Finite Element Method Course No. 2D1260

Implementation of FEM for Poissions Equation

COMPUTER ASSIGNMENT 2

Submitted by: Mayur Pal & Ole Kristian.

Department of Numerical Analysis and Computer Science, NADA
The Royal Institute of Technology
Stockholm, Sweden

February- 2004-01-29

Abstract

The numerical methods based on finite difference approximation are quite efficient in solving the ordinary differential equations in one and two dimension whereas for sovling partial differential equation in more than two dimension an advance methodology is required. The finite elements method (FEM) is an important and efficient numerical analysis tool used in solving partial differential equations in higher dimensions. In FEM the representation of the differential equation is done in integrated form and then approximating this integrated form using a selected set of basis functions also called as ansatz function which is defined on the domain of the problem. This results in a linear system of equations which could be solved to obtain an approximate solution to the differential equation. The linear system of equations so obtained is normally very large. Hence it requires advance computational algebra to solve this system of equations. In FEM the basis functions are selected in such a way that the system is sparse. This allows FEM to compete with Finite Difference methods in terms of computational efficiency. In this project the Finite Elements Method for the Poisson equation is implemented using MATLAB. Here we have tried to make our program more interactive my adding GUI to it, The user is asked to fill-in the specified Requirements for obtaining the FEM solution through graphical user interface. The use of GUI makes the program more handy. Since the linear system resulting from the FEM problem is sparse, symmetric and positive definite, the use of the preconditioned conjugate gradient method reduces the total required computational time. Thus, in addition to the main FEM routine, a program to solve a sparse, symmetric positive definite matrix using preconditioned conjugate gradients method is also implemented.

FEM

The Finite Elements Method is a numerical method for solving differential equations. It is specially applied for solving partial differential equations. The objective of this project is to implement the Finite Elements Method (FEM) to solve the Poisson equation:

$$\nabla \cdot (k(x,y)\nabla u(x,y)) = f(x,y), \ (x,y) \in \Omega \qquad (1)$$
$$u \mid_{\partial\Omega} = g, \ (x,y) \in \partial\Omega$$

on a rectangular domain. The solution obtained using FEM is approximate and is denoted by the capital letter U. In FEM the approximate solution U of the differential equation is expressed as a sum of basis functions.

$$U(x,y) = \sum a_i \phi_i (x,y) \qquad (2)$$

where ϕ_i is the i^{th} basis function and a_i is its coefficient.

The problem of FEM is then to select a convenient set of basis functions ϕ_i and to determine the corresponding coefficient a_i so that the approximate solution $U(x,y) = \sum a_i \phi_i$ is close to the true solution u(x,y).The determination of the coefficients a_i is thus essentially an error minimization problem. Two methods are popular, namely **the Ritz procedure and Galerkin's Method.**

In the Ritz procedure, the differential equation is transformed into a minimization problem. A functional, an integral expression whose minimization yields the solution of the differential equation, is first derived. The approximate solution U is then substituted in the functional and a minimization is performed with respect to the coefficients a_i. This will yield a system of equations whose solution gives a_i.

The Ritz procedure can not be applied for differential equations that have no minimum principle. A more **general method is the Galerkin's Method.** In Galerkin's method the variational form of the differential equation is used. The variational form of a PDE is obtained by multiplying the PDE with a test function, integrating over the domain and applying integration by parts to simplify the expression.[1] Once the expression is obtained Galerkin's orthogonality principle can be applied to determine the coefficients a_i. This means $\mathbf{U = \sum a_i \phi_i}$ must satisfy the variational equation for each basis function ϕ_i. This yields enough number of equations to solve for all coefficients. (For a full discussion of the Galerkin and Ritz methods and the principle behind them one is referred to FEM text books like [1] or [3])

Both **Galerkin's and Ritz methods yield a system of linear equations** that must be solved to obtain the coefficients a_i. The systems of equations that arise as a result of these methods are usually very large. The computational cost of solving these systems is greatly reduced if the system is sparse. The selection of the basis function is thus motivated by the desire to get a sparse final linear system of equations. The roof functions (piecewise linear polynomials) are used in FEM for this purpose.

These roof functions are also called sometimes as Hat Functions due to their peculiar shape. The roof functions require the domain to be decomposed into small parts. In a two dimensional problem like the one this project is trying to solve requires generation of a mesh on the domain. Each basis function is associated with a node in the mesh. The basis function is such that it has a value of one on that node and zero everywhere else. The coupling between the basis functions is minimized and this results in a sparse linear system of equations.

The mesh that is used in FEM is usually unstructured i.e. there is no regularity in the elements of the mesh. With the roof functions each node of the mesh has one and exactly one basis function associated with it. The usual practice of obtaining the values of the basis functions and their derivatives, which are needed in the computation of the solution, is to use mapping. A standard element with known shape and coordinate is used to obtain the basis functions and their derivatives and a mapping is used from the local coordinate of the standard element to the global coordinate (actual position) of each of the elements in the mesh. Usually quadrilateral or triangular elements are used and a type of mapping called isoparametric mapping is used. This scheme is especially well suited for computer implementation. The FEM deals with integrated form of the differential equation. Thus the computation process involves the evaluation of a large number of integrals. The more general and easy to numerically compute the integrals arising in the solution process of FEM. The **Gaussian quadrature** schemes are popular in this respect and will be used in this project.

Formulation of FEM

The first step in the solution of a partial differential equation using FEM is to derive the variational form of the problem. This is done by multiplying both sides of the PDE with a function and integrating over the domain of definition of the problem. After simplification using integration by parts the following weak form of the differential equation (1) is obtained.

$$\int k\nabla u \,.\nabla v \, d\Omega \;=\; -\int fv \, d\Omega \qquad (3)$$

According to Galerkin, in the finite element solution u is substituted by U and v is substituted by ϕ_i. This yields a linear system of equations

$$\mathbf{S\,a = f} \qquad (4)$$

Where

$$(S)_{ij} = \int k\nabla\phi_i \,.\nabla\phi_j \, \mathbf{dx} \quad \text{and} \quad (\mathbf{f})_i = \int f\,\phi_i \, \mathbf{dx}$$

The above formula may change if there are Neumann boundary conditions

$\mathbf{a} = [a_1, a_2, a_3, \ldots.]^T$

A better notation is

$\mathbf{S} = \int k\mathbf{B}^T\mathbf{B}\, d\mathbf{x}; \quad \mathbf{f} = \int \boldsymbol{\phi}^T f\, d\mathbf{x}$,

where $\boldsymbol{\phi} = [\phi_1, \phi_2, \phi_3, \ldots]^T$

and $\mathbf{B} = \begin{bmatrix} \partial\phi_1/\partial x_1 & \partial\phi 2/\partial x_1 \ldots \\ \partial\phi_1/\partial x_2 & \partial\phi_2/\partial x_2 \ldots \end{bmatrix}$

For performing the integration using the local coordinates, we also need to have the Jacobian matrix J of the mapping. In this case the integration is done on the local coordinates and the mapping handles the transformation into the global system. For each element a local stiffness matrix and load vector is computed and this is assembled in the global matrix. The formula for the local stiffness matrix in the FEM formulation is

$\mathbf{S} = \iint k\mathbf{B}^T\mathbf{J}^{-T}\mathbf{J}^{-1}\mathbf{B}\, \det(\mathbf{J})\, d\xi\, d\eta$

Basis Functions and Isoparametric Mapping

In this Project two different element types are used. These are quadrilateral and triangular elements. The basis functions to be used depend on the element type and hence they are separately derived.

Quadrilateral Elements

For the quadrilateral element type we the standard element shown in the figure below.

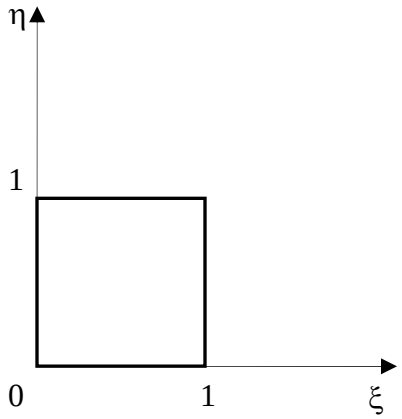


Figure 1: The Standard Quadrilateral Element

The basis functions are derived using Lagrange Polynomials. Thus the above standard quadrilateral can be described by the following four basis function.

$\varphi_1(\xi,\eta) = l^1_1(\xi)l^1_1(\eta) = (\xi-1)(\eta-1)$
$\varphi_2(\xi,\eta) = l^1_2(\xi)l^1_1(\eta) = \xi(1-\eta)$
$\varphi_3(\xi,\eta) = l^1_2(\xi)l^1_2(\eta) = \xi\eta$
$\varphi_4(\xi,\eta) = l^1_1(\xi)l^1_2(\eta) = (1-\xi)\eta$

Hence the B matrix and the Jacobian for mapping is as follows

B = $\begin{bmatrix} \eta\text{-}1 & 1\text{-}\eta & \eta & -\eta \\ \xi\text{-}1 & -\xi & \xi & 1\text{-}\xi \end{bmatrix}$

J = $\begin{bmatrix} (X_1 - X_2 + X_3 - X_4)\,\eta - X_1 + X_2 & (Y_1 - Y_2 + Y_3 - Y_4)\,\eta - Y_1 + Y_2 \\ (X_1 - X_2 + X_3 - X_4)\,\xi - X_1 + X_4 & (Y_1 - Y_2 + Y_3 - Y_4)\,\xi - Y_1 + Y_4 \end{bmatrix}$

Triangular Elements

For the triangular element type the standard element shown in the figure below

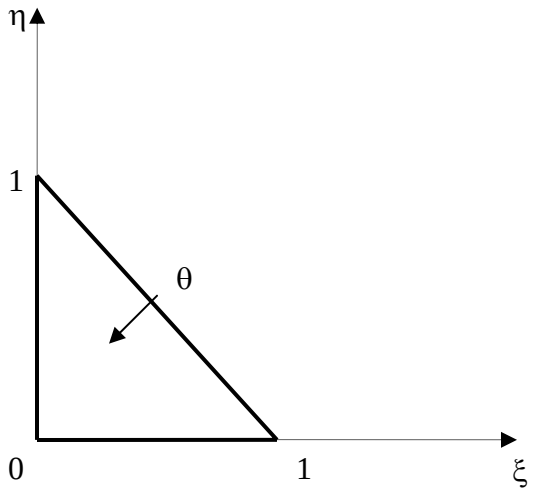


Figure 2: The Standard Triangular Element

For the triangular element the three basis functions are formulated as below
$\varphi_1 = l^0{}_1(\xi) l^0{}_1(\eta) l^0{}_1(\theta) = \theta = 1 - \xi - \eta$
$\varphi_2 = l^1{}_2(\xi) l^0{}_1(\eta) l^0{}_1(\theta) = \xi$
$\varphi_3(\xi,\eta) = l^0{}_1(\xi) l^1{}_2(\eta) l^0{}_1(\theta) = \eta$

Hence the B matrix and the Jacobian for mapping is as follows

B = $\begin{bmatrix} -1 & 1 & 0 \\ -1 & 0 & 1 \end{bmatrix}$

$$J = \begin{bmatrix} x_2 - x_1 & y_2 - y_1 \\ x_3 - x_1 & y_3 - y_1 \end{bmatrix}$$

Gaussian Quadrature

For both element types, the implementation of the program requires the evaluation of integrals. The integration in the program is performed using the Gaussian quadrature method. For the quadrilateral element a 4 point Gaussian quadrature method is used while for the triangular element one-point method is used. To perform this quadrature we need to determine the coordinates of the quadrature points as well as the multiplying factor or Côtes number.

By using Lagrange's polynomials and minimizing the quadrature error with respect to the location of the quadrature points (which is obtained by finding the roots of Legendre's polynomials) the following result was obtained for the quadrilateral and triangular elements.

Quadrilateral Elements

Côtes Numbers:

w1 = w2 = 0.5

Quadrature Points:

$(\frac{1}{2} + \frac{1}{2}\sqrt{1/3}, \frac{1}{2} + \frac{1}{2}\sqrt{1/3})$
$(\frac{1}{2} + \frac{1}{2}\sqrt{1/3}, \frac{1}{2} - \frac{1}{2}\sqrt{1/3})$
$(\frac{1}{2} - \frac{1}{2}\sqrt{1/3}, \frac{1}{2} + \frac{1}{2}\sqrt{1/3})$
$(\frac{1}{2} - \frac{1}{2}\sqrt{1/3}, \frac{1}{2} - \frac{1}{2}\sqrt{1/3})$

Triangular Elements

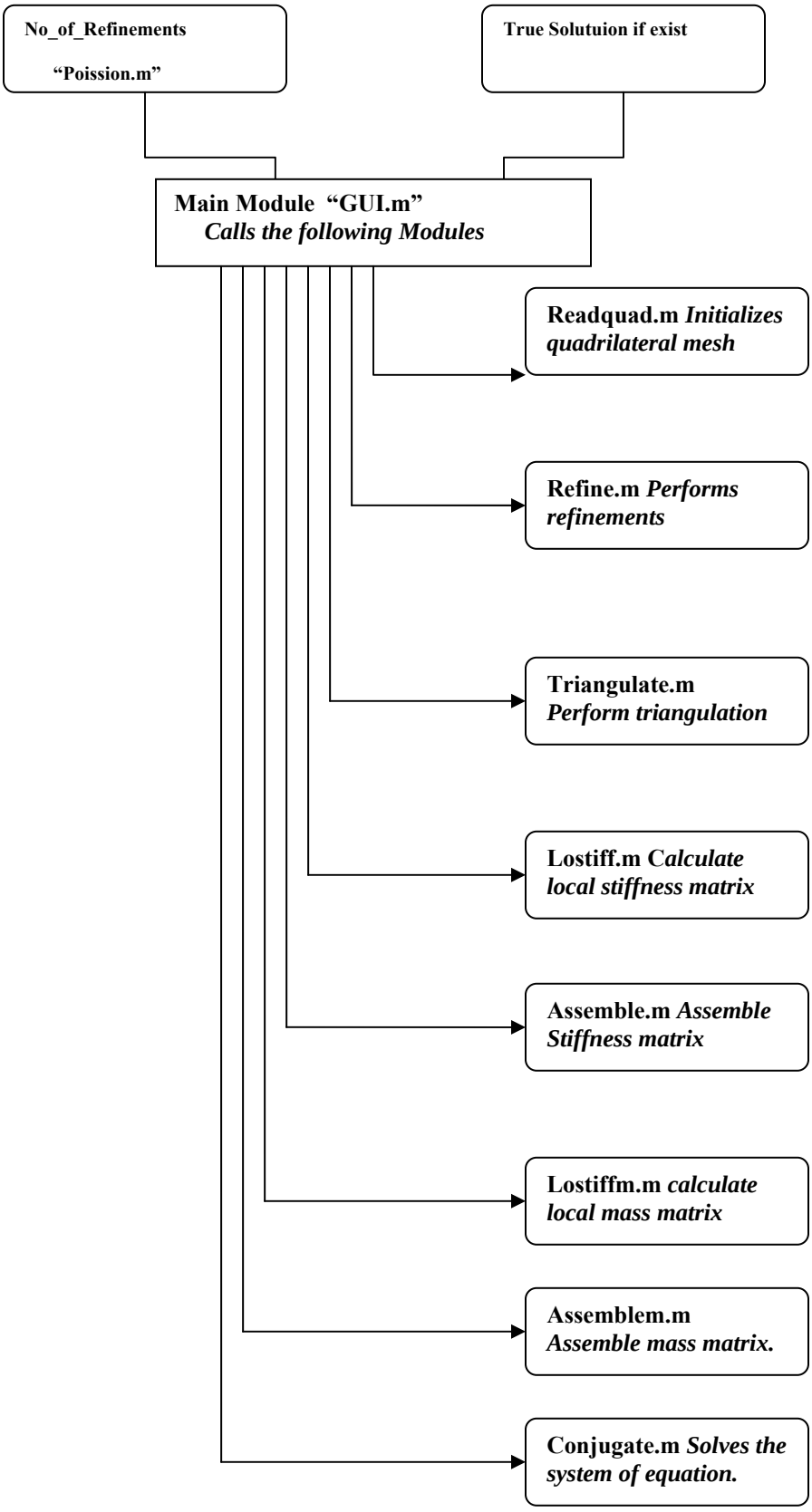
Côtes Number:

w = 1/6

Quadrature Points:

(0, 0.5)
(0.5, 0)
(0.5, 0.5)

PROGRAM CONSTRUCTION



Program Description

As the above flowchart describes how the program is implemented the main routine that call most of the subroutines is the GUI.m this files generates the graphical user interface and asks the user for the input. For more clear description the following figure of the GUI is presented below.

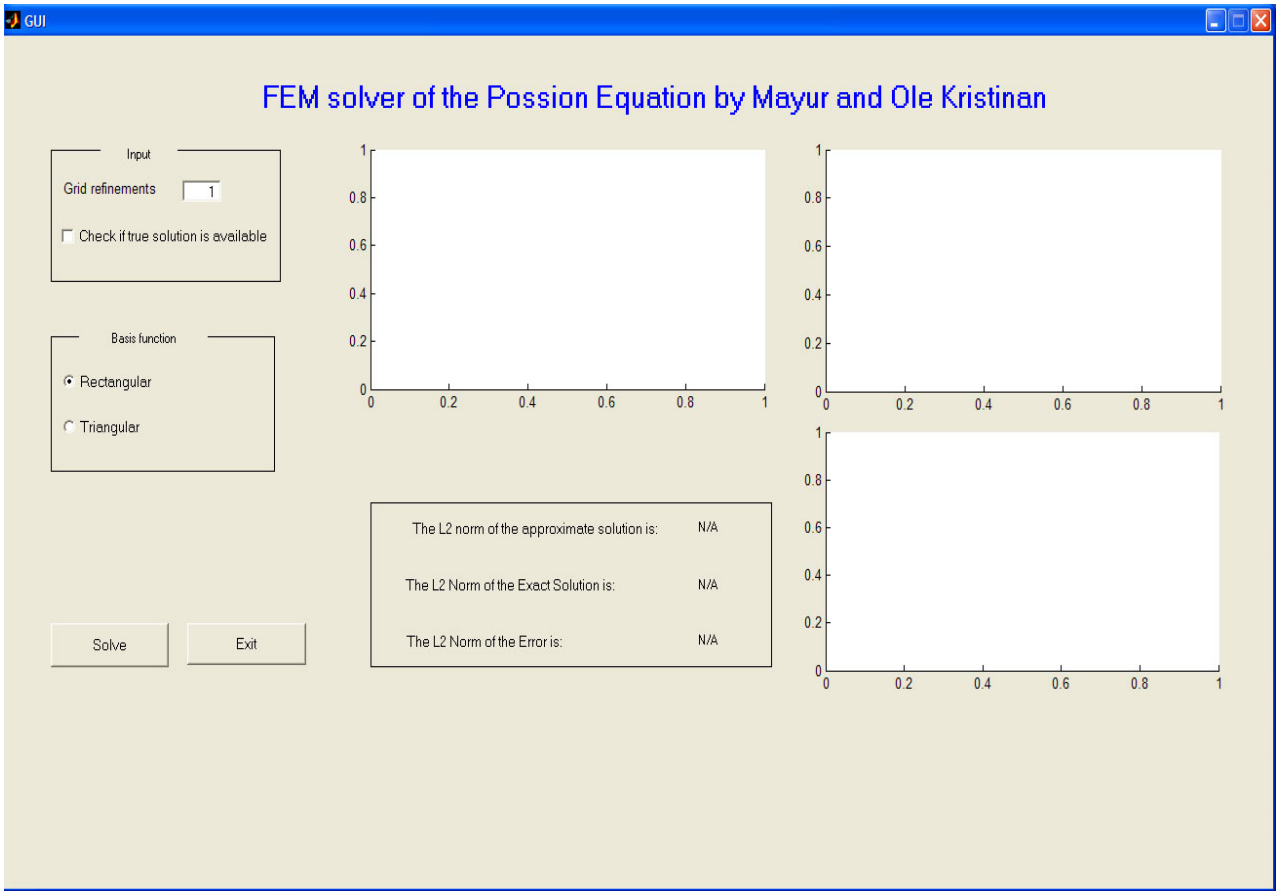


Figure 3. The GUI input for the FEM program

As one could easily see from the above figure that the user input is required in the terms of Number of refinements and also the Type of the mesh to be use i.e either triangular or rectangular. Then the corresponding FEM solution is shown in along with the true solution if the true solution exist and if the true solution does not exist the only the FEM solution is shown along with the description of the L2 norm of the exact and the approximate solution along with the error in the solution. The error tend to decrease as the number of refinements are increased.

Presentation of the Results

As a test, the program was run on an equation whose solution is known. In this experiment the following function was used.

$u = \sin 4\pi x + \exp(y)$ .....(5)

The diffusion coefficient k was selected to be x.

Then the corresponding full differential equation is

$\nabla\cdot(x\nabla u) = 4\pi \cos 4\pi x - (4\pi)^2 x \sin 4\pi x + x(\exp(y))$ .....(6)

where u is defined for $0 \le x \le 1$, $0 \le y \le 1$

and with a boundary condition of

$u = \sin 4\pi x + \exp(y)$ on all boundaries

The result obtained by running the program for this problem is shown in the figure

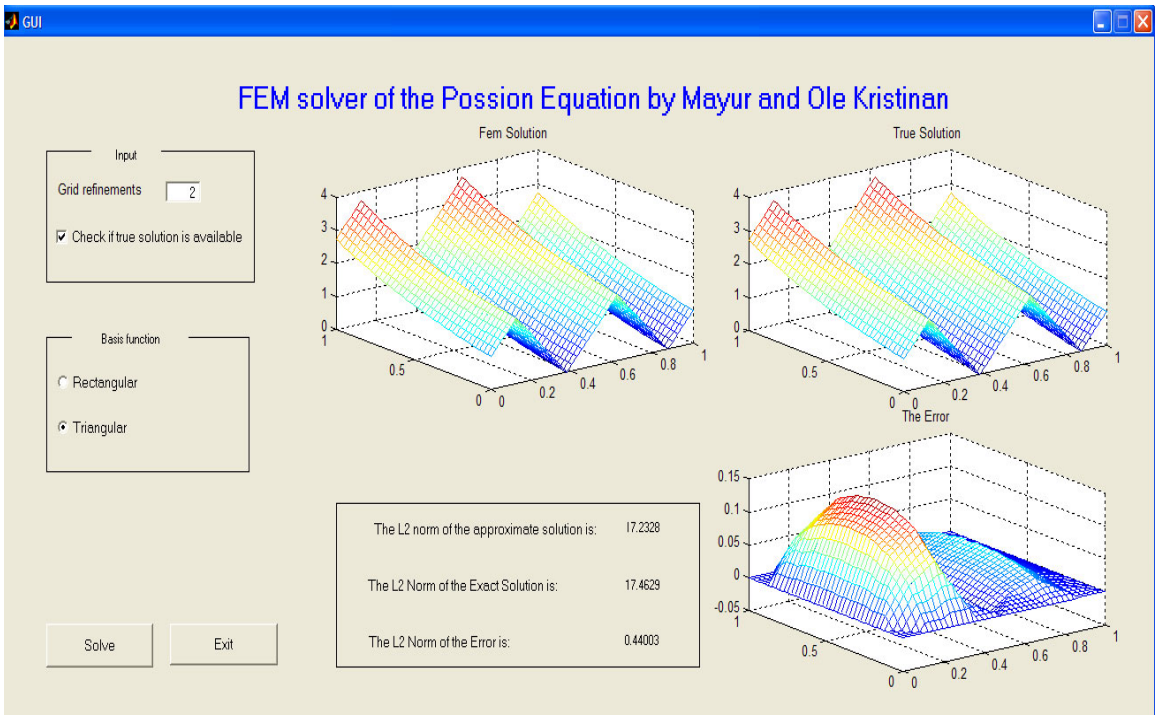


Figure 4

A more detailed presentation of the result will be given during the presentation which is to be given on 2nd feb 2004.

Convergence of Fem solution

As the pictures clearly show the program is giving a close approximation to the true solution of the differential equation. The following table shows L2 norm measurement of the solution and the error. To measure the L2 norm we have used two approaches. The first was to just apply MATLAB's norm function on the solution or error vector, and the second was to implement our own norm function. This norm is $(u^TMu)^{\frac{1}{2}}$, where u is the solution vector and M is its corresponding mass matrix. This norm is a better measurement as it is less affected by an increase in the number of components of the u vector.

L2 norm measurement results are displayed in Table 1. As can be seen from the progression the error is cut drastically as the mesh size is reduced. This is especially apparent with the use of latter measurement of L2 norm. Thus the FEM solution is obviously converging to the true solution.

Table1: L2 Norm Measurement Results

Number of Mesh Refinements	L2 norm for Triangular Element				L2 norm for the Quadrilateral Element			
	Using Mass Matrix		Using MATLAB's norm function		Using Mass Matrix		Using MATLAB's norm function	
	Solution	Error	Solution	Error	Solution	Error	Solution	Error
1	11.8246	0.2889	8.8344	1.5012	6.2096	0.0134	9.1273	0.4687
2	15.1138	0.0108	17.2328	0.4400	7.3134	0.0036	17.2859	0.3584
3	16.3032	5.00 e-004	32.7875	0.1827	7.9296	8.31e-004	32.6598	0.3223

Real Application of the program.

The program was tested for its applicability to real life problems. Our program was tested for heat distribution on a printed circuit board given by the teacher.

The Problem

Figure 5 shows a printed circuit board with various electronic components mounted. The electronic components generate heat. The board itself is mounted in a rack such that the ambient temperature is held constant at 20 degree Celsius. In the equilibrium, the devices have temperatures as follows:

- The large processor (e): 60 degrees Celsius.
- The two driving circuits (b, d): 50 degrees Celsius.
- The stabilizer (c): 40 degrees Celsius.

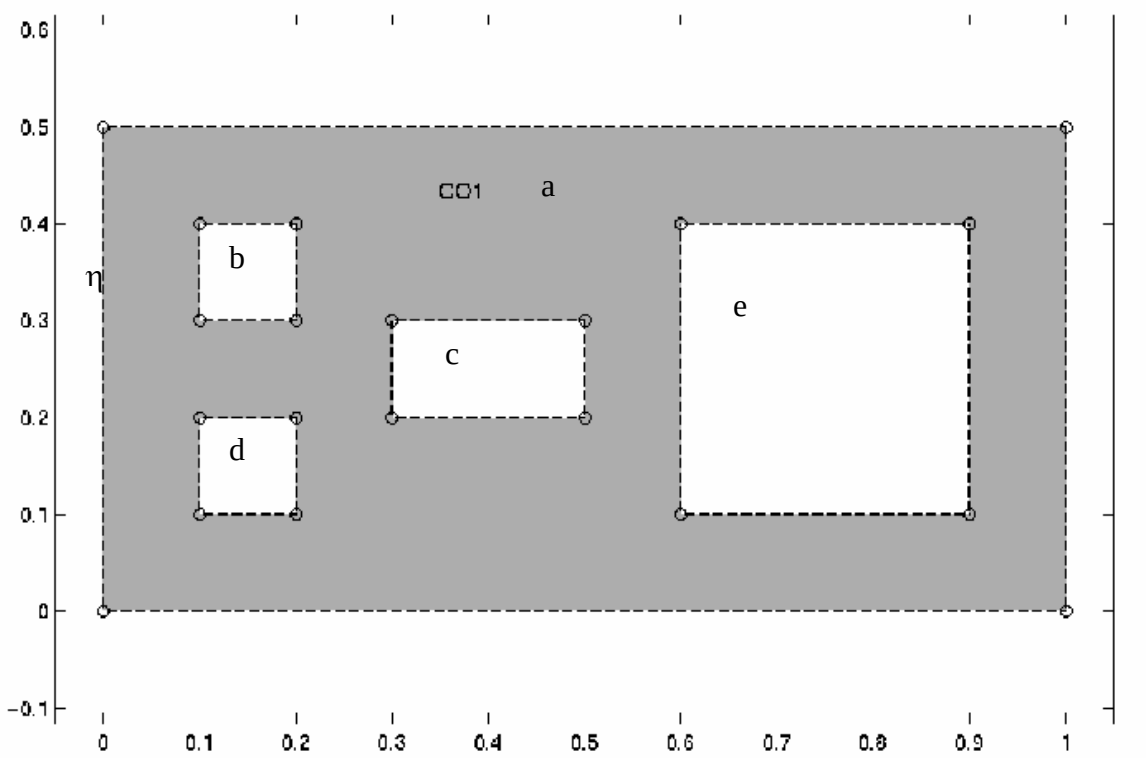


Figure 5: Printed Circuit Board Heat Distribution Problem

The problem is finding the temperature distribution on the board. The differential equation that describes this problem is

$$\Delta u = 0 \tag{8}$$

with the Dirichlet Boundary conditions mentioned above

Modifying the Program to Solve the Problem

The FEM program that we wrote can not be used as it is. A few modifications are necessary. One is scaling the dimensions of the domain of definition to match the problem. And the other is introducing boundary conditions in the interior of the domain. Otherwise, the main routines for the program remain unchanged.

A few lines of code handle the first problem. In our implementation we used a kind of mapping from the original domain to the newly defined domain by using a linear scaling of the dimensions in the mesh.

The second problem is handled by locating all the nodes that appear with in the internal regions whose temperature is fixed, i.e. the components, and fixing the temperatures to the respective values. This is very reminiscent of handling of normal boundary conditions.

Results

The modified program was applied to the problem and the following result was obtained. For comparison the same problem was solved using the FEMLAB package.

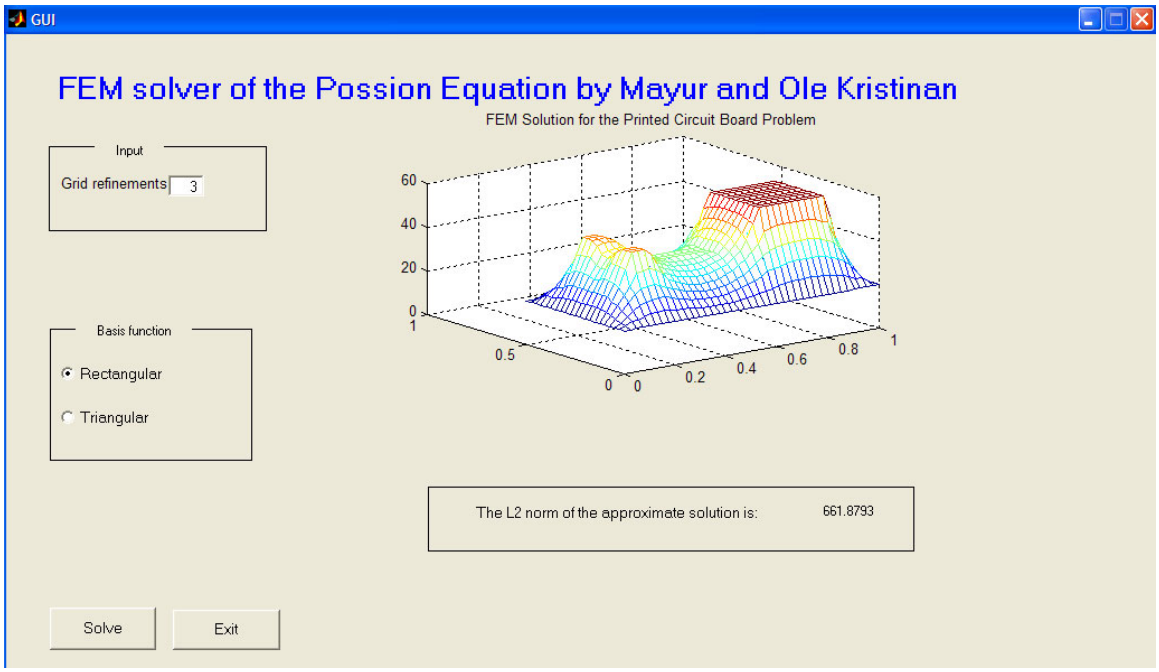


Figure 6: Solution to the Printed Circuit Board Heat Distribution Problem

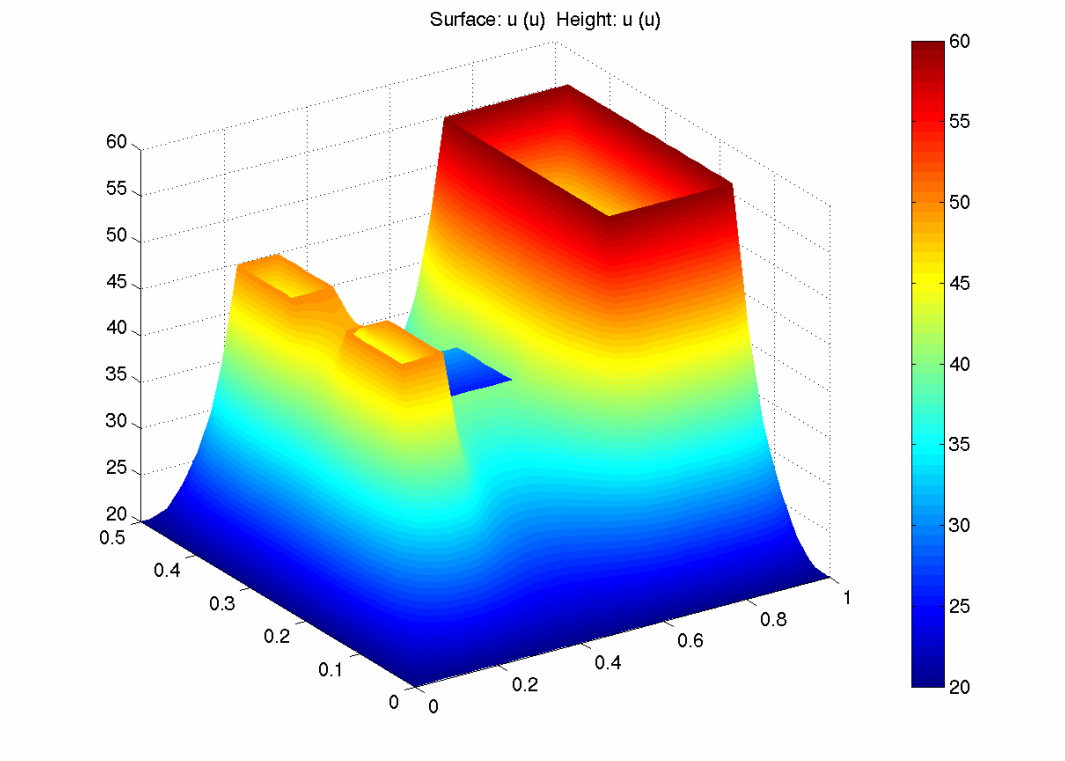


Figure 7: FEMLAB Solution to the Printed Circuit Board Heat Distribution Problem

References

1. Peter Hansbo, The Elements of Finite Elements; October 8, 1998. Lecture handout for the course Finite Elements, NADA, KTH, Stockholm
2. J. E. Akin, Application and Implementation of Finite Element Methods, Academic Press Inc. (London) Ltd. 1982, London
3. O. C. Zienkiewicz, Robert L. Taylor, Finite Element Method: Volume 1, The Basis; Butterworth-Heinemann. 2000