İTÜ

Computational Fluid Dynamics

UUM 510E

Project 2

By

Muhammet Berati Kılıç

511251184

Course given by:

Mehmet Şahin

Aeronautics and Astronautics Engineering / Graduate School

This document was prepared using a modified LaTeX template originally developed by Javad Ibrahimli.

CONTENTS

1 Problem Statement **2**

2 Methodology **3**

2.1 Galerkin Finite Element Formulation 3

2.2 Q1 Element Formulation and Coordinate Transformation 4

2.3 Gauss Quadrature and Local Stiffness Matrix 6

2.4 Mesh Generation and Element Connectivity 8

2.5 Global Assembly 10

2.6 Boundary Conditions and Solution of the Linear System 11

2.7 Error and Convergence Definition 13

3 Results and Discussion **14**

3.1 Numerical Error Results 14

3.2 Comparison With the Finite Difference Method 14

3.3 Convergence 16

3.4 Visualization 17

Conclusion **18**

References **19**

1. PROBLEM STATEMENT

This project considers the two dimensional Laplace equation around a 2D cylinder with a domain $[0, 2\pi] \times [1, 2]$.The governing equation in cartesian coordinates given by:

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0 \tag{1.1}$$

The analytical solution is:

$$u(r, \theta) = U_\infty (r - R/r) \sin(\theta) \tag{1.2}$$

The boundary conditions are the Dirichlet boundary conditions at:

$$u(1, \theta) = 0 \quad \text{and} \quad u(2, \theta) = \left(2 - \frac{1}{2}\right) \sin(\theta)$$

so

$$U_\infty = 1 \text{ and } R = 1$$

The problem is solved using the Galerkin finite element method (FEM) with quadrilateral Q1 elements. Uniform meshes of 81×41, 161×81, and 321×161 are used. The numerical solution is compared with the analytical solution, and the error is also compared with the finite difference method (FDM) results obtained in Project 1.

2. METHODOLOGY

2.1. Galerkin Finite Element Formulation

The governing equation, the Laplace equation, can be given in strong form as:

$$\nabla^2 u = 0$$

The Galerkin finite element formulation starts by multiplying the governing equation by a test function N_i and integrating it over the domain. This gives the error moment equation as:

$$\iint \nabla^2 u \, N_i \, dxdy = 0 \tag{2.1}$$

Using vector relation:

$$\nabla \cdot (\nabla u \, N_i) = \nabla^2 u \, N_i + \nabla u \cdot \nabla N_i$$

the first term in equation 2.1 can be rewritten as:

$$\nabla^2 u \, N_i = \nabla \cdot (\nabla u \, N_i) - \nabla u \cdot \nabla N_i \tag{2.2}$$

Substituting equation 2.2 into equation 2.1 gives:

$$\iint \nabla \cdot (\nabla u \, N_i) \; dxdy - \iint \nabla u \cdot \nabla N_i \, dxdy = 0$$

Using Green's theorem for first integral gives the weak form:

$$\oint n \cdot \nabla u \, N_i \, dl - \iint \nabla u \cdot \nabla N_i \, dxdy = 0 \tag{2.3}$$

Equation 2.3 can be written as:

$$\oint \frac{\partial u}{\partial n} N_i \, dl - \iint \nabla u \cdot \nabla N_i \, dxdy = 0 \tag{2.4}$$

This equation represents the weak form of the governing equation and provides the mathematical basis for the finite element discretization. In the Galerkin finite element method, the test functions, N_i, are selected from the same space as the shape functions used to approximate the solution. For the following Q1 formulation, this Galerkin property is reflected in equation 2.9.

2.2. Q1 Element Formulation and Coordinate Transformation

The Q1 quadrilateral element representation used in the present formulation is shown in figure 2.1. The figure illustrates the relation between the element selected from the two dimensional cylindrical domain and the corresponding reference element.

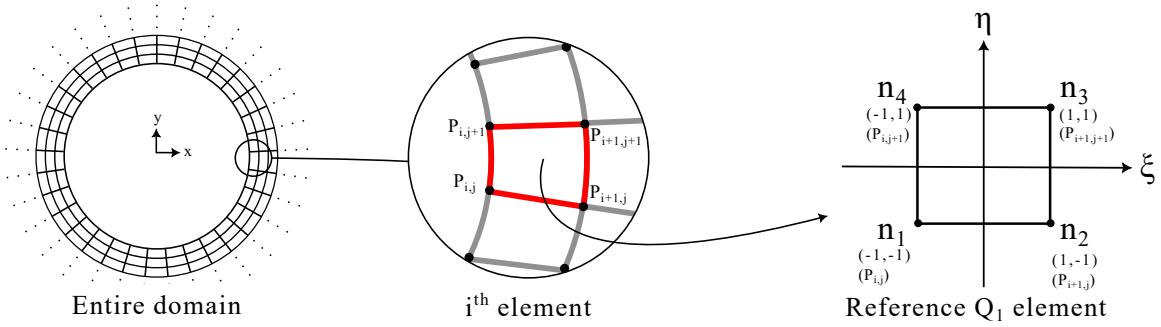


Figure 2.1: Representation of the Q1 quadrilateral element used in the finite element formulation. (see section 2.4)

The four node Q1 element is defined on the reference square, where the reference coordinates satisfy $-1 \leq \xi \leq 1$ and $-1 \leq \eta \leq 1$. The local node order on this reference element is taken as node 1 at $(-1, -1)$, node 2 at $(1, -1)$, node 3 at $(1, 1)$, and node 4 at $(-1, 1)$.

The corresponding Q1 shape functions are:

$$N_1(\xi, \eta) = \frac{(1-\xi)(1-\eta)}{4} \qquad N_2(\xi, \eta) = \frac{(1+\xi)(1-\eta)}{4}$$
$$N_3(\xi, \eta) = \frac{(1+\xi)(1+\eta)}{4} \qquad N_4(\xi, \eta) = \frac{(1-\xi)(1+\eta)}{4}$$

Their derivatives with respect to the reference coordinates are:

$$\frac{\partial N_1}{\partial \xi} = -\frac{1-\eta}{4} \qquad \frac{\partial N_1}{\partial \eta} = -\frac{1-\xi}{4}$$
$$\frac{\partial N_2}{\partial \xi} = \frac{1-\eta}{4} \qquad \frac{\partial N_2}{\partial \eta} = -\frac{1+\xi}{4}$$
$$\frac{\partial N_3}{\partial \xi} = \frac{1+\eta}{4} \qquad \frac{\partial N_3}{\partial \eta} = \frac{1+\xi}{4}$$
$$\frac{\partial N_4}{\partial \xi} = -\frac{1+\eta}{4} \qquad \frac{\partial N_4}{\partial \eta} = \frac{1-\xi}{4}$$

These shape functions have two roles in the formulation. First, they are used to approximate the unknown solution inside the element. Second, the same functions are used to describe the coordinate transformation from the reference element to the actual element in the x, y plane. In the present problem, the mesh points are first defined in polar coordinates and then transformed to the cartesian plane by $x = r\cos\theta$ and $y = r\sin\theta$. Therefore, each element has four cartesian nodal coordinates, (x_1, y_1), (x_2, y_2), (x_3, y_3), and (x_4, y_4). Using the Q1 shape functions, the cartesian coordinates inside an element are interpolated as:

$$x(\xi,\eta) = \sum_{j=1}^{4} N_j(\xi,\eta)x_j \qquad y(\xi,\eta) = \sum_{j=1}^{4} N_j(\xi,\eta)y_j \tag{2.5}$$

where x_j and y_j are the cartesian coordinates of the jth local node.

The coordinate derivatives required for the transformation are obtained by differentiating equation 2.5 with respect to the reference coordinates:

$$x_\xi = \sum_{j=1}^{4} x_j \frac{\partial N_j}{\partial \xi} \qquad x_\eta = \sum_{j=1}^{4} x_j \frac{\partial N_j}{\partial \eta} \qquad y_\xi = \sum_{j=1}^{4} y_j \frac{\partial N_j}{\partial \xi} \qquad y_\eta = \sum_{j=1}^{4} y_j \frac{\partial N_j}{\partial \eta}$$

The Jacobian determinant of the transformation is:

$$J = x_\xi y_\eta - x_\eta y_\xi$$

This determinant is used to transform the integration differential between the two coordinate systems:

$$dx\,dy = J\,d\xi d\eta \tag{2.6}$$

The Jacobian determinant also appears in the transformation of derivatives. Since the shape functions are defined in terms of ξ and η, but the stiffness matrix requires derivatives with respect to x and y, the inverse transformation terms are needed. These terms are obtained as:

$$\xi_x = \frac{y_\eta}{J} \qquad \xi_y = -\frac{x_\eta}{J} \qquad \eta_x = -\frac{y_\xi}{J} \qquad \eta_y = \frac{x_\xi}{J}$$

Using the chain rule, the derivatives of the shape functions with respect to the cartesian coordinates are:

$$N_{i,x} = N_{i,\xi}\xi_x + N_{i,\eta}\eta_x \qquad N_{i,y} = N_{i,\xi}\xi_y + N_{i,\eta}\eta_y$$

Thus, the Q1 shape functions are first defined on the reference element, and their derivatives are then transformed to the cartesian coordinate system before the element stiffness matrix is evaluated.

2.3. Gauss Quadrature and Local Stiffness Matrix

The element integrals in the finite element formulation are evaluated numerically by Gauss quadrature. This method approximates an integral by evaluating the function at selected Gauss points and multiplying these values by the corresponding Gauss weights.

For a one dimensional integral over the interval $[-1, 1]$, the Gauss quadrature rule with M integration points is written as:

$$\int_{-1}^{1} f(\xi)\, d\xi \cong \sum_{m=1}^{M} f(\xi_m) w_m$$

where ξ_m is the mth Gauss point and w_m is the corresponding Gauss weight. Since the Q1 quadrilateral element is two dimensional, the integration is performed in both ξ and η directions. Therefore, the two dimensional Gauss quadrature rule is written as:

$$\int_{-1}^{1}\int_{-1}^{1} f(\xi,\eta)\, d\xi d\eta \cong \sum_{m=1}^{M}\sum_{n=1}^{N} f(\xi_m,\eta_n) w_m w_n$$

where ξ_m and η_n are the Gauss points in the ξ and η directions, respectively. The corresponding weights are denoted by w_m and w_n

For the Q1 element used in this project, a 2×2 Gauss quadrature rule is used. Therefore, $M = 2$ and $N = 2$. The Gauss points and weights are [1]:

$$\xi_{1,2} = \mp\frac{1}{\sqrt{3}} \qquad \eta_{1,2} = \mp\frac{1}{\sqrt{3}} \qquad w_1 = w_2 = 1$$

Thus, the element integral is evaluated at four Gauss points inside the reference element.

The local stiffness matrix is obtained from the domain term of the weak form given by:

$$\iint \nabla u \cdot \nabla N_i \, dxdy \tag{2.7}$$

Using the Q1 shape functions u can be written as:

$$u = \sum_{j=1}^{4} N_j u_j$$

The gradient of the unknown field becomes:

$$\nabla u = \sum_{j=1}^{4} u_j \nabla N_j \tag{2.8}$$

Substituting equation 2.8 into equation 2.7 gives:

$$\iint \nabla u \cdot \nabla N_i \, dxdy = \sum_{j=1}^{4} u_j \iint \nabla N_j \cdot \nabla N_i \, dxdy \tag{2.9}$$

Therefore, the coefficient multiplying the nodal value u_j can be defined as the local stiffness matrix entry:

$$k_{ij}^{(e)} = \iint \nabla N_j \cdot \nabla N_i \, dxdy$$

Expanding the dot product gives:

$$k_{ij}^{(e)} = \iint \left[\frac{\partial N_j}{\partial x}\frac{\partial N_i}{\partial x} + \frac{\partial N_j}{\partial y}\frac{\partial N_i}{\partial y} \right] dxdy$$

Using the coordinate transformation to the reference element given by equation 2.6 the local stiffness matrix entry is written as:

$$k_{ij}^{(e)} = \int_{-1}^{1}\int_{-1}^{1}\left[\frac{\partial N_j}{\partial x}\frac{\partial N_i}{\partial x} + \frac{\partial N_j}{\partial y}\frac{\partial N_i}{\partial y}\right] J\, d\xi d\eta \tag{2.10}$$

Applying the 2×2 Gauss quadrature rule to equation 2.10 gives:

$$k_{ij}^{(e)} \cong \sum_{m=1}^{2}\sum_{n=1}^{2}\left[N_{j,x}N_{i,x} + N_{j,y}N_{i,y}\right] J w_m w_n$$

This expression shows that the derivatives of the shape functions and the Jacobian determinant are evaluated at each Gauss point. The resulting values are multiplied by the Gauss weights and summed to obtain the local stiffness matrix entry $k_{ij}^{(e)}$.

Since the Q1 element has four local nodes, the indices i and j take the values 1 to 4. Therefore, each element produces a 4×4 local stiffness matrix as:

$$k^{(e)} = \begin{bmatrix} k_{11}^{(e)} & k_{12}^{(e)} & k_{13}^{(e)} & k_{14}^{(e)} \\ k_{21}^{(e)} & k_{22}^{(e)} & k_{23}^{(e)} & k_{24}^{(e)} \\ k_{31}^{(e)} & k_{32}^{(e)} & k_{33}^{(e)} & k_{34}^{(e)} \\ k_{41}^{(e)} & k_{42}^{(e)} & k_{43}^{(e)} & k_{44}^{(e)} \end{bmatrix}$$

2.4. Mesh Generation and Element Connectivity

The computational mesh is generated uniformly in the radial and angular directions within the domain intervals. The radial and angular mesh spacings are:

$$\Delta r = \frac{1}{N_r - 1} \qquad \Delta\theta = \frac{2\pi}{N_\theta}$$

where N_r is the number of radial nodes for each angle and N_θ is the number of angular nodes for each radial layer.

The point $\theta = 2\pi$ is not stored as an additional node because it coincides with $\theta = 0$. Therefore, the angular direction is closed through the element connectivity.

The total number of nodes is $N_r N_\theta$, and total number of elements is $(N_r - 1)N_\theta$. A global node number is assigned to each mesh point by:

$$P(i, j) = (i - 1)N_\theta + j \tag{2.11}$$

where i is the radial index and j is the angular index of node. Here, $P(i, j)$ denotes the global node number.

For one Q1 quadrilateral element, the four local nodes are selected as:

$$\begin{aligned} n_1 &= P(i, j) \\ n_2 &= P(i + 1, j) \\ n_3 &= P(i + 1, j + 1) \\ n_4 &= P(i, j + 1) \end{aligned}$$

For the last angular elements (N_θth element of each radial layer), $j + 1$ values equal to 1. This closes the mesh around the cylinder and connects the last angular line back to the first angular line.

The vector gives the global node numbers corresponding to the four local nodes of element e called element connectivity vector and written as:

$$C^{(e)} = [n_1, n_2, n_3, n_4]$$

2.5. Global Assembly

After the local stiffness matrix $k^{(e)}$ is computed for an element, it is inserted into the global stiffness matrix.

The placement of the local stiffness matrix entries into the global stiffness matrix is performed by using the local node permutation of the element. This local node permutation gives the relation between the local node numbers of an element and their corresponding global node numbers. Figure 2.2 shows this permutation on a schematic four element domain. In the selected element, the local node numbers are matched with the corresponding global node numbers, and this relation determines how the entries of the local stiffness matrix are transferred to the appropriate rows and columns of the global stiffness matrix A.

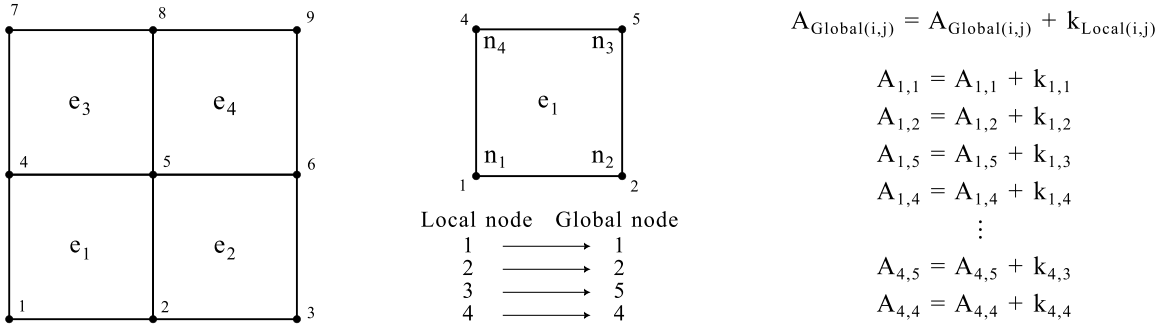


Figure 2.2: Schematic representation of the local node permutation for a selected element in a four element domain.

Each entry $k_{ij}^{(e)}$ contributes to the global matrix according to the global node numbers stored in the element connectivity vector as:

$$C^{(e)} = \left[C^{(e)}(1), C^{(e)}(2), C^{(e)}(3), C^{(e)}(4)\right]$$

Here, $C^{(e)}(i)$ is the global node number corresponding to the local node i of element e. Therefore, the local row and column indices of the element stiffness matrix are converted into global row and column indices by using the local node permutation stored in the connectivity vector. If this placement procedure is written as a rule, the global assembly rule [2] can be written as:

$$A_{C^{(e)}(i),C^{(e)}(j)} = A_{C^{(e)}(i),C^{(e)}(j)} + k_{ij}^{(e)} \qquad i = 1,2,3,4 \quad \text{and} \quad j = 1,2,3,4 \tag{2.12}$$

This operation is repeated for all elements in the mesh. If two neighboring elements share a node or an edge, their local stiffness matrices contribute to some of the same global matrix entries. These contributions are not overwritten. Instead, they are added to the same global matrix location according to equation 2.12.

Figure 2.3 continues the same four element schematic example given in figure 2.2 and illustrates how the local stiffness matrix entries of all elements are placed into the corresponding locations of the global stiffness matrix. The colors are used only for visualization, in order to indicate the element contributions to the global matrix.

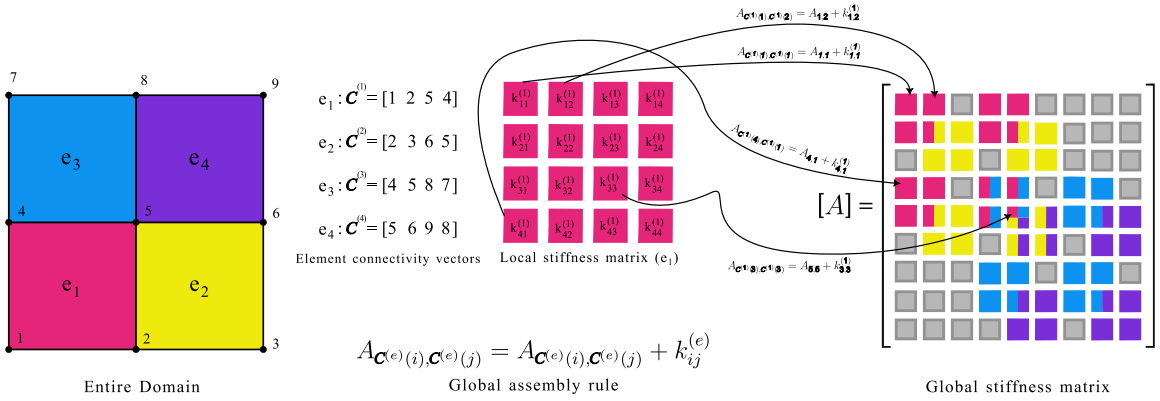


Figure 2.3: Schematic representation of the global assembly process for a four element domain.

Before the boundary conditions are implemented, A represents the assembled global stiffness matrix of the finite element system. Since each Q1 element only connects four local nodes, the resulting global matrix contains many zero entries. Therefore, the global matrix is stored in sparse form. In MATLAB sparse matrix notation, this structure can be described by three arrays which store the row indices, column indices, and numerical values of the nonzero entries, respectively [3]. After all element contributions are written in this form, the global stiffness matrix is formed without storing the zero entries.

2.6. Boundary Conditions and Solution of the Linear System

After the global assembly is completed, the finite element equations form the following linear algebraic system:

$$A\vec{x} = \vec{b}$$

where A is the assembled global stiffness matrix, $\overrightarrow{x}$ is the vector of nodal unknowns, and $\overrightarrow{b}$ is the right hand side vector. The boundary values are defined as Dirichlet boundary conditions. For the present problem, the inner and outer boundary values are:

$$u(1,\theta) = 0, \qquad u(2,\theta) = \left(2 - \frac{1}{2}\right)\sin\theta$$

Using the global node numbering given in equation 2.11, the nodes on the inner and outer boundary are represented respectively by:

$$P(1,j), \qquad P(N_r,j) \qquad j = 1,2,\ldots,N_\theta$$

Therefore, the Dirichlet values are implemented directly on the corresponding nodal unknowns. For the inner and outer boundaries, respectively:

$$x_{P(1,j)} = 0, \qquad x_{P(N_r,j)} = \left(2 - \frac{1}{2}\right)\sin\theta_j \qquad j = 1,2,\ldots,N_\theta$$

These equations replace the original algebraic equations associated with the boundary nodes. For each boundary node, the corresponding equation is converted into a defined value equation. This means that the coupling terms in that row are removed, the diagonal coefficient of the boundary unknown is set to one, and the right hand side is set to the defined boundary value. As a result, the boundary unknown is forced to satisfy the Dirichlet condition exactly.

After the boundary equations are replaced by the defined nodal value equations, the symbol A is used for the modified coefficient matrix of the final algebraic system. Similarly, $\overrightarrow{b}$ denotes the right hand side vector after the boundary values are inserted.

The final sparse linear system is solved using MATLAB's direct solver [4] (command, `x=A\b`). The obtained vector $\overrightarrow{x}$ contains the numerical solution values at all global nodes. Finally, these nodal values are arranged according to the global node numbering in equation 2.11 to form the numerical solution field on the 2D cylindrical domain.

2.7. Error and Convergence Definition

The numerical accuracy is investigated by comparing the computed solution with the analytical solution at the same grid points. The error is evaluated by the normalized L_2 error which is equivalent to the root mean square error and given by:

$$Error = \frac{\|u_{i,j} - u_{\text{analytic}}\|_2}{\sqrt{i_{\max} j_{\max}}} \tag{2.13}$$

Here, $u_{i,j}$ denotes the numerical solution, u_{analytic} denotes the analytical solution, and $i_{\max}$ and $j_{\max}$ represent the total numbers of grid points in the radial and angular directions, respectively.

The spatial convergence rate is determined by comparing the error values between meshes of increasing density. Since the discretization involves both radial and angular grid spacings, the convergence rate is evaluated in terms of Δr and $\Delta\theta$.

For the radial direction, convergence rate is given by:

$$p_{12_{\Delta r}} = \frac{\log\left(E_{\text{old}}/E_{\text{new}}\right)}{\log\left(\Delta r_{\text{old}}/\Delta r_{\text{new}}\right)} \tag{2.14}$$

For the angular direction, convergence rate is given by:

$$p_{12_{\Delta\theta}} = \frac{\log\left(E_{\text{old}}/E_{\text{new}}\right)}{\log\left(\Delta\theta_{\text{old}}/\Delta\theta_{\text{new}}\right)} \tag{2.15}$$

Where the p values represents convergence rate for consecutive meshes.

3. RESULTS AND DISCUSSION

3.1. Numerical Error Results

The Galerkin finite element solution was computed on three uniform meshes. The number of nodes, the number of Q1 quadrilateral elements, and the computed error values are listed in Table 3.1. The error is calculated using equation 2.13.

Table 3.1: Numerical error values obtained by the Q1 Galerkin finite element method.

Mesh	Nodes	Elements	Error
81×41	3321	3280	5.906×10^{-7}
161×81	13041	12960	1.481×10^{-7}
321×161	51681	51520	3.709×10^{-8}

As shown in Table 3.1, the error decreases consistently as the mesh is refined. This behavior indicates that the Q1 finite element approximation improves systematically with mesh refinement.

3.2. Comparison With the Finite Difference Method

The finite element error obtained in the present study is compared with the finite difference error obtained in Project 1. The comparison is given in Table 3.2 for the same mesh resolutions. For all three mesh levels, the Q1 FEM error is smaller than the corresponding FDM error.

Table 3.2: Comparison of FDM errors and Q1 FEM errors.

Mesh	FDM Error	FEM Error	FDM/FEM
81×41	4.027×10^{-5}	5.906×10^{-7}	68.180
161×81	1.036×10^{-5}	1.481×10^{-7}	69.980
321×161	2.629×10^{-6}	3.709×10^{-8}	70.890

As shown in Table 3.2, the ratio between the FDM and FEM errors remains close to the same range for all mesh levels. This indicates that, for the present problem and for the same mesh resolutions, the Q1 Galerkin finite element solution gives lower error values than the finite difference solution used in Project 1. Possible reasons for this difference include the fact that the Q1 Galerkin FEM formulation has a larger interaction region than the five point stencil used in the FDM solution, and the two methods use different discretisations. Through the global assembly process, an interior FEM node can receive contributions from the surrounding quadrilateral elements, which may contribute to the lower FEM error values observed in this problem.

Figures 3.1 and 3.2 show the variation of the error with respect to Δr and $\Delta\theta$ in logarithmic scale. In both figures, the FEM error curve is plotted together with the FDM error curve. The FEM curve remains below the FDM curve over the tested meshes.

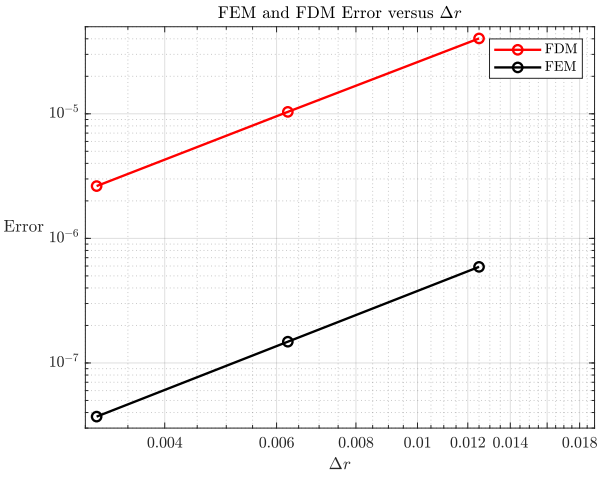


Figure 3.1: Comparison of FDM and Q1 FEM errors with respect to radial mesh spacing.

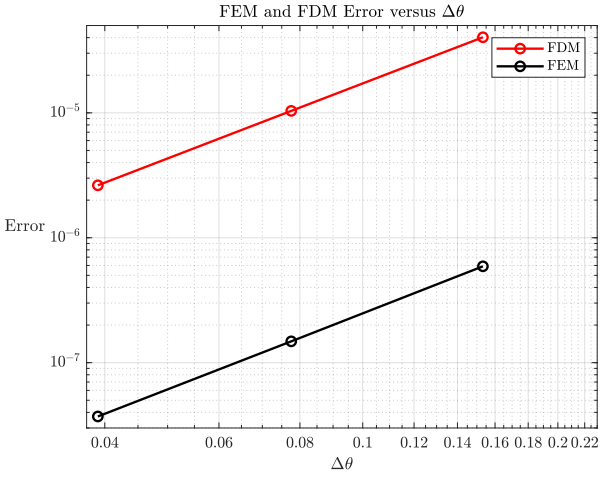


Figure 3.2: Comparison of FDM and Q1 FEM errors with respect to angular mesh spacing.

3.3. Convergence

The observed convergence rates were calculated using equations 2.14 and 2.15. The obtained rates are listed in Table 3.3.

Table 3.3: Observed spatial convergence rates.

Spacing	Mesh 1 - Mesh 2	Mesh 2 - Mesh 3
Δr	1.995	1.998
$\Delta\theta$	2.031	2.016

As shown in Table 3.3, the observed convergence rates are close to two in both coordinate directions. This indicates that the implemented Q1 Galerkin finite element formulation yields an approximately second order convergence trend in this study. The small deviations from exactly second order behavior can be attributed to the use of finite mesh levels.

Figure 3.3 illustrates the convergence behavior in logarithmic form. In a logarithmic convergence plot, the slope of the error curve corresponds to the observed convergence rate. Since the plotted data follow an almost straight line with slopes close to two, the figure is consistent with the approximately second order convergence behavior.

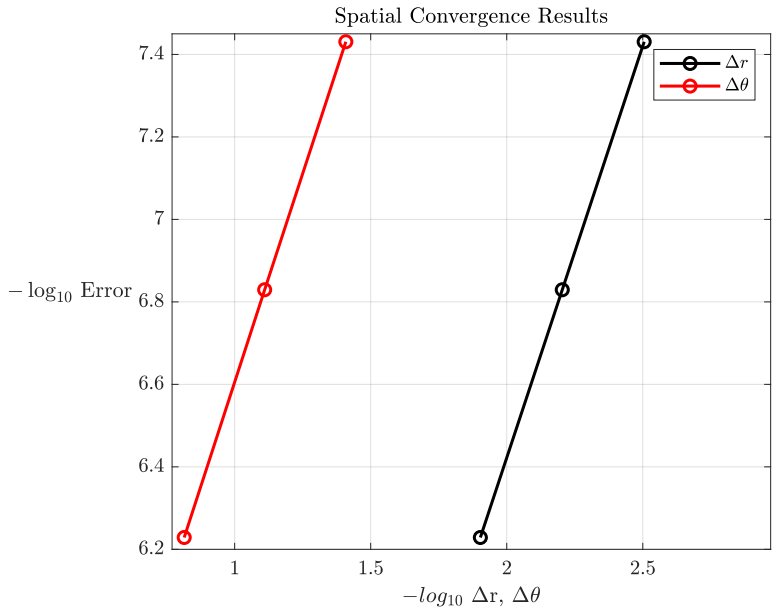


Figure 3.3: Spatial convergence results for the Q1 Galerkin finite element solution.

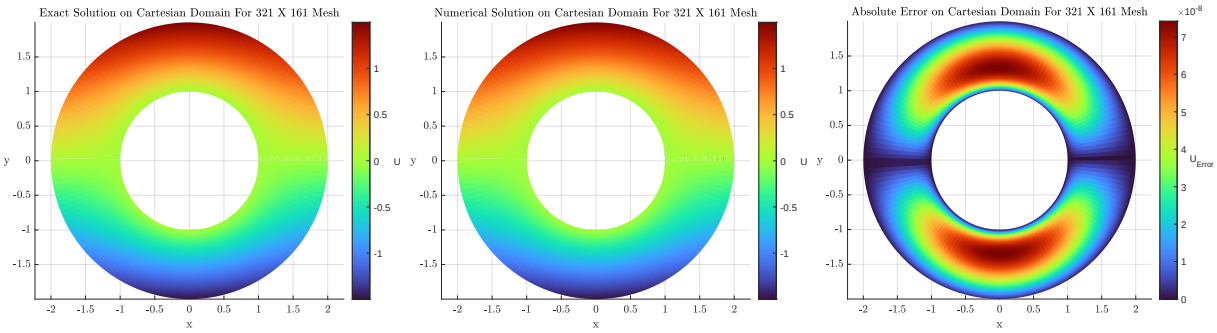
3.4. Visualization

The analytical solution, numerical FEM solution, and absolute error distribution of 321×161 mesh are displayed in the cartesian plane in figure 3.4. The mesh is generated by using the polar coordinates, but the results can be represented on the physical (cartesian) domain through:

$$x = r\cos\theta \qquad y = r\sin\theta$$

This representation places the computed solution directly on the 2D cylindrical geometry of the problem. Therefore, the spatial variation of the analytical solution, numerical FEM solution, and absolute error can be examined in the physical domain.

The absolute error plot is useful because it shows the spatial distribution of the difference between the numerical and analytical solutions. Even when the analytical and numerical solution fields appear visually very close, the error plot makes the remaining local differences more visible.



(a) Analytical solution. **(b)** Numerical FEM solution. **(c)** Absolute error.

Figure 3.4: Analytical solution, numerical FEM solution, and absolute error distribution represented on the cartesian domain for the fine mesh.

CONCLUSION

In this study, the two dimensional Laplace equation on an 2D cylindrical domain was solved using the Q1 Galerkin finite element method. The local stiffness matrices were computed using 2×2 Gauss quadrature and assembled into a sparse global system. Dirichlet boundary conditions were imposed directly on the algebraic equations.

The numerical results show that the error decreases systematically as the mesh is refined. The observed convergence rates are close to two in both radial and angular directions, indicating an approximately second order convergence behavior.

The comparison with the finite difference method shows that the Q1 Galerkin FEM solution gives lower error values for all tested mesh resolutions. The cartesian visualizations also show that the numerical FEM solution agrees well with the analytical solution on the cylindrical domain. Overall, the implemented method provides an accurate and consistent solution for the given Laplace problem.

BIBLIOGRAPHY

[1] E. W. Weisstein, Legendre-Gauss Quadrature, https://mathworld.wolfram.com/Legendre-GaussQuadrature.html, accessed: 14 May 2026.

[2] G. P. Nikishkov, Introduction to the Finite Element Method, https://www.iitg.ac.in/mech/documents/128/introfem.pdf, used pages: 25 and 39, accessed: 14 May 2026 (2004).

[3] The MathWorks, Inc., sparse: Create sparse matrix, https://www.mathworks.com/help/matlab/ref/sparse.html, accessed: 7 April 2026.

[4] The MathWorks, Inc., mldivide: Solve systems of linear equations Ax = B for x, https://www.mathworks.com/help/matlab/ref/double.mldivide.html, accessed: 7 April 2026.