ISSN (Print) : 232 - 3765
ISSN (Online): 227 - 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 9, September 2014

Formulation of Finite Element Method for 1D and 2D Poisson Equation

Navuday Sharma

PG Student, Dept. of Aerospace and Avionics, Amity University, Noida, Uttar Pradesh, India

ABSTRACT: The Finite Element Method (FEM) introduced by engineers in late 50's and 60's is a numerical technique for solving problems which are described by Ordinary Differential Equations (ODE) /Partial Differential Equations (PDE) with appropriate boundary/initial conditions or to solve problems that can be formulated as a functional minimization. FEM provides greater flexibility to model complex geometries. It can handle general boundary conditions and variable material properties. It has a solid theoretical foundation which gives added reliability and makes it possible to mathematically analyze and estimate the error in the approximate solution. This paper gives an introduction and methodology to solve a PDE using FEM in 1D and 2D in the simplest way possible such that the young researchers who has less mathematical or engineering background can also understand this technique. Only Poisson equation is solved in this paper. The result of the solution the PDE's is also shown computationally using the open source software of FEniCS.

KEYWORDS: FEM 1D, FEM 2D, Partial Differential Equation, Poisson equation, FEniCS

I.INTRODUCTION

Equations like Laplace, Poisson, Navier-stokes appear in various fields like electrostatics, boundary layer theory, aircraft structures etc. So with solutions of such equations, we can model our problems and solve them. To solve such PDE's with FEM some prerequisites are required. Every PDE has strong form and weak form. The strong form of PDE, implies that the relationship must be satisfy at every mathematical point in the domain. Solving the strong form is not always efficient and may not give smooth solutions for a particular problem. Although it may give the accurate result but obtaining the solution is a difficult task. This is true especially in case of complex domains and/or different material interfaces etc. Moreover, incorporating boundary conditions is not so easy.

In order to overcome the above difficulties, weak formulations are preferred. They reduce the continuity requirements on the approximation and allow to construct basis or trial functions which are mostly simple polynomials (generally taken as Lagrange polynomial). Weak forms never give accurate solutions because of the reduction in the requirements of smoothness and weak imposition of boundary conditions. But obtaining the solution becomes an easy task. After that error minimization can be done to remove the inaccuracies obtained in the weak form results.

Improving the accuracy of a solution in weak formulations depend upon the type of problem you are solving. In some cases, for example elliptic problems, only mesh refinement is good enough and but when weak formulations are applied to Stokes and Navier-Stoke flows, one needs to use efficient stabilization techniques, along with mesh refinement, to get accurate results. The accuracy can also be improved by using higher-order basis functions. The whole idea of FEM revolves around choosing the appropriate trial or basis functions. A basis function is an element of a particular basis for a functionspace. Every continuous function in thefunction space can be represented asalinear combination of basis functions, just as every vector in a vector space can be represented as a linear combination of basis vectors.

This paper includes functions in $L^2(I)$ which are known as square-integrable function. It is a real orcomplex-valued measurable function for which the integral of the square of the absolute value is finite. Thus, if

ISSN (Print) : 232 - 3765
ISSN (Online): 227 - 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 9, September 2014

$$\int_{-\infty}^{\infty} |f(x)|^2 \, dx < \infty$$

(P_G): Given f in $L^2(I)$, find u in V such that

a(u, v) = b(v) for all v ϵ V

Then *f* is square integrable on the real line$(-\infty, +\infty)$
The major part of FEM includes the Galerkin idea. Galerkin methods are a class of methods for converting a continuous operator problem (such as a differential equation) to a discrete problem. In this method, we choose trial functions or basis functions $\phi_i(x)$, where $1 \leq i \leq N$ and then we choose test functions (an infinitely differentiable function) $v_i(x)$, where $1 \leq i \leq N$; N = 0, 1, 2, 3……. Very often $v_i(x) = \phi_i(x)$. Thousands of function can be chosen with today's computing power which was not possible earlier. But the functions should be simple polynomials.

II.LITERATURE SURVEY

The Finite Element Method is one of the most important techniques in development of computation methods. It has evolved from solving structural engineering problems to almost every domain in science and technology. FEM algorithms are implemented in Finite Element Analysis and engineering problems are solved using softwares like ANSYS, SAMCEF etc. To analyze frame structures, two classical beam theory are used, namely, Euler-Bernoulli theory and Timoshenko beam theory. The formulation of element stiffness and mass matrices using these two theories is based on FEM. Current research is done in the field of finite element shell analysis. For large structural problems, the structure is divided into many parts and local matrices are being developed for each substructure. The local matrices are then combined to get a global matrix of the system and then error minimization is done. The same methodology is followed in this paper to solve the Poisson Equation.

III.SOLUTION OF TWO POINT BOUNDARY VALUE 1D PROBLEM USING FEM

STEP 1: The Poisson problem given is:
$\Delta u = f$; where f =1 with the boundary conditions given as u(1)= 0 …… (3.1)
The 'u' in above equation is the unknown that we have to find. 'f(x, y)' is the prescribed function and 'v' is the test functions or the virtual displacements. As the finite element idea is to choose some trial functions ϕ, our approximate term 'u' will be a combination of these trial functions. Also we take ϕ_i's = v_i's.

STEP 2: Weak Formulation Method
For a boundary value problem defined using an operator of order 2m, admissible space chosen as $H^m(\Omega)$ along with all essential boundary conditions [1][4] i.e. boundary conditions involving derivatives of u of order <= m-1. In this case we choose the admissible space as $V = H^1_0(I)$ [2][3]. So the boundary condition for the test function would be v(0) = v(1) =1.
Now multiplying the test function 'v' by equation 3.1, we get the weak form.

Where $a(u,v) = \int_0^1 u' v' \, dx, \; for \; all \; u, v \; \epsilon \; V$

$b(v) = \int_0^1 v \; dx, \;\; for \; all \; v \; \epsilon \; V$

STEP 3: Galerkin Finite Element Problem
Firstly let's create a finite dimensional sub space V_h of V. For any positive integer N+1, let $I = \{0 = x_0 < x_1 < \ldots x_{N+1} = 1\}$ be a partition of I into subintervals (finite elements) $I_j = (x_j, x_{j-1})$, $1 \leq i \leq N+1$, with length $h = x_j - x_{j-1}$. The discrete solution will be sought in V_h defined by:
$V_h = \{v_h: \; v_h \; \epsilon \; C^0(I), \; v_h \; \epsilon \; P_1(I_j), \; 1 \leq i \leq N+1, \; v_h(0) = v_h(1) = 0$

10.15662/ijareeie.2014.0309001

ISSN (Print) : 232 - 3765
ISSN (Online): 227 - 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 9, September 2014

- V_h is a subspace of V.
- V_h is finite dimensional, since on each subinterval I_j, $v_h \in P_1(I_j)$; i.e. we have 2(N+1) degrees of freedom. Continuity constraints at the nodal points $x_1, \ldots\ldots, x_N$ and boundary conditions $v_h(0) = v_h(1) = 0$ imply

dim V_h = 2(N+1) – N = N.

This is the simplest choice for V_h. However other choices are also possible.

The Galerkin finite element problem (P^h_G) corresponding to (P_G) can be defined as:

STEP 4: Construction of the Basis Functions

(P^h_G): Given f in $L^2(I)$, find u_h in V_h such that

$a(u_h, v_h) = b(v_h)$ for all $u_h \in V_h$ i.e.

$$\int_0^1 u_h' v_h' \, dx = \int_0^1 v_h' \, dx \quad for\ all\ u, v \in V$$

$$\phi^i_h(x_j) = \begin{cases} \{(x-x_{i-1})/(x_i-x_{i-1})\} & \text{in } [x_{i-1}, x_i] \\ \{(x-x_{i+1})/(x_i-x_{i+1})\} & \text{in } [x_i, x_{i+1}] \\ \{0\} & \text{otherwise} \end{cases}$$

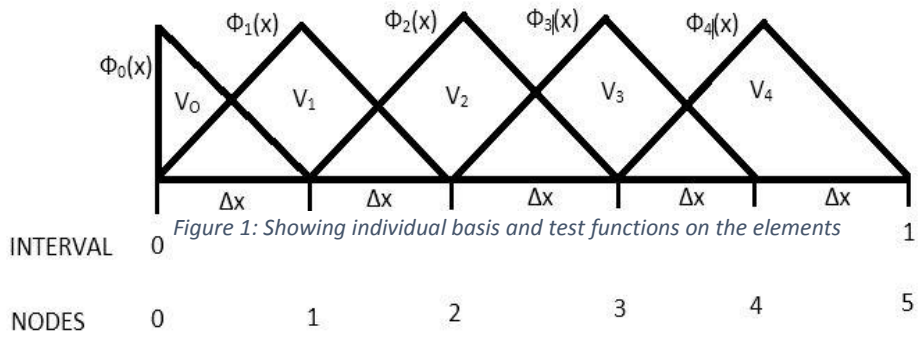


Figure 1: Showing individual basis and test functions on the elements

The main job is to create the basis function which will be the main step for creating the Matlab code. We take a 5 node grid in 1D and create the piecewise linear elements as shown in the figure 1. Although quadratic, parabolic etc elements can also be chosen for increasing the accuracy of the obtained solution but the linear elements make the calculations simpler. For each element, we have a different basis function and test function. Please note that we take $\boldsymbol{\phi_i}$**'s =** $\boldsymbol{v_i}$**'s** i.e. trial function = test function respectively. The distance between the two nodes is Δx.

Now we define the basis function for our current problem as:

$\Phi_0 = \frac{1}{\Delta x}$ in [0, 1] node and 0 otherwise

$\phi_1 = \frac{1}{\Delta x}$ in [0, 1], $\frac{-1}{\Delta x}$ in [1, 2] and 0 otherwise

$\phi_2 = \frac{1}{\Delta x}$ in [1, 2], $\frac{-1}{\Delta x}$ in [2, 3] and 0 otherwise

$\phi_3 = \frac{1}{\Delta x}$ in [2, 3], $\frac{-1}{\Delta x}$ in [3, 4] and 0 otherwise

ISSN (Print) : 232 - 3765
ISSN (Online): 227 - 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 9, September 2014

$\phi_4 = \frac{1}{\Delta x}$ in [3, 4], $\frac{-1}{\Delta x}$ in [4, 5] and 0 otherwise
Now as we know $u_h = \sum_{i=1}^{N} u_h(x_i)\, \phi^i_h(x) = \sum_{i=1}^{N} u_i\, \phi^i_h(x)$
This implies that $u_h = u_0\, \Phi_0(x) + u_1\, \phi_1(x) + u_2\, \phi_2(x) + u_3\, \phi_3(x) + u_4\, \phi_4(x)$

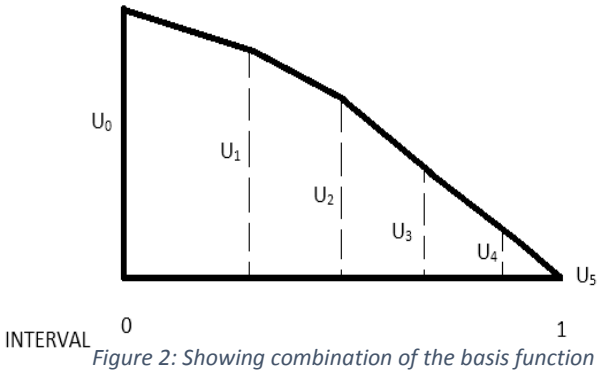


Figure 2: Showing combination of the basis function

The figure 2 shows the typical combination of the basis function. At node 0, only ϕ_0 is present, so maximum height is u_0. At node 1, only ϕ_1 is present, so the height is u_1. Coefficients of ϕ have a physical meaning. They are displacements at the node. We just need to find 5 unknown ϕ_i's to get u's. Now in the weak form a(u, v) = b(v) i.e.

$\int_0^1 u_h' v_h' dx = \int_0^1 v_{hi}' dx \; for\ all\ u, v \in V$, i = 0,1,2,3,4 …… (3.2)
$u_h' = u_0\, \Phi_0'(x) + u_1\, \phi_1'(x) + u_2\, \phi_2'(x)\, u_3\, \phi_3'(x) + u_4\, \phi_4'(x)$
Now, let's find the RHS of the equation 3.2,
$\int_0^1 v_{h0} = \int_0^1 \phi_0 = \frac{1}{2} * \Delta x * 1$
Please note here that ϕ_i's = v_i's and from figure 1, ϕ_{h0} exist between the nodes [0, Δx] and integration of ϕ_{h0} element will be the area under the triangular element. Similarly,
$\int_0^1 v_{h1} = \int_0^1 \phi_1 = \frac{1}{2} * \Delta x * 2$
$\int_0^1 v_{h2} = \int_0^1 \phi_2 = \frac{1}{2} * \Delta x * 2$
$\int_0^1 v_{h3} = \int_0^1 \phi_3 = \frac{1}{2} * \Delta x * 2$
$\int_0^1 v_{h4} = \int_0^1 \phi_4 = \frac{1}{2} * \Delta x * 2$
Now the derivatives of the above values can be substituted in equation 3.2 and 'u' can be found.

STEP 5: WRITING THE SOLUTION IN MATRIX FORM
The matrix required is KU = F
$K = [(\phi^i_h, \phi^j_h)]_{1 \le i,j \le N}$
$U = (U_i)_{1 \le i \le N}$
$F = (b(\phi^j_h))_{1 \le j \le N}$

ISSN (Print) : 232 - 3765
ISSN (Online): 227 - 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 9, September 2014

$$\begin{bmatrix} K_{00} & K_{01} & K_{02} & K_{03} & K_{04} \\ K_{10} & K_{11} & K_{12} & K_{13} & K_{14} \\ K_{20} & K_{21} & K_{22} & K_{23} & K_{24} \\ K_{30} & K_{31} & K_{32} & K_{33} & K_{34} \\ K_{40} & K_{41} & K_{42} & K_{43} & K_{44} \end{bmatrix} \begin{bmatrix} u_0 \\ u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} F_0 \\ F_1 \\ F_2 \\ F_3 \\ F_4 \end{bmatrix}$$

$$\begin{bmatrix} 1 & -1 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & -1 & 2 \end{bmatrix} \begin{bmatrix} u_0 \\ u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \Delta x \begin{bmatrix} \frac{1}{2} \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

Figure 3: Domain taken for the 2D problem

IV.FINITE ELEMENT METHOD IN 2D

FEM is actually used for solving 2D problems. If a problem is given in 1D with some boundary conditions, it could be integrated simply and boundary conditions can be imposed. But when a 2D problem is given, then FEM is required. This paper presents FEM in 1D, just to explain the methodology of FEM. Finite element method formulation in 2D would be same as in 1D. The only difference is, we have to make the mesh in a plane instead of making the elements in 1D. We can use linear, quadratic or cubic functions for constructing the mesh.
The most important part of FEM is choosing the trial functions. We chose simple functions which are easy polynomials on each element. The elements chosen for 1D FEM were linear but in 2D we choose triangular elements. In 1D, when we were using linear elements, we had triangular basis function but in 2D we will have pyramidal basis function. At any node in the 2D mesh, the pyramid function ϕ_1 is 1 and zero at all the other points just like we do in hat functions (basis function in 1D).

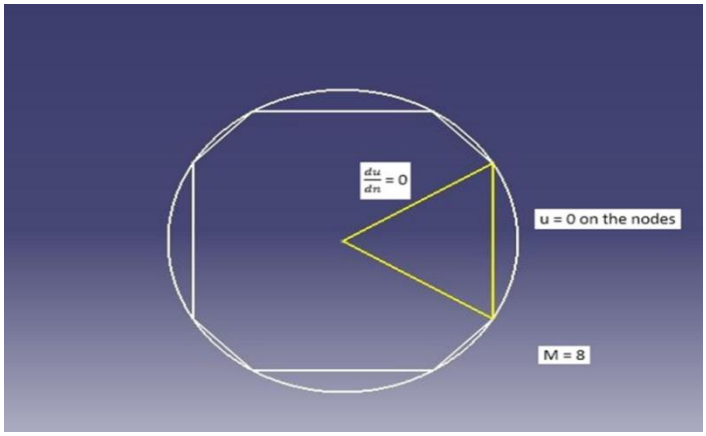


Figure 4: Domain chosen due to rotational symmetry

ISSN (Print) : 232 - 3765
ISSN (Online): 227 - 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 9, September 2014

$$u = u_1\phi_1(x, y) + u_2\phi_2(x, y) + \ldots. u_n\phi_n(x, y)$$

Let's see a Poisson problem in 2D.

$$-u_{xx} - u_{yy} = 4$$

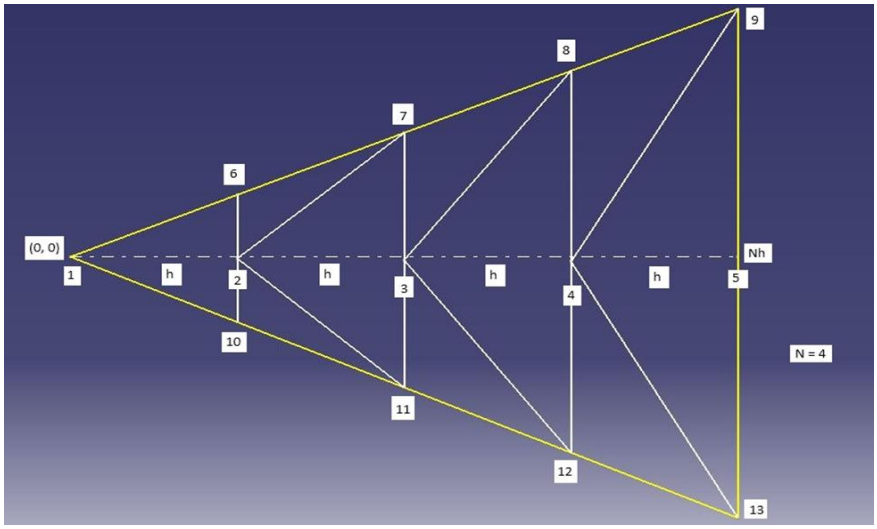
Boundary condition: u =

…… (4.1)

Let's take a circular domain to solve the problem. In the circular domain, we have to approximate the curved boundaries by straight lines. That would make our domain as a polygon. So draw the polygon with about eight sides and u=0 is true on the eight nodes. Now let's divide our domain into 8 triangles or M triangles for M sided polygon. Now we need to work only on one triangle. By rotational symmetry, we'll have the same results for all the triangles. So our domain becomes just one triangle. In that triangle, we have 'u = 0' boundary condition at two nodes and on the edge of the triangles we have Neumann boundary condition, $\frac{du}{dn} = 0$.
Now the coordinates of the triangle given in figure 3 can be calculated.

Now we create the mesh on the triangle. For that, we divide the center line of the triangle into N parts with distance between each part as 'h'. Using these N points on the line, we form the mesh as shown in figure 4 and number the nodes and the triangles.

Now if we can get the coordinates of the mesh points, we could put them in a code and solve the problem using FEM. If we use N = 4, then we would get 13 nodes and 14 triangles and we know the coordinates of every node. We can create a Matlab code for solving the problem using FEM. The code will be needing the list of the coordinates of the nodes (P) and the list of the triangles (T). The triangle tells us the threenode numbers and the list P gives us their positions.



P = (0, 0), (h, 0), (2h, 0)……
T = (1, 2, 6), (2, 7, 6)…….
Now the Matlab code will create the stiffness matrix 'K' and put in the boundary conditions i.e. u = 0 at node 3, 5, 9, which implies that at the whole edge, u = 0. After that, we can solve the matrix, KU = F. The code is creating 'K' and 'F' for us.
This is how the mesh is created and the Matlab code works for us. Now for better accuracy, we could have chosen P_2 elements instead of P_1 elements or we could have increased M i.e. number of triangles would have increased.

ISSN (Print) : 232 - 3765
ISSN (Online): 227 - 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 9, September 2014

Equation (4.1) is the strong form of our problem. Now for the weak form, we have to multiply it with a test function and integrate by parts.

$$\iint_\Omega (-u_{xx} - u_{yy})*v(x,y) = \iint_\Omega f(x,y)*v(x,y)\, dx\, dy$$

$$\iint_I \left(\frac{du}{dx} * \frac{dv}{dx} + \frac{du}{dy} * \frac{dv}{dy}\right) dxdy = \iint_I f(x,y) * v(x,y) dxdy$$

for all v(x,y) ……..…….. (4.2)

Here we are throwing one derivate of x and y onto v.
Now we have found out the weak form and we come to the finite element idea. Finite element idea is to choose the trial functions and write 'u' as a linear combination of the trial functions.
$U = u_1\phi_1(x, y) + u_2\phi_2(x, y) + \ldots\ldots\ldots\ldots\ldots u_n\phi_n(x, y)$
In our problem, we will have 13 nodes in the triangle, which is our domain. So we will have 13 trial or basis functions. We will also choose our test functions to be the same as the trial functions. Also we are working on 13 dimensions instead of infinite dimensions. For this finite element subspace or piecewise linear polynomial subspace, we plug the trial functions and the test functions in the weak form (equation 4.2). In this method, we find entries of K matrix one by one.

The other method to do this is take the elements one by one. The element will have two trial functions and so we make 2X2 local stiffness matrices. These local stiffness matrices are then combined to produce the global stiffness matrix 'K'. This is how the code will be executed in Matlab.
In earlier example, we showed, how FEM 2D is executed in the computer using a Matlab code. Now let's see, how it is done theoretically. Let us take another problem to understand the concept.

A.Problem

(P_c): $-\Delta u + u = f$ in $\Omega = (0, 1) \times (0, 1) \subset R^2$

$\frac{du}{dn} = 0$ over Γ, a square boundary of Ω. …….….. (4.3)

Here equation (4.3) is the strong form, $\Delta u = \frac{d^2u}{dx^2} + \frac{d^2u}{dy^2}$ and $\frac{du}{dn}$ is the normal derivative of 'u' in the direction of the exterior normal to Γ. Now we have to define our admissible space in which our solution and the boundary conditions exist. For defining our admissible space, we multiply our strong form with a test function and integrate it by parts. Then we get the weak form as follows.

Weak Form:

$a(u, v) = \int_\Omega (-\Delta u + u)v\, d\Omega = \int_\Omega (u'v' + uv)\, d\Omega$ for all u, v ϵ V

$L(v) = \int_\Omega fv\, d\Omega$ for all v ϵ V

From the weak form, we decide the admissible space we need,

$V = H^1(\Omega) = \{v: v \epsilon L^2(\Omega), \frac{dv}{dx}, \frac{dv}{dy} \epsilon L^2(\Omega)$

Now the Galerkin Variational Problem (P_G) corresponding to P_C will be

ISSN (Print) : 232 - 3765
ISSN (Online): 227 - 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 9, September 2014

(P_G): Given f ϵ $L^2(\Omega)$, find u ϵ V such that

a(u, v) = L(v) for all v ϵ V

Triangulation:
Triangulation [5] means subdivision of the domain into closed quadrilaterals, rectangles, triangles etc denoted by $\{T_i\}_{i=1}^{NELEM}$ o we do admissible triangulation 't_h' of closed domain $\Omega \subset R^2$.

$$(i)\ \bar{\Omega} = \cup_{i=1}^{NELEM} T_i$$

$$(ii)\ T_i \cap T_j = \begin{cases} \phi \\ \text{common vertex} \\ \text{common side} \end{cases} \quad \text{for } i \neq j$$

$$(iii)\ T_i^{\circ} \cap T_j^{\circ} = \phi \text{ for } i \neq j$$

Note that, if we have a curved domain then we approximate it with a polygon for simplicity. Now we introduce the mesh parameter,
h = max (diam T) where (diam T) is the longest side of the element chosen i.e. triangle in this case and T ϵ t_h.

Finite dimensional subspace:
To the triangulation t_h of Ω, we associate a finite dimensional linear space V_h of continuous piecewise polynomials of degree<= 1 in each triangle T of t_h, i.e.,

$V_h = \{v_h: v_h \in C^0(\Omega), v_h \in P_1(T) \text{ for all } T \in t_h$

In this example dimension of V_h is N, where N is the number of the nodes in the triangulation. After defining the sub-space V_h, our Galerkin finite element problem becomes

(P^h_G): Given f ϵ $L^2(\Omega)$, find u_h ϵ V_h such that

$a(u_h, v_h) = L(v_h)$ for all $v_h \in V_h$

Matrix Formulation:
For making the stiffness matrix 'K', we need to choose the appropriate trial functions or the basis functions $\{\phi^i_h\}$ 1<= i<=N for V_h. After finding the trial functions, we can find the approximate solution 'u_h' as,
where u_i's need to be determined.

$u_h = \sum_{i=1}^{N} u_i \phi^i_h$

Please note that we take the test functions same as the trial functions, as we did earlier. As we did earlier, we have
$a(u_h, v_h) = L(v_h)$ for all $v_h \in V_h$
$a(\sum_{i=1}^{N} u_i \phi^i_h, \phi^j_h) = L(\phi^j_h)$, 1<= i,j <=N
Therefore,
$\sum_{i}^{N} u_{i=1}\ a(\phi^i_h, \phi^j_h) = L(\phi^j_h)$, 1<= i,j <=N

$K = [a(\phi^i_h, \phi^j_h)]$, 1<= i,j <=N

$F = L(\phi^j_h)$, 1<= i,j <=N

$U = [u_1, u_2 \ldots\ldots\ldots\ldots u_N]$

ISSN (Print) : 232 - 3765
ISSN (Online): 227 - 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 9, September 2014

Therefore we got the matrix KU = F where,is the global stiffness matrix K. is the global load vector.
is the unknown vector.

Construction of Basis Function:

In this example, say we take the dimension of V_h as 5, i.e. we have 5 nodes in our domain as shown in the figure 5. We have approximated our curved circular boundary with a rectangle and divided into 4 triangles. Accuracy of our approximate solution u_h will be very less with this domain. For less error, we should take a higher polygon and divide it into more triangles.

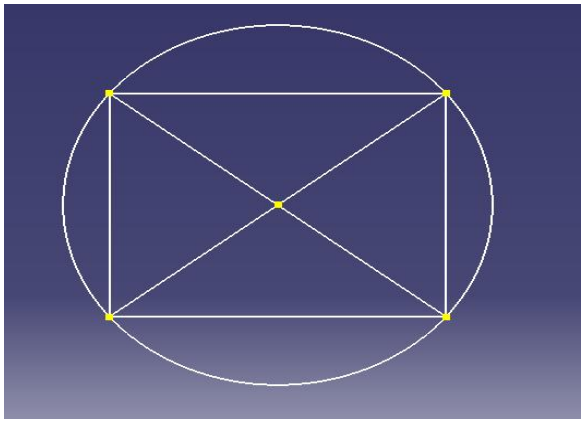


Figure 5: Circular domain chosen for the problem

To each global node a_i ($1 <= i <= N$), we associate a pyramidal function 'ϕ^i_h' as explained before. This pyramidal function has a unit height over the triangles containing the node a_i and vanishes over the triangles not containing a_i as one of their nodes. This implies,

- $\phi^i_h(a_i) = 1$, $\phi^i_h(a_j) = 0$ for all $i \neq j$.
- Height of pyramidal function is 1 at a_i.
- $\phi^i_h \in P_1(T)$ for all $T \in t_h$.

For construction of the global basis function, we need to use barycentric coordinates for a triangle. We define a closed convex hull T by,

$T = \{x: x \in R^2, x = \sum_{i=1}^{3} \lambda_i a_i, 0 <= \lambda_i <= 1, \sum_{i=1}^{3} \lambda_i = 1\}$

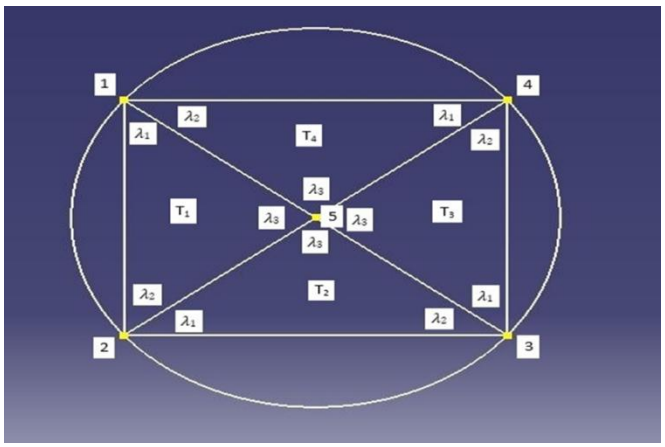


Figure 6: Representation of Barycentric Coordinates

Where λ_i is the barycentric coordinates of any point $x \in T$ and a_i are the vertices of a particular triangle chosen. Now any point x, can be represented using the barycentric coordinates.

10.15662/ijareeie.2014.0309021

Figure 7: Representation of Barycentric coordinates

ISSN (Print) : 232 - 3765
ISSN (Online): 227 - 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 9, September 2014

$x = \lambda_1 a_1 + \lambda_2 a_2 + \lambda_3 a_3$
In matrix form, above equation can be written as,

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \lambda_1 \begin{bmatrix} a_{11} \\ a_{12} \end{bmatrix} + \lambda_2 \begin{bmatrix} a_{21} \\ a_{22} \end{bmatrix} + \lambda_3 \begin{bmatrix} a_{31} \\ a_{32} \end{bmatrix}$$

i.e.
$x_1 = \lambda_1 a_{11} + \lambda_2 a_{21} + \lambda_3 a_{31}$
$x_2 = \lambda_1 a_{12} + \lambda_2 a_{22} + \lambda_3 a_{32}$
$\lambda_1 + \lambda_2 + \lambda_3 = 1$
Please note that this calculation is done for only one triangle out of many in the domain. For each triangle, we have to find individual barycentric coordinates. Then all these barycentric coordinates will be used to find thepyramidal functions. Please refer figure 6 to understand the individual barycentric coordinates.

So any point $x \epsilon T$ can be found using the barycentric coordinates. For any x, not belonging to T, representation of that point using the barycentric coordinates is not possible. Also barycentric coordinates can be found, if we know any $x \epsilon T$, or barycentric coordinates can be written as a function of x and y in 2D.
$x = \lambda_1 a_{11} + \lambda_2 a_{21} + \lambda_3 a_{31}$
$y = \lambda_1 a_{12} + \lambda_2 a_{22} + \lambda_3 a_{32}$
$\lambda_1 + \lambda_2 + \lambda_3 = 1$
Suppose we have a triangle in the domain with the coordinates, $a_1 = (1, 0)$, $a_2 = (0, 1)$, $a_3 = (0, 0)$, the barycentric coordinates obtained will be,
$\lambda_1(x, y) = x$
$\lambda_2(x, y) = y$
$\lambda_3(x, y) = 1\text{-}x\text{-}y$
The barycentric co-ordinates in triangles will be used now to define global basis functions $\{\phi^i_h\}$ $1 <= i <= N$ for V_h. Consider the example we are taking and refer figure 6.
$\Phi^1_h = \lambda_1$ in triangle T_1
λ_2 in T_4
0 otherwise
$\Phi^2_h = \lambda_1$ in T_2
λ_2 in T_1
0 otherwise
$\Phi^3_h = \lambda_2$ in T_2
λ_1 in T_3
0 otherwise
$\Phi^4_h = \lambda_2$ in T_3
λ_1 in T_4
0 otherwise
$\Phi^5_h = \lambda_3$ in T_1, T_2, T_3, T_4
0 otherwise

These are the pyramidal functions. We will put the values of barycentric coordinates that we found earlier. Now since we have 5 basis functions, we'll get a 5X5 stiffness matrix. Now we know,
$K = [a(\phi^i_h, \phi^j_h)]$, $1 <= i,j <= N$

$$k_{ij} = a(\phi^i_h, \phi^j_h) = \int_\Omega \left[\frac{\partial \phi^i_h}{\partial x} \frac{\partial \phi^j_h}{\partial x} + \frac{\partial \phi^i_h}{\partial y} \frac{\partial \phi^j_h}{\partial y} + \phi^i_h \phi^j_h \right] d\Omega$$

During implementation, we first compute the element stiffness matrices and the element load vectors (here element refers to a triangle in the domain). Then we assemble the element matrices to obtain the global stiffness matrix and the global load vector.

ISSN (Print) : 232 - 3765
ISSN (Online): 227 - 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 9, September 2014

V.RESULT AND DISCUSSION

For the same problem of Poisson equation, solution has been computed with FEniCS [6],

- for Dirichlet boundary condition shown in figure 7.
- for Dirichlet and Neumann boundary condition shown in figure 8.

$-\Delta u = f$ in Ω with boundary condition

$u = u_0$ in $d\Omega$

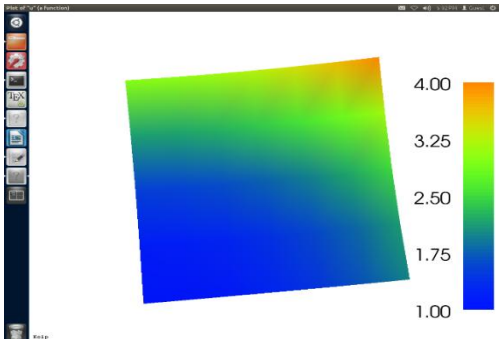


Figure 7: Solution for problem with Dirichlet boundary condition

$\Delta u = f$ in Ω

$u = 0$ on Γ_D

$u'.n = g$ on Γ_N

Where
$\Omega = [0, 1] \times [0, 1]$ (a unit square)
$\Gamma_D = \{(0, y) \cup (1, y) \subset d\Omega\}$ (Dirichlet boundary condition)
$\Gamma_N = \{(x, 0) \cup (x, 1) \subset d\Omega\}$ (Neumann boundary condition)
$g = \sin(2x)$ (normal derivative)
$f = 10x^2 + 3y^3$ (source term)

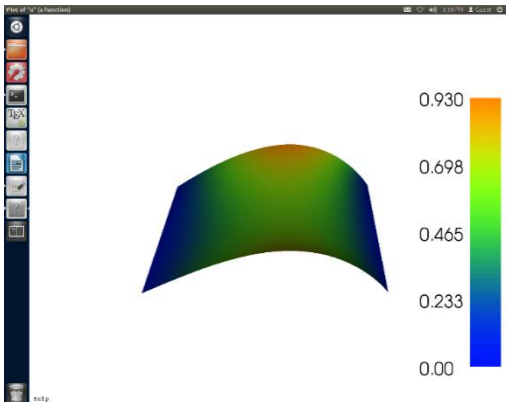


Figure 8: Solution for problem with Dirichlet and Neumann boundary condition

ISSN (Print) : 232 - 3765
ISSN (Online): 227 - 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 9, September 2014

For working with FEniCS, knowledge of python coding is required. Figure 7 and 8 shows the value of 'u' at various points in the domain. For obtaining the solution at a particular line or point in the domain, we can use the software's like 'ParaView' and 'Visit'. PDE computation can also be done by various other softwares like Freefem++ etc.

VI.CONCLUSION

This paper presents the basic understanding of Finite Element Method and the methodology to solve any problem of differential equation. The main idea of Finite Element Method is to choose the appropriate basis functions and then expressing the unknown as combination of the basis functions. Finally a stiffness matrix is generated and solution is obtained. The accuracy of solution increases by using higher order polynomials for the basis function but the calculations become difficult and hence the computation time also increases. Finite Element Method can be executed computationally on Matlab, FEniCS, FreeFem++ etc.

ACKNOWLEDGEMENT

The author is grateful to Prof. Mythily Ramaswamy, Tata Institute of Fundamental Research Centre for Applied Mathematics, Prof. Neela Nataraj, IIT Bombay and Post Doc Saumya Bajpai, for providing continuous support and guidance in understanding of Finite Element Method.

REFERENCES

1. Mark S. Gockenbach, "Understanding and implementing the Finite Element Method", pp. 29-31, 2006
2. Mark S. Gockenbach, "Understanding and implementing the Finite Element Method", pp. 39, 2006
3. Bertrand Mercier, "Lectures on Topics in Finite Element Solution of Elliptical Problems", Tata Institute of Fundamental Research Bombay, pp. 1-8, 1979
4. Bertrand Mercier, "Lectures on Topics in Finite Element Solution of Elliptical Problems", Tata Institute of Fundamental Research Bombay, pp. 11-18, 1979
5. Neela Nataraj, "Introduction to Finite Element Method Lecture", Department of Mathematics, Indian Institute of Technology Bombay
6. Anders Logg and Garth N. Wells, "Lecture Notes in Computational Science and Engineering", The FEniCS book