Building Blocks of a Finite Element Code

Submitted by,

Asif Istiak; ID: 20205083

Mechanical Design Engineering

Andong National University

Submitted to,

Professor See Jo Kim

Mechanical Design Engineering

Andong National University

Contents

Abstract

Building Block:

- ❖ Mathematical View Point
- ❖ Realize code for implementation
- ❖ Explain a general phenomenon
- ❖ Blocks of FEA

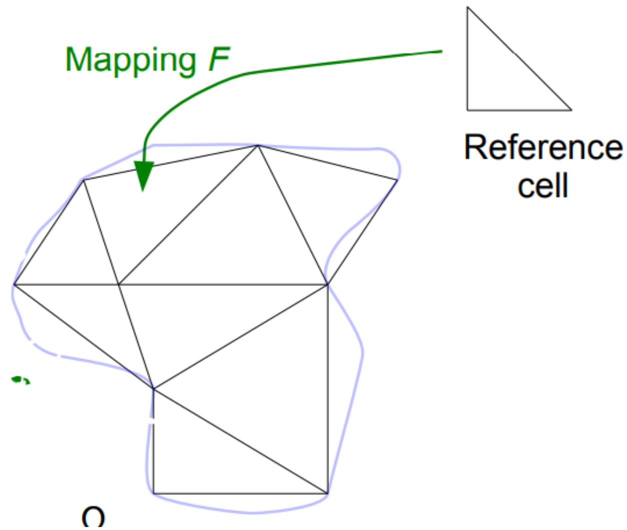
Shape function in a given domain equispaced nodes in the domain

DOI:
10.1007/s00366-018-0593-7

Introduction

Building Block

- How to consider Strong form equation to Weak form quation.
- Basic step for FEA
- Basic of Cell formation
- Code to build nodes for a particular point



Reference

https://www.math.colostate.edu/~bangerth/videos.676.4.html

BASIC CONCEPT

Mathematical View Point

$$\begin{aligned} -\Delta u &= f \quad \text{in } \Omega \\ u &= 0 \quad \text{on } \partial\Omega \end{aligned}$$

Strong form of Poisson equation

$$(\nabla \varphi, \nabla u) = (\varphi, f) \quad \forall \varphi$$

Weak form with a test function

function u(x) from an infinite dimensional function space

$$u_h(x) = \sum_{j=1}^{N} U_j \varphi_j(x)$$

finite dimensional function of the form

$$(\nabla \varphi_i, \nabla u_h) = (\varphi_i, f) \quad \forall i = 1 \ldots N$$

Linearly independent, this yields N equations for N coefficients

Galerkin method

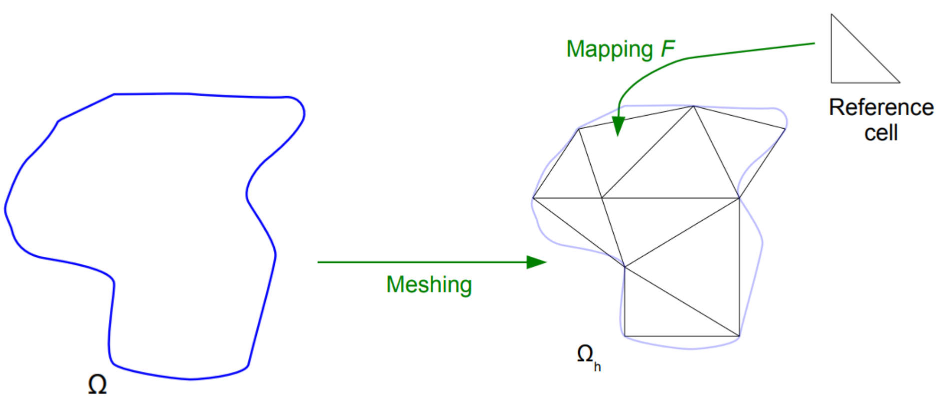
Steps

Number of Steps

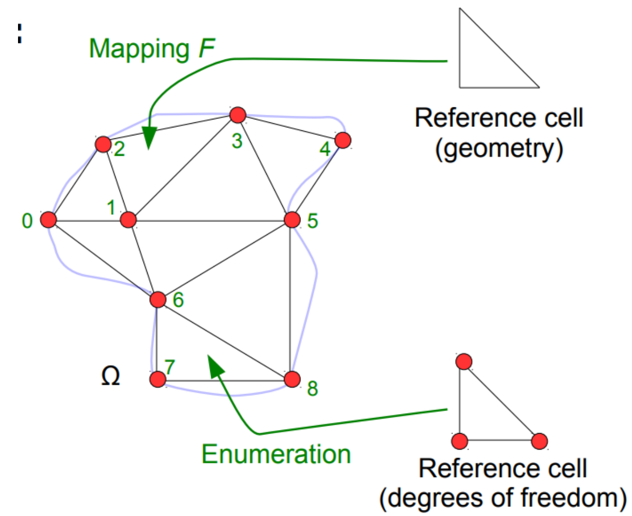
Subdivision of the domain → Mesh → shape function

Each cell=mapping(ref. cell)

DOF of global Mesh



Basis Function



Shape Function

Steps

Linear System

Given the definition $u_h = \sum_{j=1}^{N} U_j \varphi_j(x)$, we can expand the bilinear form

$$(\nabla \varphi_i, \nabla u_h) = (\varphi_i, f) \quad \forall i = 1 \ldots N$$

to obtain:

$$\sum_{j=1}^{N} (\nabla \varphi_i, \nabla \varphi_j) U_j = (\varphi_i, f) \quad \forall i = 1 \ldots N$$

This is a linear system

$$AU = F$$

with

$$A_{ij} = (\nabla \varphi_i, \nabla \varphi_j) \qquad F_i = (\varphi_i, f)$$

For large-scale computations, data structures and algorithms must be parallel

- Direct solvers
- Iterative solvers
- Parallel solvers

Quadrature

$$A_{ij} \approx \sum_K \sum_{q=1}^{Q} J_K^{-1}(\hat{x}_q) \hat{\nabla} \hat{\varphi}_i(\hat{x}_q) \cdot J_K^{-1}(\hat{x}_q) \hat{\nabla} \hat{\varphi}_j(\hat{x}_q) \underbrace{|\det J_K(\hat{x}_q)| \; w_q}_{=:\mathrm{JxW}}$$

Mapping

$$\begin{aligned} A_{ij} &= (\nabla \varphi_i, \nabla \varphi_j) \\ &= \sum_K \int_K \nabla \varphi_i(x) \cdot \nabla \varphi_j(x) \\ &= \sum_K \int_{\hat{K}} J_K^{-1}(\hat{x}) \hat{\nabla} \hat{\varphi}_i(\hat{x}) \cdot J_K^{-1}(\hat{x}) \hat{\nabla} \hat{\varphi}_j(\hat{x}) \; |\det J_K(\hat{x})| \end{aligned}$$

Steps

Linear System

Given the definition $u_h = \sum_{j=1}^{N} U_j \varphi_j(x)$, we can expand the bilinear form

$$(\nabla \varphi_i, \nabla u_h) = (\varphi_i, f) \quad \forall i = 1 \ldots N$$

to obtain:

$$\sum_{j=1}^{N} (\nabla \varphi_i, \nabla \varphi_j) U_j = (\varphi_i, f) \quad \forall i = 1 \ldots N$$

This is a linear system

$$AU = F$$

U,F= Stored as array(Sparse Matrix)

with

$$A_{ij} = (\nabla \varphi_i, \nabla \varphi_j) \qquad F_i = (\varphi_i, f)$$

A=store it in compressed row format(Sparse Matrix)

For large-scale computations, data structures and algorithms must be parallel

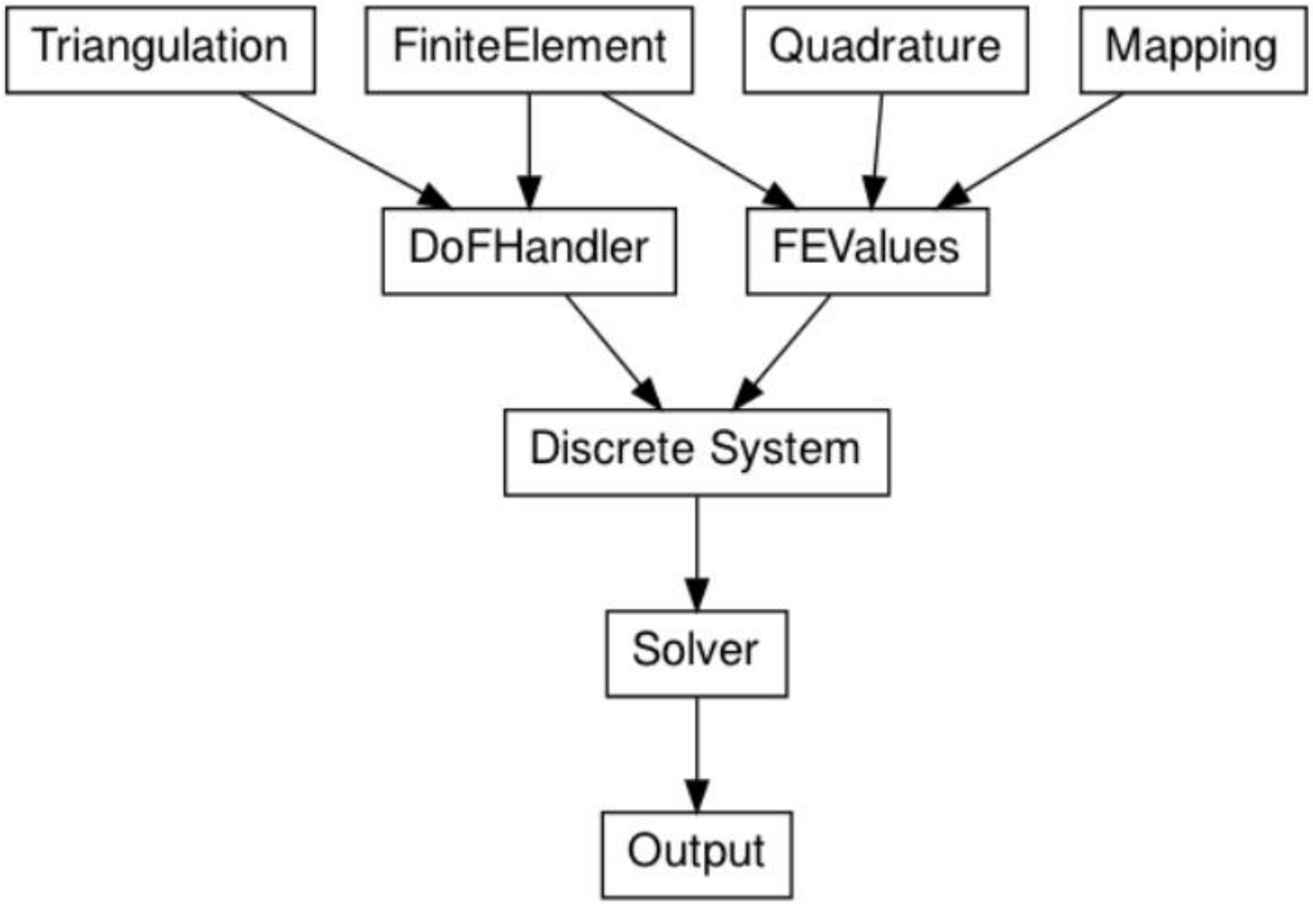
- Direct solvers
- Iterative solvers
- Parallel solvers

After Solving(Post Processing)

- Visualize
- Evaluate for quantities of interes
- Estimate the error

Flow Chart

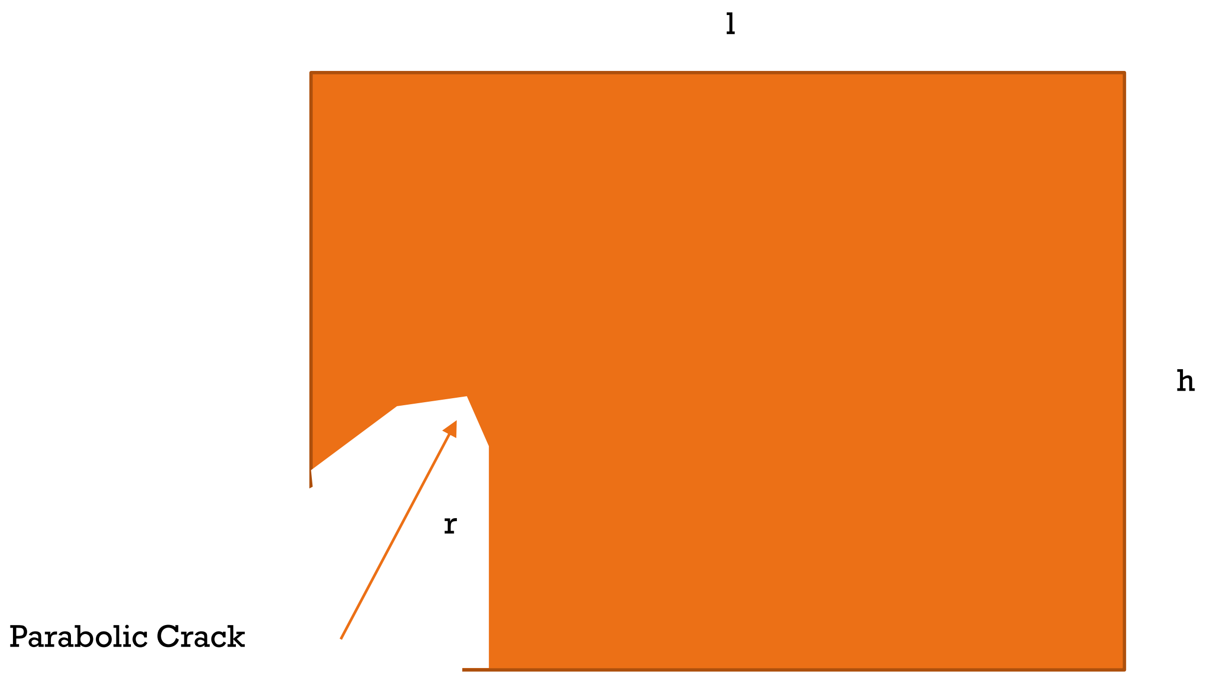
These sections will be explicitly or implicitly applied in the codes



Conclusions

- concepts that need to be represented by software components.
- Other components relate to solve PDE
- Code for building

VS Code Demonstration



Thank You