Finite Element Method: Poisson Equation

Pratham Lalwani

Contents

Chapter 1 **Introduction** Page 2

Chapter 2 **Theory** Page 3

2.1 Approximating functions 3
Linear Interpolation — 3 • L^2-Projection — 3

2.2 The Finite Element Method 4
Weak Form of a PDE — 4 • Mass and Stiffness Matrices — 4

2.3 Error Analysis of FEM for Poisson Problem 4

Chapter 3 **Implementation** Page 5
Computation of Mass and Stiffness Matrices — 5 • Pseudocode of Finite Element for Poisson Equation — 5 • Pseudocode for Adaptive Finite Element — 5

Chapter 4 **Numerical Results** Page 7
Convergence of Interpolants — 7 • Example Problem: Poisson Equation on Different Domains — 7 • Convergence Plots — 7 • Comparison with Finite Difference — 8

Chapter 5 **Conclusion** Page 9

5.1 Pros and Cons of using Finite Element 9

Chapter 1

Introduction

Solving PDEs is one of the most ubiquitous and challenging problems in Mathematics. Some PDEs have a closed form analytic solution under some circumstances but most are unsolvable and thus are solved by numerical methods. There are various methods to solve PDEs but by far the most flexible one of them is the Finite Element Method.

The Finite Element method is to solve various forms of PDEs .. Include Examples . It is used widely in industry due to it's flexibility and reliability.

This report will outline the basics of finite element method and what are it's pros and cons.

Chapter 2

Theory

2.1 Approximating functions

Given data points $\{(x_i, y_i)\}_{i=1}^N$, an interpolant is a function f such that $f(x_i) = y_i$. The process of finding such an f is called interpolation. There are many ways to perform interpolation, including polynomial interpolation, Fourier Transform, and Spline interpolation.

One of the easiest interpolation techniques, which is fundamental to FEM, is Piecewise Linear Interpolation. This method divides the domain into n intervals (elements) and fits a linear function between the endpoints (nodes).

There are two primary ways to achieve a linear approximation in this context: Piecewise linear interpolation and the L^2-Projection method.

2.1.1 Linear Interpolation

Let $\Omega = [a, b]$ be partitioned into nodes $x_0, x_1, \ldots, x_N$. We define basis functions $\phi_i(x)$ (often called "hat functions") such that:

$$\phi_i(x_j) = \delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases} \tag{2.1}$$

The linear interpolant $\pi_h u$ of a continuous function u is given by:

$$\pi_h u(x) = \sum_{i=0}^{N} u(x_i)\phi_i(x) \tag{2.2}$$

This approximation is exact at the nodes but introduces error inside the intervals, which depends on the mesh size h.

2.1.2 L^2-Projection

While interpolation forces exact values at nodes, L^2-projection seeks the "best fit" approximation in an integral sense. We seek a function u_h in our finite element space V_h that minimizes the L^2 error:

$$\min_{v_h \in V_h} \|u - v_h\|_{L^2(\Omega)}^2 \tag{2.3}$$

Practically, this leads to a system of equations involving the mass matrix, ensuring the error is orthogonal to the approximation space.

2.2 The Finite Element Method

2.2.1 Weak Form of a PDE

The classical (strong) form of a PDE requires the solution to be twice differentiable. FEM relies on the **Weak Form**, which relaxes these requirements.

Consider the 1D Poisson problem:

$$-u''(x) = f(x) \quad \text{on } (0,1), \quad u(0) = u(1) = 0 \tag{2.4}$$

To derive the weak form, we multiply by a test function v and integrate by parts. The weak formulation is: Find $u \in V$ such that for all $v \in V$:

$$\int_0^1 u'(x)v'(x)\,dx = \int_0^1 f(x)v(x)\,dx \tag{2.5}$$

This bilinear form is usually denoted $a(u,v) = L(v)$.

2.2.2 Mass and Stiffness Matrices

To solve this numerically, we substitute $u \approx u_h = \sum_j \xi_j \phi_j$ and choose test functions $v = \phi_i$. This yields a linear system $A\xi = b$.

- **The Stiffness Matrix (A or K):** Represents the discrete Laplacian.

$$A_{ij} = a(\phi_j, \phi_i) = \int_\Omega \phi_j'(x)\phi_i'(x)\,dx \tag{2.6}$$

- **The Mass Matrix (M):** Represents the L^2 inner product.

$$M_{ij} = (\phi_j, \phi_i) = \int_\Omega \phi_j(x)\phi_i(x)\,dx \tag{2.7}$$

2.3 Error Analysis of FEM for Poisson Problem

A fundamental result in FEM theory is **Céa's Lemma**. For piecewise linear elements on a mesh with size h, the error estimates are:

1. **Energy Norm (H^1):** $\|u - u_h\|_{H^1} \leqslant Ch\|u\|_{H^2}$ (Linear convergence).
2. L^2 **Norm:** $\|u - u_h\|_{L^2} \leqslant Ch^2\|u\|_{H^2}$ (Quadratic convergence).

This implies that halving the mesh size reduces the L^2 error by a factor of 4.

Chapter 3

Implementation

3.0.1 Computation of Mass and Stiffness Matrices

In practice, we iterate over elements rather than nodes. For a uniform 1D mesh with element size h:

The local stiffness matrix for a single element is:

$$K_{local} = \frac{1}{h}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \tag{3.1}$$

The local mass matrix for a single element is:

$$M_{local} = \frac{h}{6}\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \tag{3.2}$$

These local matrices are then "assembled" into the global system matrices based on the global node indices.

3.0.2 Pseudocode of Finite Element for Poisson Equation

The general procedure for solving the Poisson equation in 1D or 2D is similar, differing mostly in mesh generation and basis function definitions.

Algorithm 1: FEM for Poisson Equation ($-\Delta u = f$)

```
  Input: Mesh $\mathcal{T}$, Load function $f$
  Output: Solution vector $\mathbf{u}$
1 Initialize Global Stiffness Matrix $K = 0$, Load Vector $b = 0$;
2 for each element $T \in \mathcal{T}$:
3     Compute local stiffness matrix $K_{loc}$;
4     Compute local load vector $b_{loc}$ using quadrature;
5     Assemble $K_{loc} \to K$ and $b_{loc} \to b$ based on global indices;
6 Apply Boundary Conditions:
7 for each boundary node $i$:
8     Modify row $i$ of $K$ and $b_i$ to enforce Dirichlet BCs;
9 Solve linear system $K\mathbf{u} = \mathbf{b}$;
```

3.0.3 Pseudocode for Adaptive Finite Element

Adaptive FEM (AFEM) improves accuracy by refining the mesh only where the error is high, rather than uniformly.

Algorithm 2: Adaptive Finite Element Method (SOLVE-ESTIMATE-MARK-REFINE)

```
Input: Initial coarse mesh $\mathcal{T}_0$, Tolerance $\epsilon$
Output: Refined Solution $u_k$
1 $k \leftarrow 0$;
2 while Estimated Error $> \epsilon$:
3     SOLVE: Compute discrete solution $u_k$ on $\mathcal{T}_k$;
4     ESTIMATE: Compute error indicator $\eta_T(u_k)$ for each element $T$;
5     MARK: Select set of elements $\mathcal{M} \subset \mathcal{T}_k$ with largest $\eta_T$ (e.g., Dörfler marking);
6     REFINE: Bisect elements in $\mathcal{M}$ to generate $\mathcal{T}_{k+1}$;
7     $k \leftarrow k + 1$;
```

Chapter 4

Numerical Results

4.0.1 Convergence of Interpolants

To verify the theory, we test the interpolation error of a known function $u(x) = \sin(\pi x)$. Plotting the error against mesh size h on a logarithmic scale reveals:

- Slope ≈ 2 for the L^2 norm.
- Slope ≈ 1 for the H^1 semi-norm.

4.0.2 Example Problem: Poisson Equation on Different Domains

We examine the behavior of FEM on the Poisson equation $-\Delta u = 1$ with Dirichlet boundary conditions on three domains:

Unit Square

On a unit square, the mesh is regular, and the solution is smooth. We observe optimal convergence rates ($O(h^2)$ in L^2 norm). The stiffness matrix is highly structured.

Unit Circle

For a circular domain, the boundary is approximated by a polygon. This geometric approximation introduces a "variational crime." As the mesh is refined, the polygonal boundary approaches the true circle, but the error near the boundary can dominate if not handled correctly.

L-Shaped Domain (Re-entrant Corner)

The L-shaped domain contains a re-entrant corner (a 270° interior angle). The exact solution typically has a singularity at this corner, where the gradient goes to infinity ($\nabla u \sim r^{-1/3}$). Because u is not in H^2, the standard theoretical convergence rates fail. Uniform refinement yields suboptimal convergence ($< O(h)$). This is a prime candidate for the **Adaptive FEM** described earlier, which heavily refines the mesh near the corner to recover optimal rates.

4.0.3 Convergence Plots

As shown in Table 4.1, the numerical order of convergence matches the theoretical prediction of 2 for smooth domains. However, for the L-shape domain, the order drops to approximately 0.66 without adaptivity.

Mesh size (h)	L^2 **Error**	**Order** (p)
0.1	1.25e-3	–
0.05	3.12e-4	2.00
0.025	7.80e-5	2.00

Table 4.1: Convergence rates for the Poisson problem on a square using linear elements.

4.0.4 Comparison with Finite Difference

- **Geometry:** Finite Difference (FD) relies on structured grids (squares/rectangles), making it difficult to implement on curved domains (like the circle). FEM handles unstructured triangular/tetrahedral meshes naturally.
- **Boundary Conditions:** Implementing Neumann (derivative) boundary conditions is more natural in FEM (via the weak form) than in FD.
- **Simplicity:** FD is generally easier to implement for simple box domains, whereas FEM requires complex data structures (mesh connectivity, assembly).

Chapter 5

Conclusion

5.1 Pros and Cons of using Finite Element

Pros:

- **Geometric Flexibility:** Can model complex, irregular shapes standard in engineering (cars, turbines, bones).
- **Mathematical Rigor:** Strong theoretical foundations for error analysis (L^2 and H^1 error bounds).
- **Material Heterogeneity:** Easily handles different materials in different elements.
- **Boundary Conditions:** Natural handling of Neumann and Robin boundary conditions.

  Cons:

- **Complexity:** Implementation is significantly more involved than Finite Difference, requiring mesh generation and matrix assembly.
- **Computational Cost:** Assembly of the global matrix can be expensive, and the resulting systems are often unstructured, requiring sophisticated solvers.