Finite Element Method

Julian Fritzsch
Department of Quantum Matter Physics, University of Geneva, CH-1211 Geneva, Switzerland and
School of Engineering, University of Applied Sciences of Western Switzerland, CH-1950 Sion, Switzerland

I. GENERAL IDEA

This chapter is basically a summary of [1, 2]. For a motivation of the finite element method we use the heat equation

$$-\nabla^2 f(\boldsymbol{x}) = \rho(\boldsymbol{x}), \qquad x \in \Omega, \tag{1}$$
$$f(\boldsymbol{x}) = f_D(\boldsymbol{x}), \qquad x \in \partial\Omega_D, \tag{2}$$
$$\nabla f(\boldsymbol{x}) \cdot \boldsymbol{n} = f_N(\boldsymbol{x}), \qquad x \in \partial\Omega_N, \tag{3}$$

where $\partial\Omega_D$ is the part of the boundary where Dirichlet conditions are applied and $\partial\Omega_N$ is the part where Neumann conditions are applied ($\boldsymbol{n}$ is the outward facing normal). This equation can be turned into the so-called weak form by multiplying with an arbitrary test function $g((x))$ and integrate over Ω

$$-\int g(\boldsymbol{x})\nabla(\nabla f(\boldsymbol{x}))\mathrm{d}\Omega = -\int_{\partial\Omega} g(\boldsymbol{x})\nabla f(\boldsymbol{x})\boldsymbol{n}\mathrm{d}\boldsymbol{x} + \int \nabla g(\boldsymbol{x})(\nabla f(\boldsymbol{x}))\mathrm{d}\boldsymbol{x} = \int g(\boldsymbol{x})\rho(\boldsymbol{x})\mathrm{d}\Omega. \tag{4}$$

We demand that the test functions vanish on the boundary with Dirichlet conditions applied: $g(\boldsymbol{x}) = 0,\ \boldsymbol{x} \in \partial\Omega_D$. It follows that

$$\int \nabla g(\boldsymbol{x})(\nabla f(\boldsymbol{x}))\mathrm{d}\boldsymbol{x} = \int_{\partial\Omega_N} g(\boldsymbol{x}) f_N(\boldsymbol{x})\mathrm{d}\boldsymbol{x} + \int g(\boldsymbol{x})\rho(\boldsymbol{x}). \tag{5}$$

To discretize the functions we rewrite them as

$$f(\boldsymbol{x}) \approx \sum_{i\in\mathcal{N}} f_i u_i(\boldsymbol{x}), \qquad g(\boldsymbol{x}) \approx \sum_{i\in\mathcal{N}} g_i u_i(\boldsymbol{x})\mathrm{d}\Omega, \tag{6}$$

where $\mathcal{N}$ denotes the set of all nodes, f_i and g_i are the nodal values and $u_i(\boldsymbol{x})$ are the shape functions that form a basis of the vector space. We demand that the test functions fulfill $u_i(\boldsymbol{x}_j) = \delta_{ij}$, where $\boldsymbol{x}_i$ is the position of the ith node. Inserting (6) into (5) yields

$$\sum_{i,j} g_i f_j \int \nabla u_i(\boldsymbol{x})(\nabla u_j(\boldsymbol{x}))\mathrm{d}\Omega = \sum_i g_i \int_{\partial\Omega_N} u_i(\boldsymbol{x}) f_N(\boldsymbol{x})\mathrm{d}\boldsymbol{x} + \sum_i g_i \int u_i(\boldsymbol{x})\rho(\boldsymbol{x})\mathrm{d}\Omega. \tag{7}$$

The above equation can be written in linearized form

$$\boldsymbol{g}^\top \boldsymbol{K}\boldsymbol{f} = \boldsymbol{g}^\top \boldsymbol{\rho}. \tag{8}$$

As this equation must hold for arbitrary $\boldsymbol{g}$ the problem can be solved by solving the linear equation

$$\boldsymbol{K}\boldsymbol{f} = \boldsymbol{\rho}. \tag{9}$$

The stiffness matrix $\boldsymbol{K}$ is given by

$$\boldsymbol{K}_{ij} = \int \nabla u_i(\boldsymbol{x})(\nabla u_j(\boldsymbol{x}))\mathrm{d}\Omega, \tag{10}$$

the force vector $\boldsymbol{\rho}$ is given by

$$\boldsymbol{\rho} = \int_{\partial\Omega_N} u_i(\boldsymbol{x}) f_N(\boldsymbol{x})\mathrm{d}\boldsymbol{x} + \int u_i(\boldsymbol{x})\rho(\boldsymbol{x})\mathrm{d}\Omega. \tag{11}$$

To calculate the integrals we separate the space into elements (in our case triangles) and evaluate the integrals on each element. This means the integration is turned into a sum of integrals over the different elements $\int \mathrm{d}\Omega \to \sum_E \int \mathrm{d}\Omega_E$.

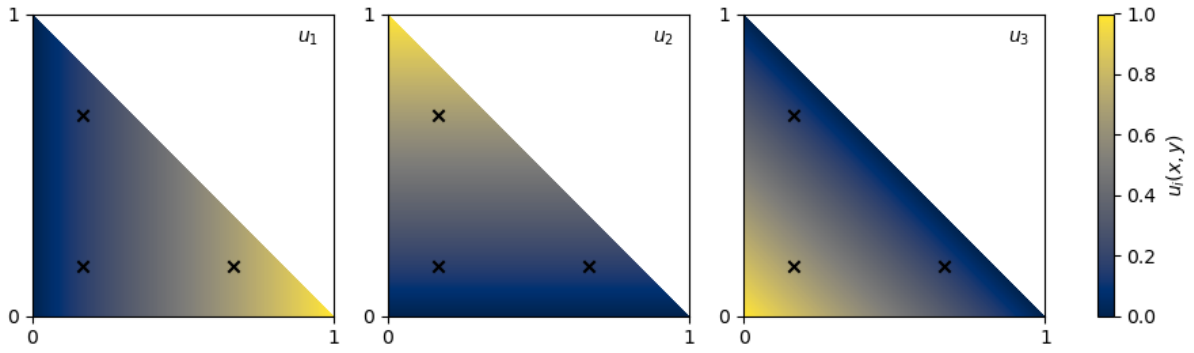

FIG. 1. Shape of the three shape functions for the unit triangle. The crosses show the location of the quadrature points.

The calculations can be simplified by defining the test functions on a reference element and then transforming them to fit the elements on our grid. Typical vector spaces to approximate the solution space are the vector spaces of polynomials. For our investigations we use linear polynomials. Instead of using the basis $u_i = 1, x, y$ we choose Lagrange polynomials that are defined by their nodal values. For the unit triangle with nodes $P_1 = (0, 0)$, $P_2 = (0, 1)$, and $P_3 = (1, 0)$ they are given by

$$\hat{u}_1(\hat{\boldsymbol{x}}) = \hat{x}, \quad \hat{u}_2(\hat{\boldsymbol{x}}) = \hat{y}, \quad \hat{u}_3(\hat{\boldsymbol{x}}) = 1 - \hat{x} - \hat{y}. \tag{12}$$

The hat denotes that the quantity corresponds to a reference quantity. It is straight forward to show that these functions satisfy our requirements. The shape of the shape functions is shown in Fig. 1. To transform the values of the shape functions of arbitrary cells onto the reference cell we map the reference position vector to the position vectors of the cell as

$$\boldsymbol{x} = \boldsymbol{a} + \boldsymbol{B}\hat{\boldsymbol{x}} = h(\hat{\boldsymbol{x}}), \tag{13}$$

where $\boldsymbol{a}$ and $\boldsymbol{B}$ are an appropriate vector and matrix, respectively. An important property is that

$$u_i(\boldsymbol{x}) = u_i(h(\hat{\boldsymbol{x}})) = \hat{u}_i(\hat{\boldsymbol{x}}). \tag{14}$$

The gradient and the integration measure become

$$\nabla u_i(\boldsymbol{x}) = \frac{1}{\det(J\hat{\boldsymbol{x}})}\nabla\hat{u}_i(\hat{\boldsymbol{x}}), \quad \mathrm{d}\Omega_E = \det(J(\hat{\boldsymbol{x}}))\mathrm{d}\hat{\boldsymbol{x}}, \tag{15}$$

where $J(\boldsymbol{x})$ is the Jacobian of $h(\boldsymbol{x})$. For a more detailed description see [2].

Finally we use a quadrature rule to evaluate the integrals. This means the integral is approximated by weighted sum of the function evaluated at the quadrature points. In our case we are using three quadrature points as shown in Fig. 1 with weights 1/6. The integrals will then turn into sums over cells and quadrature points.

$$\sum_E \int k\nabla u_i \nabla u_j \mathrm{d}\Omega_E = \sum_E \sum_q \nabla\hat{u}_i(\hat{\boldsymbol{x}}_q)\nabla\hat{u}_j(\hat{\boldsymbol{x}}_q)\omega_q \frac{1}{\det(J(\hat{\boldsymbol{x}}_q))}, \tag{16}$$

where $\hat{\boldsymbol{x}}_q$ are the quadrature points, $\omega(\hat{\boldsymbol{x}}_q)$ are the quadrature weights. The advantage is that we now only need to keep track of the transform between the cells and the references cell and do not need to evaluate the shape functions for each cell.

II. APPLICATION TO THE SWING EQUATIONS

We want to apply the finite element method to our continuum model [3]

$$m(\boldsymbol{x})\ddot{\theta}(\boldsymbol{x}) + d(\boldsymbol{x})\dot{\theta}(\boldsymbol{x}) = p(\boldsymbol{x}) + \nabla\left(\begin{bmatrix} b_x(\boldsymbol{x}) & 0 \\ 0 & b_y(\boldsymbol{x}) \end{bmatrix}\nabla\theta(\boldsymbol{x})\right), \tag{17}$$

where we have assumed that the susceptance tensor is diagonal. The boundary conditions are given by

$$\boldsymbol{n}\left(\begin{bmatrix} b_x(\boldsymbol{x}) & 0 \\ 0 & b_y(\boldsymbol{x}) \end{bmatrix} \nabla\theta(\boldsymbol{x})\right) = 0. \tag{18}$$

Eq. (17) can be rewritten as two first order differential equations (we are dropping the dependencies on $\boldsymbol{x}$ for the ease of reading)

$$\begin{bmatrix} \mathbb{I} & 0 \\ 0 & m \end{bmatrix} \partial_t \begin{bmatrix} \theta \\ \omega \end{bmatrix} = \begin{bmatrix} \omega \\ -d\omega + \nabla(\boldsymbol{B}\nabla\theta) + p \end{bmatrix}, \tag{19}$$

where $\boldsymbol{B} = \mathrm{diag}(b_x, t_y)$.

Using a similar approach as in the previous section, this equation reads in its weak form

$$\begin{bmatrix} \boldsymbol{M}_1 & 0 \\ 0 & \boldsymbol{M}_2 \end{bmatrix} \partial_t \begin{bmatrix} \hat{\boldsymbol{\theta}} \\ \hat{\boldsymbol{\omega}} \end{bmatrix} = \begin{bmatrix} 0 & \boldsymbol{K}_{12} \\ \boldsymbol{K}_{21} & \boldsymbol{K}_{22} \end{bmatrix} \begin{bmatrix} \hat{\boldsymbol{\theta}} \\ \hat{\boldsymbol{\omega}} \end{bmatrix} + \begin{bmatrix} 0 \\ \boldsymbol{P} \end{bmatrix}, \tag{20}$$

where $\hat{\boldsymbol{\theta}}$ and $\hat{\boldsymbol{\omega}}$ are the vectors of the nodal values of θ and ω. The mass matrix is given by

$$\boldsymbol{M}_{1,ij} = \int u_i(\boldsymbol{x}) u_j(\boldsymbol{x}) \mathrm{d}\Omega, \qquad \boldsymbol{M}_{2,ij} = \int m(\boldsymbol{x}) u_i(\boldsymbol{x}) u_j(\boldsymbol{x}) \mathrm{d}\Omega. \tag{21}$$

The stiffness matrices are given by

$$\boldsymbol{K}_{12,ij} = \int u_i(\boldsymbol{x}) u_j(\boldsymbol{x}) \mathrm{d}\Omega, \qquad \boldsymbol{K}_{21,ij} = -\int \nabla u_i(\boldsymbol{x}) (\boldsymbol{B}(\boldsymbol{x}) \nabla u_j(\boldsymbol{x})) \mathrm{d}\Omega, \boldsymbol{K}_{22,ij} = -\int d(\boldsymbol{x}) u_i(\boldsymbol{x}) u_j(\boldsymbol{x}) \mathrm{d}\Omega. \tag{22}$$

Finally, the force vector is given by

$$\boldsymbol{P}_i = \int p(\boldsymbol{x}) u_i(\boldsymbol{x}) \mathrm{d}\Omega. \tag{23}$$

It is important to notice that all matrices are time-independent such we only need to assemble these matrices once and can then use standard tools to solve (20).

III. STABLE SOLUTION AND MACHINE LEARNING

To obtain the stable solution for our problem we need to solve

$$P(\boldsymbol{x}) + \nabla(\boldsymbol{B}\nabla\theta(\boldsymbol{x})) = 0. \tag{24}$$

The weak form reads

$$\boldsymbol{K}_{21}\hat{\boldsymbol{\theta}} = -\boldsymbol{P}. \tag{25}$$

To ensure the well-posedness of the problem we need to enforce the value of $\hat{\boldsymbol{\theta}}$ on one node. We set $\hat{\theta}_i = 0$, where i is closest to the slack bus of the continuous system. This corresponds to setting

$$\boldsymbol{K}_{21,ij} = \boldsymbol{K}_{21,ji} = \delta_{ij}, \qquad \boldsymbol{P}_i = 0. \tag{26}$$

We want to learn the parameters of $\boldsymbol{B}$ from the stable solutions of the system. For this we need to rewrite $\boldsymbol{K}_{12}$ in vectorized form. It turns out that we can write for a single cell

$$\boldsymbol{K}_{\mathrm{cell}} = \boldsymbol{A}_{\mathrm{cell}} \boldsymbol{B}'_{\mathrm{cell}} \boldsymbol{A}_{\mathrm{cell}}^{\top}, \tag{27}$$

where (for three quadrature points per cell $\boldsymbol{x}_i$)

$$\begin{aligned} \boldsymbol{A}_{\mathrm{cell}} = &\begin{bmatrix} \partial_x u_1(\boldsymbol{x}_1) & 0 & \partial_x u_1(\boldsymbol{x}_2) & 0 & \partial_x u_1(\boldsymbol{x}_3) & 0 \\ 0 & \partial_y u_1(\boldsymbol{x}_1) & 0 & \partial_y u_1(\boldsymbol{x}_2) & 0 & \partial_y u_1(\boldsymbol{x}_3) \\ \partial_x u_2(\boldsymbol{x}_1) & 0 & \partial_x u_2(\boldsymbol{x}_2) & 0 & \partial_x u_2(\boldsymbol{x}_3) & 0 \\ 0 & \partial_y u_2(\boldsymbol{x}_1) & 0 & \partial_y u_2(\boldsymbol{x}_2) & 0 & \partial_y u_2(\boldsymbol{x}_3) \\ \partial_x u_3(\boldsymbol{x}_1) & 0 & \partial_x u_3(\boldsymbol{x}_2) & 0 & \partial_x u_3(\boldsymbol{x}_3) & 0 \\ 0 & \partial_y u_3(\boldsymbol{x}_1) & 0 & \partial_y u_3(\boldsymbol{x}_2) & 0 & \partial_y u_3(\boldsymbol{x}_3) \end{bmatrix} \\ &\cdot \mathrm{diag}\left(\sqrt{\det(\omega_1 J(\boldsymbol{x}_1))}, \sqrt{\det(\omega_1 J(\boldsymbol{x}_1))}, \sqrt{\det(\omega_2 J(\boldsymbol{x}_2))}, \right. \\ &\left. \sqrt{\det(\omega_2 J(\boldsymbol{x}_2))}, \sqrt{\det(\omega_3 J(\boldsymbol{x}_3))}, \sqrt{\det(\omega_3 J(\boldsymbol{x}_3))}\right). \end{aligned} \tag{28}$$

Note that this expression can be expanded to higher order quadrature rules. It is important to point out that (6) needs the values of $\boldsymbol{b}$ in the quadrature points. We deal with this the following way: We provide the nodal values of $\boldsymbol{b}$ and project them onto the quadrature points. This can be done in a straight forward fashion using the results of Section I. Using an appropriate projection matrix matrix $\boldsymbol{P}$ we can write

$$\boldsymbol{B}'_{\text{cell}} = \boldsymbol{P}\,\text{diag}\left(\boldsymbol{b}_x(\boldsymbol{x}_{N1}), \boldsymbol{b}_y(\boldsymbol{x}_{N1}), \boldsymbol{b}_x(\boldsymbol{x}_{N2}), \boldsymbol{b}_y(\boldsymbol{x}_{N2}), \boldsymbol{b}_x(\boldsymbol{x}_{N3}), \boldsymbol{b}_y(\boldsymbol{x}_{N3})\right), \tag{29}$$

where $\boldsymbol{x}_{Ni}$ denotes the coordinates of the ith node. Now we need to assemble the global matrix $\boldsymbol{A}$. Each row corresponds to one node in the grid and we simply need to identify the local nodes of the elements with their global index. The columns correspond to all the quadrature points that we can label arbitrarily, we just need to pay attention to correctly assemble the projection matrix. By identifying the global index of the nodes of each cell and adding the values of $\boldsymbol{A}_{\text{cell}}$ to the corresponding entries, $\boldsymbol{A}$ is constructed. To enforce the slack bus using this approach, we have to set the rows corresponding to the slack node to zero. Finally, we add a matrix $\boldsymbol{C}$ that has a single entry on the diagonal at the slack bus node. The stiffness matrix can then be written as

$$\boldsymbol{K}_{21} = \boldsymbol{A}\boldsymbol{P}\boldsymbol{B}'\boldsymbol{A}^\top + \boldsymbol{C}. \tag{30}$$

---

[1] Introduction to FEM · Ferrite.jl.

[2] Finite element spaces - ntnu (2018).

[3] L. Pagnier, J. Fritzsch, P. Jacquod, and M. Chertkov, Toward model reduction for power system transients with physics-informed PDE, IEEE Access **10**, 65118–65125 (2022).