IMPLEMENTATION OF FINITE ELEMENT METHOD FOR PARABOLIC INTERFACE PROBLEM

A Project Report Submitted
in Partial Fulfilment of the Requirements for

Summer Internship

at

Indian Institute Of Technology Guwahati

by
Jinank Jain
(Roll No. UG201210017)
B.Tech, 2nd Year
Computer Science and Engineering
Indian Institute Of Technology Jodhpur

to the

DEPARTMENT OF MATHEMATICS
INDIAN INSTITUTE OF TECHNOLOGY GUWAHATI
GUWAHATI - 781039, INDIA

July, 2014

CERTIFICATE

This is to certify that the work contained in this project report entitled as **"Implementatiom of Finite Element Method for Parabolic Interface Problem" Jinank Jain (Roll No. UG201210017)** to Indian Institute of Technology towards partial requirement of **Summer Internship** which has been carried out by him under my supervision and that it has not been submitted elsewhere for the award of any degree.

Guwahati - 781 039	(Dr. Rajen Kumar Sinha)
July 2014	Project Supervisor

ABSTRACT

Finite Element Method is a numerical method for finding approximate solution to boundary value problems for differntial equation. It uses variational method to minimize an error function and produce a stable solution.

Contents

Chapter 1

The Finite Element Method

1.1 Introduction

From the ancient times, scientists and philosophers have been curious about different physical phenomenon occuring in the nature and have tried to understand and analyze the same. Almost every phenomenon today, whether simple or complex, can be described using the laws of physics with the help of mathematical modeling.

Definition 1.1.1 A mathematical model is a description of a system using mathematical concepts and language. The process of developing a mathematical model is termed *mathematical modeling*.

Most of the practical problems of engineering involve very complex differential and/or integral equations posed on geometrically complicated domains. Solving and analyzing these models analaytically is too complex and will take much longer time. Howerever with the help of a computer ans some numerical methods it can be convenient to analyze these and it also proves to be very useful to analyze the effects of different paramaters on the system effectively.

Definition 1.1.2 The study of algorithms that use numerical approximation for the problems of mathematical analysis is called a *numerical analysis*.

There exists various numerical meyhods to solve the differential equations but the most powerful of these numerical methods is the ***finite element method* (or FEM)**. It is a technique for finding an approximate solution of boundary value and initial value problems characterized by partial differential equation. It produces a stable solution of the problem to minimize the error using the variational method.

1.2 The Basic Idea

The most distinctive feature of finite element method that seperates it from others is the division of a given domain into a set of simple subdomains, called finite elements. Any geometric shape that allows computation of the solution or its approximation, or provides necessary relations among the values of the solution at selected points, called nodes, of the subdomain, qualifies as finite element. Other features of the method include seeking continuous, often polynomial, approximations of the solution over each element in terms of nodal values, and assembly of elements equations by imposing the interelement continuity of the solution and balance of interelement forces.

There are three stages in the whole process where errors are generally introduced in most cases. The first is the partition of the domain into smaller subdomains and then assembling it back to generate the original domain which introduces some errors in the domain during the processs. Second stage is when element equations are derived. The dependent unknowns(u) of the problem are approximated with the idea that any continous function can be represented by a linear combination of unknown functions ϕ_i and undetermined coefficents c_i ($u \approx u_h = \Sigma c_i \phi_i$) . Algebraic relations among the undetermined coefficients c_i are obtained by satisfying the govering equations over each element in a weighted integral snese. The approximation functions ϕ_i are often taken to be polynomails and are derived using the concepts from interpolation theory.

Therefore they are termed as *interpolation functions*. So in the second stage , errors are introduced both in representing the solution u as well as in evaluating the integrals. And lastly errors are introduced in solving the assembled system of equations.

1.3 Implementaion with Analysis

To better understand how to implement the finite element method to a problem, we take an example from [11].

Example 1.3.1. Approximation of the perimeter of a circle.
Consider the problem of determining the perimeter of a circle of radius R without using the formula ($P = 2\pi r$) for the perimater of a circle. Ancient mathematicians used to approximate value of the perimeter by straight line segments as π was not known. Thus, the approximate value of the perimeter is obtained by adding the length pf the line segments used to represent it.

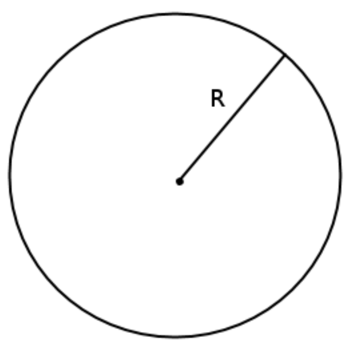

Fig 1.1 A circle of radius R

With the help of this example we outline the basic ideas and steps involved in the finite element analysis of a problem.

1. Finite Element Discretization: First, the perimeter(domain of this problem) is divided into a collection of finite (n) number of subdoamins called line segments. This is called *discretization of the domain.* ome errors would be introduce here because we will need an infinite number of line elements to represent the exact perimeter. Each subdomain(i.e., line segment) is called an *element*. The collection of these elements is called *the finite element mesh*. The points at which elements are connected to each other are called *nodes*. In this case, we discretize the perimeter into a mesh of five line segments making $n = 5$. The mesh is said to be uniform if all the elemnents are of same length; otherwise, it is called a *nonuniform* mesh.

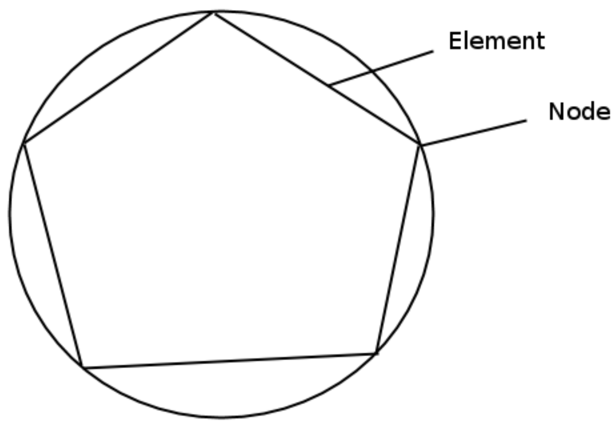

Fig 1.2 Discretization of circle with n=5

2. Element Equation: An element(i.e., line segment, Ω_e) is isolated and its required properties (length in this case) are computed by appropriate means. Let h_e be the of the element Ω_e in the mesh. For a typical Ω_e, h_e is given by

$$h_e = 2R \sin \frac{1}{2}\theta_e \tag{1.1}$$

where R is the radius of the circle and $\theta_e < \pi$ is the angle subtended by the line segement. The above equations are called *element equations.*

3. **Assembly of elements equations and solutions:** The approximate value of the perimeter of the circle is obtained by putting together the element properties in a meaningful way and the process is known as the *assembly of the element equations.* In the present case it follows the property that the perimeter of the polygon Ω_h(circle approximated by assembly of elements) is equal to the sum of the lengths of the individual elements:

$$P_n = \sum_{e=1}^{n} h_e \tag{1.2}$$

Then P_n represents an approximation to the actual perimeter, P. If the mesh is uniform, or h_e is the same for each of the elements in the mesh, then $\theta_e = \frac{2\pi}{n}$, and we have

$$P_n = n\left(2R\sin\frac{\pi}{n}\right) \tag{1.3}$$

4. **Convergence and error estimate:** Since we know the solution to this simple problem ($P = 2\pi R$), we can easily estimate the error in the approximation and show that the approximate solution P_n converges to the exact value P as we increase the number of line segments used to approximate the perimeter(i.e., as $n \to \infty$). Consider a typical element Ω_e. The error in the approximation is equal to the differnce between the length of the arc and that of the line segment

$$E_e = \mid S_e - h_e \mid \tag{1.4}$$

where $S_e = R\theta_e$ is the arc length. Thus the error estimate for an element for an element in the mesh is given by

$$E_e = R\left(\frac{2\pi}{n} - 2\sin\frac{\pi}{n}\right) \tag{1.5}$$

The total error or the global error is given by multiplying E_e by n:

$$E = nE_e = 2R\left(\pi - n\sin\frac{\pi}{n}\right) = 2\pi R - P_n = P - P_n \tag{1.6}$$

We now show that E goes to zero as $n \to \infty$. Letting $x = \frac{1}{n}$, we have

$$P_n = 2Rn\sin\frac{\pi}{n} = 2R\frac{\sin\pi x}{x} \tag{1.7}$$

and

$$\lim_{n\to\infty} P_n = \lim_{n\to\infty}(2R\frac{\sin\pi x}{x}) = 2\pi R \tag{1.8}$$

1.4 Summary

FEM is a numerical method to solve Boundary Value Problem(BVPs)(for e.g, structural and solid mechanics problem in engineering). The fundamental concept behind FEM is that any continous quantity such as temperature, pressure etc. can be approximated by a discrete model composed of a set of piecewise continous polynomial functions defined over a finite number of subdomains (elements). It has applications in areas like heat transfer, fluid mechnaics etc.

Advantages of the finite element method:

- Extensive application: applies to all physical problems in BVP or structural and solid mechanics.
- Application to composite materials: material properties in adjacent do not need to be same.
- Applies to irregularly shaped boundaries as well: any boundary can be approximated using elements with straight sides or matched exactly using elements with curved boundaries.
- Scalable mesh: size of the elements can be vaired allowing the element grid or mesh to be expanded or refined as per the requirement.
- Mixed boundary conditions handling: boundary conditions such as discontinous surface loadings present no difficulties.

Disadvantages and Limitations:

- Gives solution only at nodal points.
- Gives an approximate solution.

Chapter 2

Finite Element Method: Parabolic Interface

2.1 Introduction

The domain which we would consider for our study be rectangular domain $\Omega = (0,2) \times (0,1)$, having an interface at x = 1 such that $\Omega_1 = (0,1) \times (0,1)$ and $\Omega_2 = (1,2) \times (0,1)$. Consider the parabolic interface problem of the form:

$$u_t - \bigtriangledown.(\beta \bigtriangledown u) = f(u) \tag{2.1}$$

where β is different at both side of equation. $\beta_1 = 1$ for $x < 1$ and $\beta_2 = 0.5$ for $x > 1$. The dirchelet boundary conditions are $u(x,0) = 0$, $u(x,1) = 0$, $u(0,y) = 0$ and $u(2,y) = 0$. Time varies from 0 to 0.1 s. The known soultion for u is given by:

$$\begin{aligned} u(x,y,t) &= (exp(\sin t))(\sin \pi x. \sin \pi y) \qquad x < 1 \\ u(x,y,t) &= (exp(\sin t))(\sin 2\pi x. \sin \pi y) \qquad x > 1 \end{aligned} \tag{2.2}$$

The value obtained for f using equation 2.1 and 2.2 is as follows:

$$
\begin{aligned}
f(x,y,t) &= (exp(\sin t))(\sin \pi x.\sin \pi y)(\cos t + 2\pi^2) \qquad x < 1 \\
f(x,y,t) &= (-exp(\sin t))(\sin 2\pi x.\sin \pi y)(\cos t + 2.5\pi^2) \qquad x > 1
\end{aligned}
\tag{2.3}
$$

2.2 Basic Data Structure Scheme

The basic data structure scheme that we are going to follow during all implem-ntaion is as follows:

- For maintaing coordinates of nodal values we have made a file named c4n.mat (c4n stands for coordinates for nodes) which could be easily loaded using a simple octave command load filename.mat.
- For maintaing node numbering of triangular elements which is always done in anticlockwise fashion is stored in file named n4e.mat (n4e stands for nodes for element).

- For maintaining node numbering of dirchelet boundary value points we have made a filenamed n4sDb.mat(where n4sDb stands for nodes for dirchelet boundary)

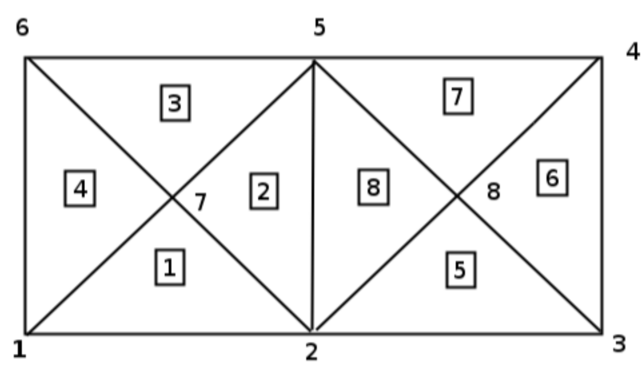

Fig 2.1 Sample mesh

Sample data stored in corresponding files for the given mesh in Fig 2.1

c4n.mat	n4e.mat	n4sDb.mat
0 0	1 2 7	1 2
1 0	2 5 7	2 3
2 0	5 6 7	3 4
2 1	6 1 7	4 5
1 1	2 3 8	5 6
0 1	3 4 8	6 1
0.5 0.5	4 5 8	
1.5 0.5	5 2 8	

2.3 Evaluation of Mass Matrix

In order to evaluate mass matrix you need to follow this algorithm:

Algorithm: Assembly of mass matrix

1: M = size of c4n

2: N = size of n4e

3: Allocate memory of size $M \times M$ to a matrix B and initialize all matrix entries to zero

4: **for** i = 1,2,3....,N **do**

5: Compute the 3×3 local element of mass matrix B^I given by

$$G = \begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix}$$

$$B^I = \frac{1}{24} \times G \times \text{ (Area of } i_{th} \text{ triangular element)}$$

Add B^I_{11} to B_{ii}

Add B^I_{12} to B_{ii+1}

Add B^I_{13} to B_{ii+2}

Add B^I_{21} to B_{i+1i}

Add B^I_{22} to B_{i+1i+1}

Add B^I_{23} to B_{i+1i+2}

Add B^I_{31} to B_{i+2i}

Add B^I_{32} to B_{i+2i+1}

Add B^I_{33} to B_{i+2i+2}

end for

2.4 Evaluation of Stiffness Matrix

In order to evaluate stiffness matrix you need to look into some details about triangulations. For a trinagular element T let (x_1, y_1), (x_2, y_2) and (x_3, y_3) be the vertices of the trinangle and η_1, η_2, and η_3 be the corresponding basis function in S, i.e.,

$$\eta_j(x_k, y_k) = \delta_{jk}, \qquad j, k = 1, 2, 3$$

A moments reflection reveals

$$\eta_j(x, y) = \det \begin{pmatrix} 1 & x & y \\ 1 & x_{j+1} & y_{j+1} \\ 1 & x_{j+2} & y_{j+2} \end{pmatrix} / \det \begin{pmatrix} 1 & x_j & y_j \\ 1 & x_{j+1} & y_{j+1} \\ 1 & x_{j+2} & y_{j+2} \end{pmatrix}$$

whence

$$\nabla \eta_j(x,y) = \tfrac{1}{2|T|} \begin{pmatrix} y_{j+1} - y_{j+2} \\ x_{j+2} - x_{j+1} \end{pmatrix}$$

Here, the indices are to be considered modulo 3, and $|T|$ is the area of T, i.e.,

$$2|T| = \det \begin{pmatrix} x_2 - x_1 & x_3 - x_1 \\ y_2 - y_1 & y_3 - y_1 \end{pmatrix}$$

The resulting entry of stiffness matrix is

$$A_{jk} = \beta \int_T \nabla \eta_j (\nabla \eta_k)^T dx = \beta \tfrac{|T|}{4T^2} (y_{j+1} - y_{j+2}, x_{j+2} - x_{j+1}) \begin{pmatrix} y_{k+1} - y_{k+2} \\ x_{k+2} - x_{k+1} \end{pmatrix}$$

Here is the parameter β is different on both sides of interface. This is written simultaneously for all indices as:

$$A = \tfrac{|T|}{2}.GG^T \text{ with G:=} \begin{pmatrix} 1 & 1 & 1 \\ x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \end{pmatrix}^{-1} \begin{pmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix}$$

Algorithm: Assembly of stifness matrix

1: M = size of c4n

2: N = size of n4e

3: Allocate memory of size $M \times M$ to a matrix A and initialize all matrix entries to zero

4: **for** i = 1,2,3...,N **do**

5: if T is on left hand side then take $\beta = 1$ else $\beta = 0.5$

6: Compute the 3×3 local element of stiffness matrix A^I as explained above with proper value of β.

Add A_{11}^I to A_{ii}
Add A_{12}^I to A_{ii+1}
Add A_{13}^I to A_{ii+2}
Add A_{21}^I to A_{i+1i}
Add A_{22}^I to A_{i+1i+1}
Add A_{23}^I to A_{i+1i+2}
Add A_{31}^I to A_{i+2i}
Add A_{32}^I to A_{i+2i+1}
Add A_{33}^I to A_{i+2i+2}
end for

2.5 Backward Euler Scheme

First of all let us understand what is a basic backward euler scheme. Consider an ordinary differntial equation.

$$\frac{dy}{dt} = f(t, y)$$

with initial value $y(t_o) = y_o$. Here the function f and the initial data t_o and y_o are known; the function y depends on the real variable t and is unknown. A numerical method produces a sequence $y_0, y_1, y_2, ...$ such that y_k is approximates $y(t_o+kh)$, where h is called the step size. The backward Euler method computes the approximations using

$$y_{k+1} = y_k + hf(t_{k+1}, y_{k+1})$$

Now lets see how we implement this scheme for parabolic interface finite element method and see the assembly procedure.

The volume forces are used for assembling the right hand side. Using the value of f in the center of gravity (x_s, y_s) of T the integral $\int_T f\eta_j dx$ is approximated using a proper curvature rule:

$$\int_T f\eta_j dx \approx = \frac{1}{6}\det\begin{pmatrix} x_2 - x_1 & x_3 - x_1 \\ y_2 - y_1 & y_3 - y_1 \end{pmatrix} f(x_s, y_s)$$

Algorithm: Implementation of Backward Euler Scheme

1: M = size of c4n

2: N = size of n4e

3: Initialize b matrix of size M×1, u matrix of size M×1 and U matrix of size M×(number of time steps+1)

4: **for** i = 1 to final time level **do**

5: **for** each triangular element calculate the volume force and add to b.

6: $b = b + B$ * U_{i-1} where B is mass matrix.

7: $b = b$ - $(dt * A + B)$*u

8: $u = (dt * A + B)$ * b^I

9: $U_n = u$

10: **end for**

2.6 Crank Nicklson Scheme

First of all let us understand what is a basic crank nicklson scheme. Consider a partial differential equation:

$$\frac{\partial u}{\partial t} = F(u, x, t, \frac{\partial u}{\partial x}, \frac{\partial^2 u}{\partial x^2})$$

then letting $u(i\triangle x, n\triangle t) = u_i^n$, the equation for CrankNicolson method is a combination of the forward Euler method at n and the backward Euler method at$n+1$ (note, however, that the method itself is not simply the average of those two methods, as the equation has an implicit dependence on the solution):

$$\frac{u_i^{n+1}-u_i^n}{\Delta t} = F_i^n\left(u,\, x,\, t,\, \frac{\partial u}{\partial x},\, \frac{\partial^2 u}{\partial x^2}\right) \qquad \text{(forward Euler)}$$
$$\frac{u_i^{n+1}-u_i^n}{\Delta t} = F_i^{n+1}\left(u,\, x,\, t,\, \frac{\partial u}{\partial x},\, \frac{\partial^2 u}{\partial x^2}\right) \qquad \text{(backward Euler)}$$
$$\frac{u_i^{n+1}-u_i^n}{\Delta t} = \frac{1}{2}\left[F_i^{n+1}\left(u,\, x,\, t,\, \frac{\partial u}{\partial x},\, \frac{\partial^2 u}{\partial x^2}\right) + F_i^n\left(u,\, x,\, t,\, \frac{\partial u}{\partial x},\, \frac{\partial^2 u}{\partial x^2}\right)\right].$$

Now lets see how we implement this scheme for parabolic interface finite element method and see the assembly procedure.

The volume forces are used for assembling the right hand side. Using the value of f in the center of gravity (x_s, y_s) of T the integral $\int_T f\eta_j dx$ is approximated using a proper curvature rule:

$$\int_T f\eta_j dx \approx = \frac{1}{6}\det\begin{pmatrix} x_2 - x_1 & x_3 - x_1 \\ y_2 - y_1 & y_3 - y_1 \end{pmatrix} f(x_s, y_s)$$

Algorithm: Implementation of Crank Nicklson Scheme

1: M = size of c4n

2: N = size of n4e

Initialize b matrix of size M×1, u matrix of size M×1 and U matrix of size M×(number of time steps+1)

for i = 1 to final time level **do**

for each triangular element calculate the volume force and add to b.

$b = b + (B - 0.5 * A * dt) * U_{i-1}$ where B is mass matrix.

$b = b$ - $(B + 0.5 * A * dt) * u$

$u = (B + 0.5 * A * dt) * b^I$

$U_n = u$

end for

Chapter 3

Conclusion

We were able to demonstrate that the experimental order of conergences for various estimators such space, time, $L^2(\Omega)$ and H^1 were in harmony with the theortical results.
The finding of this experiment could be very much useful in study of various two heat conductors which have different heat conductivity or two fluids have different density and we could easily simulate these process with help of code developed during this project.
The things which could be further improved is that if we could deploy some advanced algorithms in refinement techniques so that they could be boost up and we could bring down the complexity of code. At last I would like to conclude on this fact that my project could contribute to some research aspects and could have some new findings.