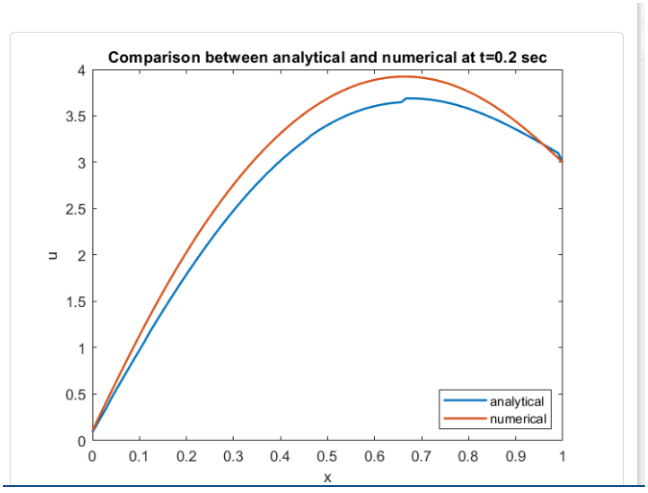


Figure 3.3: Comparison between analytical and numerical solution

Chapter 4

Problem 4

4.1 Problem 4.1

A problem similar to the previous one attempted. However in this problem a neumann condition of a constant gradient is given at the left boundary. The governing equations initial conditions and the boundary conditions are summarised below. This has been solved in 2D Domain.

$$\frac{du}{dt} = \frac{d^2u}{dx^2},\ 0 < x < 1,\ t > 0 \tag{4.1}$$

$$u(x = 0) = 0, u_x(0, t) = -1, u_x(1, t) = 0 \tag{4.2}$$

4.1.1 Code

```
load "msh3";
real dt = 0.02;
int te = 10;
real m = 100;
mesh Th = square(m, m, [x*1, y*1]);
real [ int ] xaxis ( m +1) , uline ( m +1) ;
ofstream file2 (" a.csv") ;
func u0 = 0;
fespace Vh(Th, P1);
Vh u = u0,  v, uold;
int count = 0;
problem heat(u, v)=
int2d(Th)(dx(u)*dx(v) + v*u/dt) - int2d(Th)(v*uold/dt)
+ int1d(Th, 4)(v*(-1));
for(real t=0; t<=te; t+=dt)
{
    uold = u;
    heat;
}
plot(u, wait =true, fill=true,value=true);
```

```
for (int i =0; i <= m ; i ++)
{
xaxis [ i ]= i / m ;
uline [ i ] = u [][ i ];
file2<< xaxis [ i ] <<"\t" <<"\t" << uline [ i ] << endl ;
cout << xaxis [ i ] <<"\t" << uline [ i ] << endl ;
}
```

4.1.2 Plot

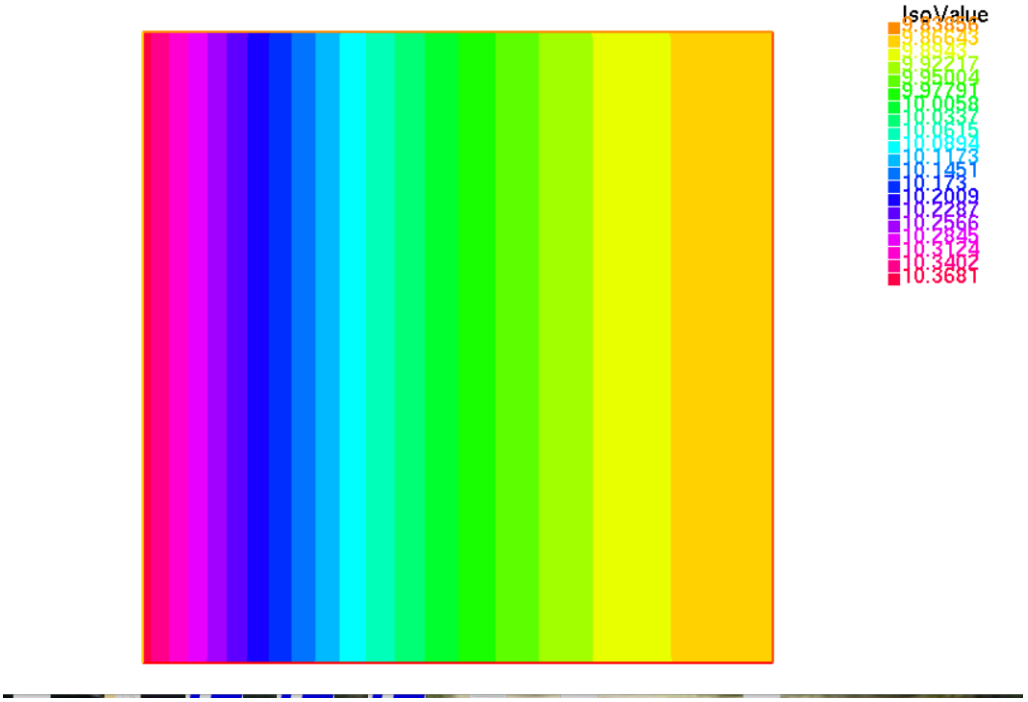


Figure 4.1: Freefem plot

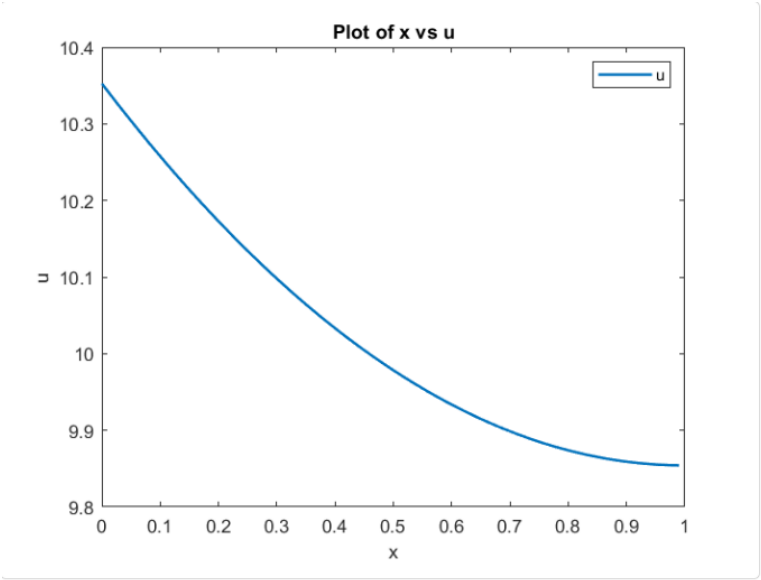


Figure 4.2: Matlab plot

4.2 Problem 4.2

Compare the obtained solution with analytical solution.

$$u(x,t) = \frac{1}{2}(x^2 + 2t) - x + \frac{1}{3} - \frac{2}{\pi^2}\sum_{n=1}^{\infty}\frac{1}{n^2}e^{-n^2\pi^2 t}cos(n\pi x) \tag{4.3}$$

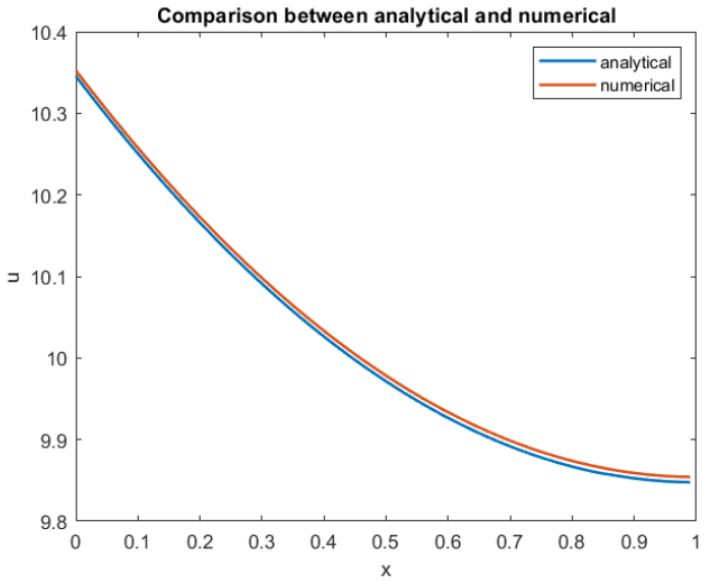


Figure 4.3: Comparison between analytical and numerical solution

Conclusion

In conclusion, the numerical analysis of parabolic equations using the finite element method has provided valuable insights into solving transient heat conduction problems. The presented problems showcased the versatility of the FreeFEM software in handling different boundary conditions and complex scenarios.

The comparison between numerical and analytical solutions in each problem demonstrated the accuracy and reliability of the finite element method. The method effectively captured temperature distributions over time, providing a practical tool for engineering simulations.

Additionally, the grid convergence and time step convergence studies emphasized the importance of selecting appropriate grid sizes and time steps for achieving reliable numerical solutions. These studies contribute to the understanding of the method's convergence behavior and aid in optimizing computational resources.

In summary, the numerical analysis conducted in this report enhances our understanding of parabolic equations and their application in modeling transient heat conduction. The finite element method proves to be a robust approach for solving such problems, offering a balance between accuracy and computational efficiency.