Block (3/3)

The Finite Element Method for Applications in Electrical Engineering

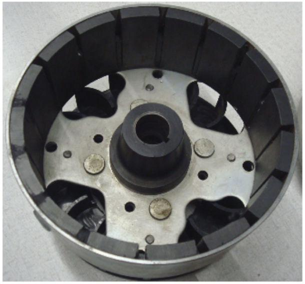
EE4375 - FEM For EE Applications

Domenico Lahaye

DIAM - Delft Institute of Applied Mathematics

Last updated February 18, 2024

Modeling of Permanent Magnet Machines



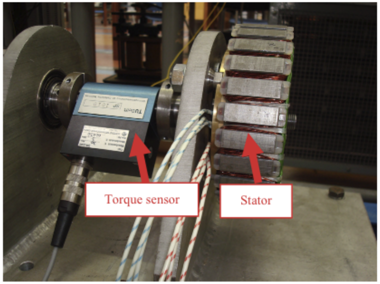
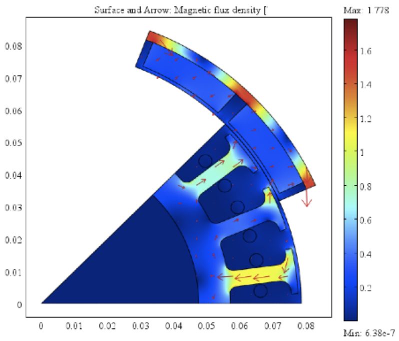


Fig. 2: Rotor of permanent magnet machine under study and test setup.

Modeling of Permanent Magnet Machines



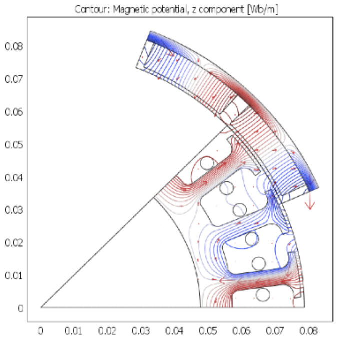


Fig. 6: Flux density and flux contour of the PM machine during load.

Modeling of Permanent Magnet Machines

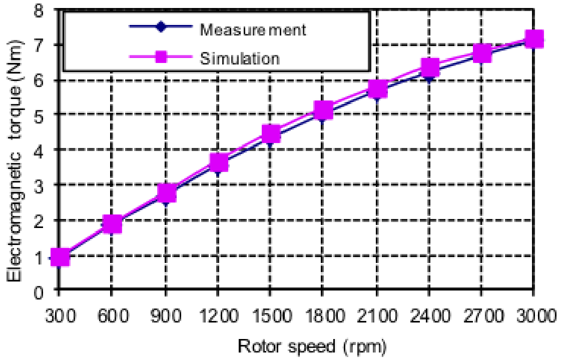
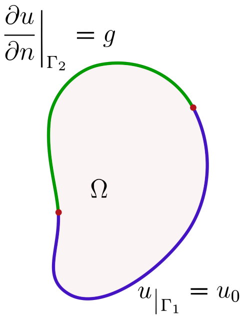


Fig.16. Mean electromagnetic torque vs. rotor speed.

2D FEM: (1/) Problem Formulation (1/4)

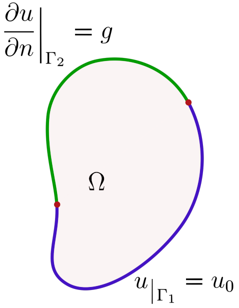
Geometry - Domain of Computation

- $(x, y) \in \Omega$ bounded domain in flat space



2D FEM: (1/) Problem Formulation (2/4)

Two Types of Boundary Conditions: $\Gamma = \Gamma_D \cup \Gamma_N$



- Dirichlet condition on Γ_D

  fix u

  (equivalent of $x = 0$ in 1D)

- Neumann condition on Γ_N

  fix $\frac{\partial u}{\partial n}$

  (equivalent of $x = 1$ in 1D)

2D FEM: (1/) Problem Formulation (3/4)

Boundary Value Problem for
Second Order Differential Equations

- given $(x, y) \in \Omega$ with $\Gamma = \Gamma_D \cup \Gamma_N$ the boundary of Ω
- given: $f(x, y)$ given function and α given number
- find: $u(x, y)$ such that

$$\begin{aligned} -\triangle u(x,y) &= f(x,y) \text{ for } (x,y) \in \Omega \text{ (differential equation on } \Omega) \\ u &= 0 \text{ (Dirichlet boundary condition on } \Gamma_D) \\ \frac{\partial u}{\partial n} &= \alpha \text{ (Neumann boundary condition on } \Gamma_N) \end{aligned}$$

- differential equation invalid on Γ - boundary conditions valid on Γ

2D FEM: (1/11) Problem Formulation (4/4)

Residual function r(x,y)

- definition: $r(x, y) = \triangle u(x, y) + f(x, y)$
- quality of the solution:
  small (large) residual is indication of good (poor) approximation
- solve $-\triangle u(x, y) = f(x, y)$ + b.c. equivalent to
- find $u(x, y)$ such that $r(x, y) = 0$ and $u(x, y)$ satisfies b.c.

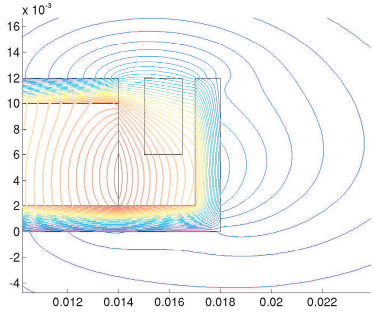
2D FEM: (2/11) Mesh Generation (1/19)

Mesh Ω^h on Ω

- nodes: $\mathbf{x}_i = (x_i, y_i)$
- edges: edge_i
- triangular elements: e_i

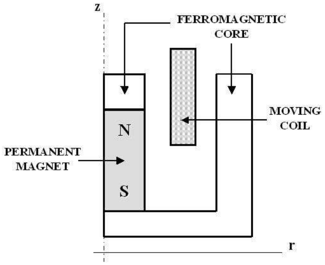
2D FEM: (2/11) Mesh Generation (2/19)

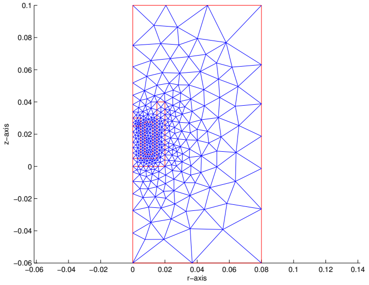
Loudspeaker Example



2D FEM: (2/11) Mesh Generation (3/19)

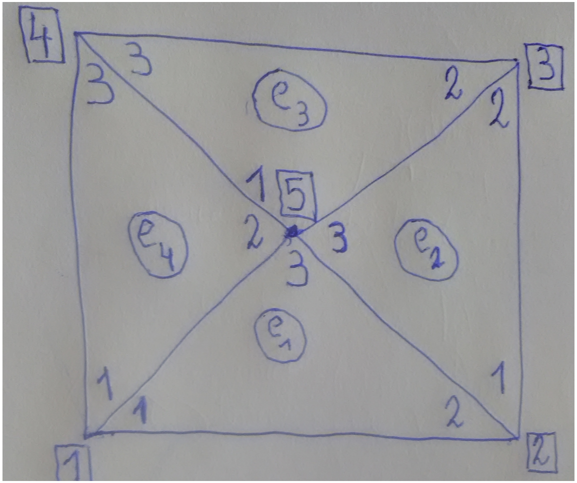
Loudspeaker Example





2D FEM: (2/11) Mesh Generation (4/19)

Four-Element Five-Node Mesh Example



2D FEM: (2/11) Mesh Generation (5/19)

Four-Element Five-Node Mesh Example

Gmsh Output for First Order Elements

- 5 nodes: 4 boundary nodes plus one interior node
- 8 edges (4 boundary edges plus 4 interior edges) not reportedly separately by Gmsh
- 12 elements: 4 triangles + 8 edges

2D FEM: (2/11) Mesh Generation (6/19)

Local and Global Numbering of Nodes

- local numbering: numbering from 1 to 3 on each triangle e_i
- global numbering: numbering from 1 to nnodes on the mesh Ω^h
- local-to-global mapping:

  on each element e_i: given local number, find global number
- see Second Homework Assignment: matrix e
- valuable bookkeeping tool for assembly of matrix and vector

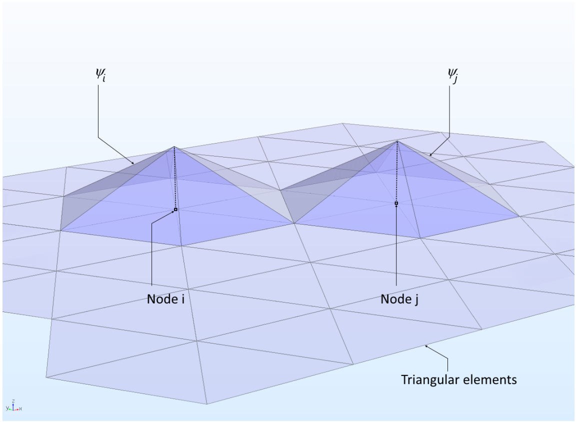
2D FEM: (2/11) Mesh Generation (7/19)

Example of Local and Global Numbering of Nodes
Four-Element Five-Node Mesh Example

- on element e_1
  local nrs 1, 2 and 3 corresponds to global nrs 1, 2 and 5
- on element e_2
  local nrs 1, 2 and 3 corresponds to global nrs 2, 3 and 5
- on element e_3
  local nrs 1, 2 and 3 corresponds to global nrs 5, 3 and 4
- on element e_4
  local nrs 1, 2 and 3 corresponds to global nrs 1, 5 and 4

2D FEM: (2/11) Mesh Generation (8/19)

Shape Functions



2D FEM: (2/11) Mesh Generation (9/19)

Definition of shape function $\phi_i(\mathbf{x}) = \phi_i(x, y)$

- linear Lagrange interpolation function $\phi_i(x, y)$
- linear means that $\phi_i(x, y) = C_1\, x + C_2\, y + C_3$
- each node $\mathbf{x}_i = (x_i, y_i)$ (including boundary nodes) has its $\phi_i(x, y)$
- $\phi_i(\mathbf{x}_j) = \delta_{ij}$ (see figure)

2D FEM: (2/11) Mesh Generation (10/19)

What is the Buzz on Element e_i?

- element e_i has three nodes $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$ (local numbering)
- elements e_i "sees" three linear basis function (local numbering)

$$\phi_1(x,y) = a_1\,x + b_1\,y + c_1$$
$$\phi_2(x,y) = a_2\,x + b_2\,y + c_2$$
$$\phi_3(x,y) = a_3\,x + b_3\,y + c_3$$

- what are a_1, b_1 and c_1? (idem for $\phi_2(x,y)$ and $\phi_3(x,y)$)
- impose $\phi_1(\mathbf{x}_1) = 1$, $\phi_1(\mathbf{x}_2) = 0$ and $\phi_1(\mathbf{x}_3) = 0$

  (idem for $\phi_2(x,y)$ and $\phi_3(x,y)$)

2D FEM: (2/11) Mesh Generation (11/19)

What is the Buzz on Element e_i?

- three conditions $\phi_1(\mathbf{x}_1) = 1$, $\phi_1(\mathbf{x}_2) = 0$ and $\phi_1(\mathbf{x}_3) = 0$ read

$$\begin{pmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{pmatrix} \begin{pmatrix} a_1 \\ b_1 \\ c_1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$$

- idem for $\phi_2(x, y)$ and $\phi_3(x, y)$

$$\begin{pmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{pmatrix} \begin{pmatrix} a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \\ c_1 & c_2 & c_3 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

- Important messsage: coordinate $x_1, \ldots, y_3$ uniquely fix $a_1, \ldots, c_3$!

2D FEM: (2/11) Mesh Generation (12/19)

Exercise

- assume $\phi_i(x, y) = a_i\, x + b_i\, y + c_i$ for $1 \leq i \leq 3$
- compute $\nabla\phi_i(x, y)$ for $1 \leq i \leq 3$
- compute $\nabla\phi_i(x, y) \cdot \nabla\phi_j(x, y)$ for $1 \leq i, j \leq 3$
- compute $A_{e_k} = \int_{e_k} \nabla\phi_i(x, y) \cdot \nabla\phi_j(x, y)\, d\Omega$

2D FEM: (2/11) Mesh Generation (13/19)

Exercise Solution

- assume $\phi_i(x,y) = a_i\,x + b_i\,y + c_i$ for $1 \leq i \leq 3$
- compute $\nabla\phi_i(x,y) = \left(\frac{\partial\phi_i(x,y)}{\partial x}, \frac{\partial\phi_i(x,y)}{\partial y}\right) = (a_i, b_i)$ for $1 \leq i \leq 3$
- compute $\nabla\phi_i(x,y)\cdot\nabla\phi_j(x,y) = \underbrace{a_i\,a_j + b_i\,b_j}_{constant-in-x-and-y}$ for $1 \leq i,j \leq 3$
- compute for $1 \leq i,j \leq 3$

$$A_{e_k} = \int_{e_k} \nabla\phi_i(x,y)\cdot\nabla\phi_j(x,y)\,d\Omega = \text{area}(e_k)\,[a_i\,a_j + b_i\,b_j]$$

- A_{e_k}: contribution to global matrix A on element e_k
- observe that A_{e_k} is independent of c_1, c_2 and c_3

2D FEM: (2/11) Mesh Generation (14/19)

How to Compute the Elementary Matrix Contribution (1/2)

- define on each element the 3-by-3 matrix

$$Emat = \begin{pmatrix} a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \\ c_1 & c_2 & c_3 \end{pmatrix}$$

- compute *Emat* by solving the linear system

$$\begin{pmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{pmatrix} \underbrace{\begin{pmatrix} a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \\ c_1 & c_2 & c_3 \end{pmatrix}}_{=Emat} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

- set third row of *Emat* equal to zero or $Emat[3,:].=0$

$$Emat = \begin{pmatrix} a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \\ 0 & 0 & 0 \end{pmatrix}$$

2D FEM: (2/11) Mesh Generation (15/19)

How to Compute the Elementary Matrix Contribution (2/2)

- then

$$Emat^T\ Emat = [a_i\, a_j + b_i\, b_j] \text{ for } 1 \leq i, j \leq 3$$

- and therefore contribution to global matrix A on element e_k

$$\text{area}(e_k)\ Emat^T\ Emat = \text{area}(e_k)\, [a_i\, a_j + b_i\, b_j] \text{ for } 1 \leq i, j \leq 3$$

2D FEM: (2/11) Mesh Generation (16/19)

Computation of the Coefficients $a_1, \ldots, b_3$ Revisited (1/4)

- motivation: solving the linear system for *Emat* on each element likely computationally expensive
- alternative: solve symbolically using Cramer's Rule instead
- linear system for a_1 and b_1 (coefficient c_1 not required)

$$\underbrace{\begin{pmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{pmatrix}}_{=X} \begin{pmatrix} a_1 \\ b_1 \\ c_1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$$

- $$a_1 = \frac{\det(X_1)}{\det(X)} \text{ and } b_1 = \frac{\det(X_2)}{\det(X)} \text{ and } c_1 \text{ not required}$$

2D FEM: (2/11) Mesh Generation (17/19)

Computation of the Coefficients $a_1, \dots, b_3$ Revisited (2/4)

- $$\begin{aligned}\det(X) &= \det\begin{pmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{pmatrix} = \det\begin{pmatrix} x_1 - x_3 & y_1 - y_3 & 0 \\ x_2 - x_3 & y_2 - y_3 & 0 \\ x_3 & y_3 & 1 \end{pmatrix} \\ &= (x_1 - x_3)(y_2 - y_3) - (x_2 - x_3)(y_1 - y_3) \\ &= 2\,\text{signed-area}(e_k)\end{aligned}$$

- $\det(X_1) = \det\begin{pmatrix} 1 & y_1 & 1 \\ 0 & y_2 & 1 \\ 0 & y_3 & 1 \end{pmatrix} = y_2 - y_3$

  $\det(X_2) = \det\begin{pmatrix} x_1 & 1 & 1 \\ x_2 & 0 & 1 \\ x_3 & 0 & 1 \end{pmatrix} = -(x_2 - x_3) = x_3 - x_2$

2D FEM: (2/11) Mesh Generation (18/19)

How to Compute the Coefficients $a_1, \ldots, b_3$ (3/4)

- thus

$$a_1 = \frac{\det(X_1)}{\det(X)} = \frac{y_2 - y_3}{2\,\text{signed-area}(e_k)}$$

- and

$$b_1 = \frac{\det(X_2)}{\det(X)} = \frac{x_3 - x_2}{2\,\text{signed-area}(e_k)}$$

- similarly for a_2, a_3, b_2 and b_3

2D FEM: (2/11) Mesh Generation (19/19)

How to Compute the Coefficients $a_1, \ldots, b_3$ (4/4)

- local matrix A_{e_k} can be computed from nodal coordinates without solving 3-by-3 linear systems

2D FEM: (3/11) Linear Combination of Shape Functions (1/3)

Linear Combination of Shape Functions

- set of functions $\{\phi_i(x) | 1 \leq i \leq n\}$ where $n = nnodes$
- linear combinations of these functions $\phi_i(x)$ can be made
- $V_0^h(\Omega)$: function space defined by all linear combinations

$$V_0^h(\Omega) = \text{span}\{\phi_1(x,y), \ldots, \phi_n(x,y)\}$$

- $u^h(x,y) \in V_0^h(\Omega)$: there exists coordinates $c_1, \ldots, c_n$ such that

$$u^h(x,y) = c_1\,\phi_1(x,y) + \ldots + c_n\,\phi_n(x,y)$$

2D FEM: (3/11) Linear Combination of Shape Functions (2/3)

Application to Finite Elements

- $u(x, y)$: exact solution of the boundary value problem
- $u^h(x, y)$: finite element approximation to $u(x)$ computed on Ω^h
- $u^h(x, y) \in V_0^h(\Omega) = \text{span}\{\phi_1(x, y), \ldots, \phi_n(x, y)\}$
- $u^h(x, y) = 0$ on Γ_D by definition of $V_0^h(\Omega)$
- expansion of $u^h(x)$ as linear combination of shape function

$$u^h(x, y) = c_1\,\phi_1(x, y) + \ldots + c_n\,\phi_n(x, y)$$

2D FEM: (3/11) Linear Combination of Shape Functions (3/3)

Application to Finite Elements

- expansion of $u^h(x)$ as linear combination of shape function

$$u^h(x,y) = c_1\,\phi_1(x,y) + \ldots + c_n\,\phi_n(x,y)$$

- $c_1, \ldots, c_n$ coordinates - $\phi_1(x,y), \ldots, \phi_n(x,y)$ basis functions
- basis functions: unique determined by the mesh
- coordinates: to by determined by solving a linear system

  one coordinate for each node $\mathbf{x}_i$ in the mesh

2D FEM: (4/11) Strong vs. Weak Equal to Zero (1/2)

Weak or Variational Formulation

- n basis functions $\phi_i(x,y)$ defined by the mesh Ω^h
- inner product: $< g(x,y), \phi_i(x,y) >= \int_\Omega g(x,y)\, \phi_i(x,y)\, d\Omega$
- $< g(x,y), \phi_i(x,y) >$ coordinate of $g(x,y)$ along the basis function $\phi_i(x,y)$
- numerically: essential in remainder of the course

$$\boxed{g(x,y) = 0 \text{ in discrete weak form}}$$

$$\boxed{\Leftrightarrow \forall \mathbf{x}_i \in \Omega^h :< g(x,y), \phi_i(x,y) >= 0}$$

n equations indexed by i where $n = nnodes$

2D FEM: (4/11) Strong vs. Weak Equal to Zero (2/2)

Applied to Finite Elements

- choose $g(x,y) = r(x,y) = \triangle u(x,y) + f(x,y)$
- enforce $g(x,y) = 0$ plus boundary conditions in discrete weak form

$$
\begin{aligned}
< g(x,y), \phi_i(x,y) > &= 0 \quad \text{for } 1 \le i \le n \\
\Leftrightarrow \int_\Omega g(x,y)\,\phi_i(x,y)\,d\Omega &= 0 \quad \text{for } 1 \le i \le n \\
\Leftrightarrow \int_\Omega -\triangle\, u(x,y)\,\phi_i(x,y)\,d\Omega &= \int_\Omega f(x,y)\,\phi_i(x,y)\,d\Omega \quad \text{for } 1 \le i \le n \\
&\quad \text{where } n = nnodes
\end{aligned}
$$

2D FEM: (5/11) Calculus of functions in 2 vars (1/12)

Two Small Exercises

- recap: Block (2/3) of this course: $F(x) = v\frac{du}{dx}$
- suppose now that $\mathbf{F} = v\,\nabla u = v\,\mathrm{grad}(u)$
- Exercise 1: compute $\mathrm{div}\,\mathbf{F} = \nabla\cdot\mathbf{F} = \nabla\cdot(v\,\nabla u)$
- Exercise 2: compute $\mathbf{F}\cdot\mathbf{n} = (v\,\nabla u)\cdot\mathbf{n}$

2D FEM: (5/11) Calculus of functions in 2 vars (2/12)

Two Small Exercises: Exercise One

- $\nabla u = \text{grad}(u)$ is a vector function

$$\nabla u = \left(\frac{\partial u}{\partial x}, \frac{\partial u}{\partial y}\right) = \frac{\partial u}{\partial x}\mathbf{i} + \frac{\partial u}{\partial y}\mathbf{j}$$

- $\mathbf{F} = v\,\nabla u$ is scalar times vector and thus again a vector function (multiply each component of vector ∇u with scalar v)

$$\mathbf{F} = v\,\nabla u = v\,\left(\frac{\partial u}{\partial x}, \frac{\partial u}{\partial y}\right) = \left(v\,\frac{\partial u}{\partial x}, v\,\frac{\partial u}{\partial y}\right)$$

- note that $\mathbf{F}$ is a vector function with two components

$$F_x = v\,\frac{\partial u}{\partial x} \quad \text{and} \quad F_y = v\,\frac{\partial u}{\partial y}$$

2D FEM: (5/11) Calculus of functions in 2 vars (3/12)

Two Small Exercises: Exercise One

- $\operatorname{div} \mathbf{F} = \nabla \cdot \mathbf{F} = \nabla \cdot (v \nabla u)$ is a scalar function

$$\begin{aligned}
\operatorname{div} \mathbf{F} = \nabla \cdot \mathbf{F} = \nabla \cdot \left(v \frac{\partial u}{\partial x}, v \frac{\partial u}{\partial y} \right) \\
&= \frac{\partial}{\partial x}\left(v \frac{\partial u}{\partial x} \right) + \frac{\partial}{\partial y}\left(v \frac{\partial u}{\partial y} \right) \\
&= v \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right) + \left(\frac{\partial u}{\partial x}\frac{\partial v}{\partial x} + \frac{\partial u}{\partial y}\frac{\partial v}{\partial y} \right) \\
&= v \triangle u + \nabla u \cdot \nabla v \\
&= v \, \mathrm{Laplacian}(u) + \underbrace{\mathrm{grad}(u) \cdot \mathrm{grad}(v)}_{inner-product-of-vectors}
\end{aligned}$$

2D FEM: (5/11) Calculus of functions in 2 vars (4/12)

Two Small Exercises: Exercise 2

- $\mathbf{F} \cdot \mathbf{n}$ is the inner product of $\mathbf{F}$ and $\mathbf{n}$

$$\begin{aligned}
\mathbf{F} \cdot \mathbf{n} &= (v\,\nabla u) \cdot \mathbf{n} \quad \text{(definition of } \mathbf{F}\text{)} \\
&= v \underbrace{(\nabla u \cdot \mathbf{n})}_{inner-product-of-vectors} \quad \text{(change order of operations)} \\
&= v\,(\frac{\partial u}{\partial x}\,n_x + \frac{\partial u}{\partial y}\,n_y) \quad \text{(inner product explicitly)} \\
&= v\,\frac{\partial u}{\partial n} \quad \text{(definition of the normal product)}
\end{aligned}$$

2D FEM: (5/11) Calculus of functions in 2 vars (5/12)

Two Small Exercises: Summary

- suppose that $\mathbf{F} = v\,\nabla u$
- Ex 1: solution div $\mathbf{F} = \nabla \cdot \mathbf{F} = \nabla u \cdot \nabla v + v\,\triangle\,u$
- Ex 2: solution $\mathbf{F} \cdot \mathbf{n} = (v\,\nabla u) \cdot \mathbf{n} = v\,(\nabla u \cdot \mathbf{n}) = v\,\frac{\partial u}{\partial n}$
- these results will be used in the next two slides

2D FEM: (5/11) Calculus of functions in 2 vars (6/12)

Recap: Integration of Function in One Variable

- assume $0 < x < 1$ or $x \in \Omega = (0, 1)$ and $F'(x) = \dfrac{d\,F(x)}{dx}$

$$\int_\Omega F'(x)\,dx = \underbrace{\int_0^1 F'(x)\,dx}_{1D-line-integral} = [F(x)]_0^1 = \underbrace{F(1) - F(0)}_{0D-point-evaluation}$$

- choose $F(x) = v(x)\,u'(x)$ (u has prime - v has no prime)
- arrive at integration by parts formula

$$-\int_0^1 u''(x)\,v(x)\,dx = \int_0^1 u'(x)\,v'(x)\,dx - [u'(x)\,v(x)]_0^1$$

- wish - goal - dream - ambitution: repeat in 2D

2D FEM: (5/11) Calculus of functions in 2 vars (7/12)

Integration by Parts in two variables

- Gauss Integration Theorem or Divergence Theorem

  $(x, y) \in \Omega$ with boundary Γ

$$\underbrace{\int_{\Omega} \nabla \cdot \mathbf{F} \, d\Omega}_{2D-surface-integral} = \underbrace{\int_{\Gamma} \mathbf{F} \cdot \mathbf{n} \, ds}_{1D-line-integral}$$

- see calculus textbook or wiki
- choose: $\mathbf{F} = v \nabla u = (v \frac{\partial u}{\partial x}, v \frac{\partial u}{\partial y})$
- requires $\nabla \cdot \mathbf{F}$ and $\mathbf{F} \cdot \mathbf{n}$ from exercise

2D FEM: (5/11) Calculus of functions in 2 vars (8/12)

Integration by Parts in two variables

- then

$$\int_\Omega \nabla \cdot \mathbf{F}\, d\Omega = \int_\Omega \nabla \cdot (v\, \nabla u)\ d\Omega = \int_\Omega [\nabla u \cdot \nabla v + v\ \triangle u]\ d\Omega$$
$$= \int_\Gamma \mathbf{F} \cdot \mathbf{n}\, ds = \int_\Gamma \frac{\partial u}{\partial n}\, v\, ds$$

- after rearranging terms:

$$\int_\Omega (-\triangle u)\, v\, d\Omega = \int_\Omega \nabla u \cdot \nabla v\, d\Omega - \int_\Gamma \frac{\partial u}{\partial n}\, v\, ds$$

2D FEM: (5/11) Calculus of functions in 2 vars (9/12)

Derivative: Integration by Parts in two variables

- integration by parts formula becomes

$$\boxed{\int_\Omega (-\triangle u)\, v \, d\Omega = \int_\Omega \nabla u \cdot \nabla v \, d\Omega - \int_\Gamma \frac{\partial u}{\partial n} v \, ds}$$

- observe that as before:

  LHS: double derivatives in *u* - no derivatives on *v*

  LHS: minus sign

  RHS: first order derivatives on both *u* and *v* - additional term on the boundary

2D FEM: (5/11) Calculus of functions in 2 vars (10/12)

Quadrature by Trapezoidal Rule

- trapezoidal rule: e_k triangle with vertices $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$ (local numbering)

$$\int_{e_k} g(x,y)\, d\Omega \approx \frac{\text{area}(e_k)}{3}\left[g(\mathbf{x}_1) + g(\mathbf{x}_2) + g(\mathbf{x}_3)\right]$$

- $\text{area}(e_k)$ can be computed using $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$ as input
- more accurate rules exist (e.g. Gauss quadrature, see references)

2D FEM: (5/11) Calculus of functions in 2 vars (11/12)

Exercise

- assume that $g(x, y) = f(x, y)\,\phi_i(x, y)$ for $1 \leq i \leq 3$
- compute

$$\begin{pmatrix} \int_{e_k} f(x,y)\,\phi_1(x,y)\,d\Omega \\ \int_{e_k} f(x,y)\,\phi_2(x,y)\,d\Omega \\ \int_{e_k} f(x,y)\,\phi_3(x,y)\,d\Omega \end{pmatrix}$$

2D FEM: (5/11) Calculus of functions in 2 vars (12/12)

Exercise Solution

- assume that $g(x,y) = f(x,y)\,\phi_i(x,y)$ for $1 \le i \le 3$
- compute

$$\begin{pmatrix} \int_{e_k} f(x,y)\,\phi_1(x,y)\,d\Omega \\ \int_{e_k} f(x,y)\,\phi_2(x,y)\,d\Omega \\ \int_{e_k} f(x,y)\,\phi_3(x,y)\,d\Omega \end{pmatrix} \approx \frac{\text{area}(e_k)}{3} \begin{pmatrix} f(\mathbf{x}_1) \\ f(\mathbf{x}_2) \\ f(\mathbf{x}_3) \end{pmatrix}$$

- contribution to global vector **f** on element e_k

2D FEM: (6/11) Discrete Weak Form (1/6)

Apply Integration by Parts on Weak Formulation

- earlier we set the residual $r(x,y) = \triangle u''(x,y) + f(x,y)$ to zero in weak form and arrived at

$$\int_\Omega (-\triangle u)\,\phi_i(x,y)\,d\Omega = \int_\Omega f(x,y)\,\phi_i(x,y)\,d\Omega \quad \text{for all } 1 \leq i \leq n$$

- apply integration by part to the LHS

$$\int_\Omega \nabla u \cdot \nabla \phi_i(x,y)\,d\Omega = \int_\Omega f(x,y)\,\phi_i(x,y)\,d\Omega + \int_\Gamma \frac{\partial u}{\partial n}(x,y)\,\phi_i(x,y)\,ds$$

for all $1 \leq i \leq n$ where n=nnodes

observe: minus sign in LHS disappeared

- first order derivative on both $u(x)$ and $\phi_i(x)$

2D FEM: (6/11) Discrete Weak Form (2/6)

- Dirichlet boundary conditions: $u = 0$ on Γ_D
- Neumann boundary conditions: $\dfrac{\partial u}{\partial n} = \alpha$ on Γ_N
- Dirichlet and Neumann boundary conditions treated differently
- boundary term in the RHS of the weak form

$$\begin{aligned}\int_\Gamma \frac{\partial u}{\partial n}(x,y)\,\phi_i(x,y)\,ds &= \int_{\Gamma_D} \frac{\partial u}{\partial n}(x,y)\,\phi_i(x,y)\,ds \\ &+ \int_{\Gamma_N} \underbrace{\frac{\partial u}{\partial n}(x,y)}_{=\alpha}\,\phi_i(x,y)\,ds\end{aligned}$$

- we thus obtain that

$$\int_\Gamma \frac{\partial u}{\partial n}(x,y)\,\phi_i(x,y)\,ds = \int_{\Gamma_D} \frac{\partial u}{\partial n}(x,y)\,\phi_i(x,y)\,ds + \int_{\Gamma_N} \alpha\,\phi_i(x,y)\,ds$$

2D FEM: (6/11) Discrete Weak Form (3/6)

- Dirichlet boundary conditions: impose that $\phi_i(x,y) = 0$ on Γ_D
- we thus obtain that

$$\int_{\Gamma} \frac{\partial u}{\partial n}(x,y)\,\phi_i(x,y)\,ds = \int_{\Gamma_N} \alpha\,\phi_i(x,y)\,ds$$

- discrete weak form becomes

$$\int_{\Omega} \nabla u \cdot \nabla \phi_i(x,y)\,d\Omega = \int_{\Omega} f(x,y)\,\phi_i(x,y)\,d\Omega + \int_{\Gamma_N} \alpha\,\phi_i(x,y)\,ds$$

for all $1 \leq i \leq n$

2D FEM: (6/11) Discrete Weak Form (4/6)

Discrete Weak Form Becomes for $1 \leq i \leq n$

$$\int_\Omega \nabla u(x,y) \cdot \nabla \phi_i(x,y)\, d\Omega = \int_\Omega f(x,y)\,\phi_i(x,y)\, d\Omega + \int_{\Gamma_N} \alpha\,\phi_i(x,y)\, ds$$

- assume $u(x,y)$ approximate by $u^h(x,y)$ where $u^h(x,y) = \sum_{j=1}^n c_j\,\phi_j(x,y)$
- thus $u'(x)$ approximate by $\nabla u^h(x,y) = \sum_{j=1}^n c_j\,\nabla\phi_j(x,y)$
- then for $1 \leq i \leq n$

$$\sum_{j=1}^n \int_\Omega \nabla\phi_j(x,y) \cdot \nabla\phi_i(x,y)\, dx\, c_j = \int_\Omega f\,\phi_i\, d\Omega + \int_{\Gamma_N} \alpha\,\phi_i(x,y)\, ds$$

2D FEM: (6/11) Discrete Weak Form (6/6)

Discrete Weak Form Becomes

- for $1 \leq i \leq n$ where $n = nnodes$

$$\sum_{j=1}^{n} \int_{\Omega} \nabla\phi_j(x,y) \cdot \nabla\phi_i(x,y)\, d\Omega\, c_j = \int_{\Omega} f(x,y)\,\phi_i(x,y)\, d\Omega + \int_{\Gamma_N} \alpha\,\phi_i(x,y)\, ds$$

- can be written in the form: for $1 \leq i \leq n$
  index i counts equations - index j counts unknowns

$$\sum_{j=1}^{n} A_{ij}\, c_j = f_i$$

- and thus as a n by n linear system

$$A\,\mathbf{c} = \mathbf{f}$$

2D FEM: (7/11) Linear System Formulation (1/3)

Expression for Matrix and Vector Elements

- Matrix elements:

$$A_{ij} = \int_\Omega \nabla\phi_j(x,y)\cdot\nabla\phi_i(x,y)\,d\Omega \text{ for } 1 \le i,j \le n$$

- Vector elements:

$$f_i = \int_\Omega f(x,y)\,\phi_i(x,y)\,d\Omega + \int_{\Gamma_N} \alpha\,\phi_i(x,y)\,ds \text{ for } 1 \le i \le n$$

2D FEM: (7/11) Linear System Formulation (2/3)

Properties of Matrix A

- A is large

  $n > 1e6$ in 3D applications in no exception

- A is sparse

  A contains many zero elements (cfr. 2D finite difference method)

- A many other cool properties $\Rightarrow$ fast solvers for $A\mathbf{c} = \mathbf{f}$ exist

2D FEM: (7/11) Linear System Formulation (3/3)

Summary: Matrix A and Right-Hand Vector $\mathbf{f}$

Treatment of the Boundary Conditions

- Dirichlet boundary conditions: $u(x, y) = 0$:

  modify equations corresponding to the boundary nodes in linear system

  see finite difference method

- Neumann boundary conditions: $\frac{\partial u}{\partial n} = \alpha$ on Γ_N

  add term $\int_{\Gamma_N} \alpha \, \phi_i(x, y) \, ds$ to vector $\mathbf{f}$

  see lab sessions for details

2D FEM: (8/11) Element-by-Element Construction of the Vector (1/2)

How does element e_k contribute to the vector $\mathbf{f}$?

- $f_{e_k} \in \mathbb{R}^3$ contribution of element e_i to global vector $\mathbf{f}$

  using local numbering of the three nodes on the element e_k

$$f_{e_k} = \begin{pmatrix} \int_{e_k} f(x,y)\,\phi_1(x,y)\,dx \\ \int_{e_k} f(x,y)\,\phi_2(x,y)\,dx \\ \int_{e_k} f(x,y)\,\phi_3)x,y)\,dx \end{pmatrix}$$

- use trapezoidal rule of integration (see earlier)

$$f_{e_k} = \begin{pmatrix} \int_{e_k} f(x,y)\,\phi_1(x,y)\,d\Omega \\ \int_{e_k} f(x,y)\,\phi_2(x,y)\,d\Omega \\ \int_{e_k} f(x,y)\,\phi_3(x,y)\,d\Omega \end{pmatrix} \approx \frac{\text{area}(e_k)}{3} \begin{pmatrix} f(\mathbf{x}_1) \\ f(\mathbf{x}_2) \\ f(\mathbf{x}_3) \end{pmatrix}$$

- given mesh Ω^h and source $f(x)$, f_{e_k} for each e_k can be computed

2D FEM: (8/11) Element-by-Element Construction of the Vector (2/2)

Finite Element Assembly of the Vector $\mathbf{f}$

- loop over all of the N elements e_k in the mesh Ω^h
- on e_k compute the local element vector $f_{e_k} \in \mathbb{R}^3$
  the local element vector has three components
- add local element vector to the global vector $\mathbf{f} \in \mathbb{R}^n$
  the global vector $\mathbf{f}$ has n components where $n = nnodes$
- $\mathbf{f} = \mathbf{f} + f_{e_k}$
  assembly requires taking the mesh connectivity into account
  connectivity here refers to mapping from local to global numbering on the element e_k

2D FEM: (9/11) Element-by-Element Construction of the Matrix (1/2)

How does element e_k contribute to the vector A?

- $A_{e_k} \in \mathbb{R}^{3\times 3}$ contribution of element e_j to global vector A

$$A_{e_k} = \left(\int_{e_k} \nabla\phi_i(x,y) \cdot \nabla\phi_j(x,y)\, d\Omega\right)_{1\leq i,j\leq 3}$$

- using derivative of the shape functions (see earlier)

$$A_{e_k} = \text{area}(e_k)\,\left(a_i\, a_j + b_i\, b_j\right)_{1\leq i,j\leq 3}$$

2D FEM: (9/11) Element-by-Element Construction of the Matrix (2/2)

Finite Element Assembly of the Matrix A

- loop over all of the N elements e_k in the mesh Ω^h
- on e_i compute the local element matrix $A_{e_k} \in \mathbb{R}^{3\times 3}$
  the local element matrix has three by three components
- add local element matrix to the global matrix $A \in \mathbb{R}^{n\times n}$
  the global matrix A has n by n components where $n = nnnodes$
- $A = A + A_{e_k}$
  assembly requires taking the mesh connectivity into account
  connectivity here refers to mapping from local to global numbering on the element e_k

2D FEM: (10/11) Implementation in Julia (1/5)

- see Lab Session

2D FDM: (10/11) Implementation in Julia (2/5)

Retrieve nodes from the gmsh mesh data structure

```
#..retrieve node identification numbers and node x, y,and z-coordinates
#..from gmsh data structure
#..the getNodes() function has two output argument
#..first output argument: node_ids: the node numbers
#..second output argument: node_coord: the x, y,and z-coordinates
#..note that coordinates are stored continguously

node_ids, node_coord, _ = gmsh.model.mesh.getNodes()
nnodes = length(node_ids)
xnode = node_coord[1:3:end]
ynode = node_coord[2:3:end]
```

2D FDM: (10/11) Implementation in Julia (3/5)

Retrieve mesh from the gmsh mesh data structure

```
#..retrieve element_types, element identification numbers and element connectivity
#..from gmsh data structure
#..the getElements() function has one input and three output arguments
#..input argument: specify elements on the surface (2) or on the boundary (1)
#..first output argument: element_type: the type of elements
#..second output argument: element_ids: element identification number
#..third output argument: element_connectivity:element connectivity or local-to-glo
#..note that elements are stored continguously

element_types, element_ids, element_connectivity = gmsh.model.mesh.getElements(2)
nelements = length(element_ids[1])
```

2D FDM: (10/11) Implementation in Julia (4/5)

Assembly of vector

```
#..initialize global vector f and local floc
f = zeros(nnodes,1)
floc = zeros(3,1)

#..loop over all elements
for element_id in 1:nelements

  %..more precise computation of the area of the element will be given later
  element_area = 1/nelements

  %..assume f(x) = 1
  floc = element_area/3*[1;1;1]

  %..perform loop over nodes of the elements
  %..and add local contribution to global entity
  for i = 1:3
    I = element_connectivity[1][3*(element_id-1)+i]
    f[I] += floc[i]
  end

end
```

2D FDM: (10/11) Implementation in Julia (5/5)

Assembly of matrix

```
#..initialize global matrix A and local matrix Aloc
A = zeros(nnodes,nnodes)
Aloc = zeros(3,3)

#..loop over all elements
for element_id in 1:nelements

  %..more precise computation of the area of the element will be given later
  element_area = 1/16

  %..see lecture for more details
  Aloc = (1/element_area)*[1 -1 -1 ; -1 1 0 ; -1 0  1]

  %..perform loop over nodes of the elements
  %..and add local contribution to global entity
  for i = 1:3
    I = element_connectivity[1][3*(element_id-1)+i]
    for j = 1:3
      J = element_connectivity[1][3*(element_id-1)+j]
      A[I,J] += Aloc[i,j]
    end
  end

end
```

2D FEM: (11/11) References

- Gauss Integration Theorem or Divergence Theorem:
  `https://en.wikipedia.org/wiki/Divergence_theorem`
- Gmsh Julia tutorials:
  `https://gitlab.onelab.info/gmsh/gmsh/-/tree/gmsh_4_8_1/tutorial/julia`