- Consider the following BVP in 2D (or 3D):

$$
\begin{aligned}
-\nabla \cdot \big(\alpha \nabla u(\boldsymbol{r})\big) + \beta u(\boldsymbol{r}) &= f(\boldsymbol{r}), && \boldsymbol{r} \in \Omega, && (82)\\
u|_{\Gamma_D} &= g^D, && \text{Dirichlet} && (83)\\
\alpha \frac{\partial u}{\partial n}\bigg|_{\Gamma_N} &= g^N, && \text{Neumann} && (84)
\end{aligned}
$$

- Here u is a unknown function, α and β are given coefficients, f, g^D and g^N are known functions, and $\Gamma = \Gamma_D \cup \Gamma_N$, $\Gamma_D \cap \Gamma_N = \emptyset$, is the boundary of Ω.

- With Dirichlet boundary condition the weak formulation reads: Find $u \in H^1(\Omega)$, $\gamma_D u = g^D$, so that

$$\langle \nabla w \, , \, \alpha \nabla u \rangle + \langle w \, , \, \beta u \rangle = \langle w \, , \, f \rangle \, , \tag{85}$$

  holds for all $w \in H^1(\Omega)$, $\gamma_D w = 0$.
- With Neumann boundary condition the weak formulation reads: Find $u \in H^1(\Omega)$ so that

$$\langle \nabla w \, , \, \alpha \nabla u \rangle + \langle w \, , \, \beta u \rangle = - < w \, , g^N >_{\Gamma_N} + \langle w \, , \, f \rangle \, , \tag{86}$$

  holds for all $w \in H^1(\Omega)$.

GENERAL RECIPE

Weak Formulation

- Weak formulation can be obtained as follows. Multiply equation

$$-\nabla \cdot \big(\alpha \nabla u(\boldsymbol{r})\big) + \beta u(\boldsymbol{r}) = f(\boldsymbol{r}), \tag{87}$$

with a testing function $w \in H^1(\Omega)$ via the L^2 symmetric product

$$-\langle w\,,\, \nabla \cdot (\alpha \varepsilon(\boldsymbol{r}) \nabla u \rangle + \langle w\,,\, \beta \nabla u \rangle = \langle w\,,\, f \rangle\,. \tag{88}$$

- Use identity

$$\nabla \cdot (u\boldsymbol{F}) = \nabla u \cdot \boldsymbol{F} + u \nabla \cdot \boldsymbol{F}, \tag{89}$$

with $\boldsymbol{F} = \alpha \nabla u$

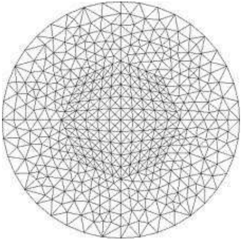
$$-\langle w\,,\, \nabla \cdot (\alpha \nabla u) \rangle = \langle \nabla w\,,\, \alpha \nabla u \rangle + \int_\Omega \nabla \cdot (w\, \alpha \nabla u)\, d\Omega. \tag{90}$$

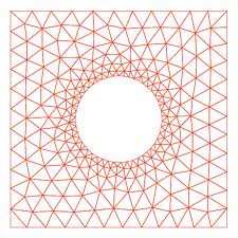
- Then Gauss divergence theorem and boundary conditions give

$$\int_\Omega \nabla \cdot (w\, \alpha \nabla u)\, d\Omega = \langle w\,,\, \alpha\, \boldsymbol{n} \cdot \nabla u \rangle_{\Gamma_N} = \left\langle w\,,\, g^N \right\rangle_{\Gamma_N}. \tag{91}$$

- Wanted weak formulation is obtained by combining above results.

- A significant obvious difference compared to 1D is the mesh, i.e., geometrical element. In 2D we use triangles and in 3D tetras.





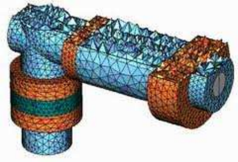


Figure: Triangle (2D) and tetra (3D) meshes.

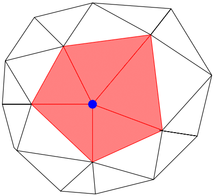
- FE space (T, P_T, Σ_T) consists of:
  - Geometrical element T, a triangle (2D) or a tetra (3D).
  - First order polynomial approximation on T.
  - dof are the values of the approximation of u at the nodes of T.

- Use piece-wise linear continuous basis functions u_n

$$u(\boldsymbol{r}) \approx u^h(\boldsymbol{r}) = \sum_{n=1}^{N_N} c_n u_n(\boldsymbol{r}), \tag{92}$$

defined as (N_n and N_m denote nodes of the mesh)

$$u_n(\boldsymbol{r}) = \begin{cases} 1 & \text{if } \boldsymbol{r} = N_n, \\ 0 & \text{if } \boldsymbol{r} = N_m,\ m \neq n, \\ \text{linear} & \text{otherwise.} \end{cases} \tag{93}$$



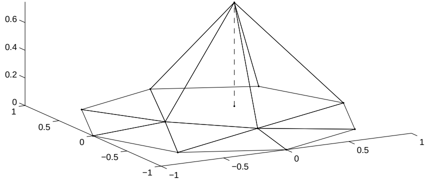


Figure: Support (left) and magnitude (right) of a linear nodal basis function.

- This approximation is unisolvent and H^1 conforming.
- It defines a linear interpolation on each element.
- The total number of dof is the number of nodes of the mesh. (Note: Dirichlet boundary data fixes the values on the boundary).
- The number of elements (triangles) associated to a node depends on the mesh.
- Extension to 3D is straightforward.

2D and 3D FINITE ELEMENT METHOD

Matrix Equation and Local Matrices

- Using Galerkin's method we obtain a matrix equation

$$\boldsymbol{A}\,\boldsymbol{x} = \boldsymbol{b}, \tag{94}$$

with elements (without boundary conditions)

$$A_{mn} = \langle \nabla u_m \,,\, \alpha \nabla u_n \rangle + \langle u_m \,,\, \beta u_n \rangle \,, \tag{95}$$

$$b_m = \langle u_m \,,\, f \rangle \,. \tag{96}$$

- Similarly as in 1D, define local matrices and local vector

$$\texttt{alok1}(i,j) = \int_{T_k} N_i^k(\boldsymbol{r})\, N_j^k(\boldsymbol{r})\, d\boldsymbol{r}, \tag{97}$$

$$\texttt{alok2}(i,j) = \int_{T_k} \nabla N_i^k \cdot \nabla N_j^k \, d\boldsymbol{r}, \tag{98}$$

$$\texttt{blok1}(i) = \int_{T_k} N_i^k(\boldsymbol{r})\, f(\boldsymbol{r})\, d\boldsymbol{r}, \tag{99}$$

$i,j = 1,\ldots,3$ (2D), $i,j = 1,\ldots,4$ (3D). Here $N_i^k = u_n|_{T_k}$.

```
for k = 1, ..., number of elements do
   % Compute local matrices alok1 and alok2
   for i = 1, ..., R+1 do
      for j = 1, ..., R+1 do
         alok1(i,j) ← ∫_{T_k} N_i^k(r) N_j^k(r) dr
         alok2(i,j) ← ∫_{T_k} ∇N_i^k(r) · ∇N_j^k(r) dr
      end for
   end for
   % Add local matrices to the global one
   for i = 1, ..., R+1 do
      for j = 1, ..., R+1 do
         A(n_i^k, n_j^k) ← A(n_i^k, n_j^k) + α_k alok2(i,j) + β_k alok1(i,j)
      end for
   end for
end for
```

- Here $R = 2$ (2D) or 3 (3D), α_k and β_k are constants in T_k.

for $k = 1, \ldots,$ number of elements **do**
  Compute local vector `blok1`
  for $i = 1, \ldots, R+1$ **do**
    $\texttt{blok1}(i) \leftarrow \int_{T_k} N_i^k(\boldsymbol{r}) f(\boldsymbol{r})\, d\boldsymbol{r}$
  end for
  Add local vector to the global one
  for $i = 1, \ldots, R+1$ **do**
    $\boldsymbol{b}(n_i^k) \leftarrow \boldsymbol{b}(n_i^k) + \texttt{blok1}(i)$
  end for
end for

- Analogously to 1D, indeces n_i^k and n_j^k are given in 2D and 3D by

$$n_i^k = \texttt{etopol}(i,k) \quad \text{and} \quad n_j^k = \texttt{etopol}(j,k). \tag{100}$$

- The algorithms, and also how the boundary conditions are enforced are identical with the 1D case. What changes is the numerical evaluation of the matrix elements.

- A 2D mesh can be described with nodes, edges and elements. Vertices of the elements (triangles) are called nodes.
- A triangular mesh can be defined using the following data structures

$$\begin{aligned} \texttt{coord} &: (2 \times N_N) \text{ matrix; } x \text{ and } y \text{ coordinates of the nodes,} \\ \texttt{etopol} &: (3 \times N_T) \text{ matrix; indeces of the nodes of the elements.} \end{aligned}$$

- Here N_N is the number of the nodes and N_T is the number of the elements.
- The x and y coordinates of the vertices of an element k are (Matlab notations):

```
p1 = coord(:,etopol(1,k));
p2 = coord(:,etopol(2,k));
p3 = coord(:,etopol(3,k));
```

- A tetra mesh in 3D can be defined using the following data structures

$$\begin{aligned}\texttt{coord} &: (3 \times N_N) \text{ matrix; } x, y \text{ and } z \text{ coordinates of the nodes,} \\ \texttt{etopol} &: (4 \times N_T) \text{ matrix; indeces of the nodes of the elements.}\end{aligned}$$

- Here N_N is the number of the nodes and N_T is the number of the elements.
- The x, y and z coordinates of the vertices of an element k are (Matlab notations):

```
p1 = coord(:,etopol(1,k));
p2 = coord(:,etopol(2,k));
p3 = coord(:,etopol(3,k));
p4 = coord(:,etopol(4,k));
```

- Let $\boldsymbol{p}_j^k$, $j = 1, 2, 3$, denote the vertices of a triangle T_k.
- The vertices of a reference triangle $\hat{T}$ are $(0, 0)$, $(1, 0)$ and $(0, 1)$.
- Define a linear mapping $\mathcal{F}_k : \hat{T} \mapsto T_k$.
- Function $\mathcal{F}_k$ maps point $(0, 0)$ to $\boldsymbol{p}_1^k$, point $(1, 0)$ to $\boldsymbol{p}_2^k$ and point $(0, 1)$ to $\boldsymbol{p}_3^k$.

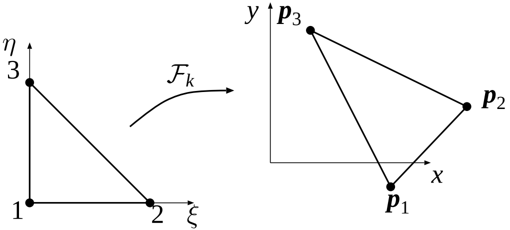


Figure: Linear mapping $\mathcal{F}_k$ from a reference triangle $\hat{T}$ to a triangle T_k.

- Mapping $\mathcal{F}_k$ can be defined as

$$\mathcal{F}_k(\xi,\eta) := \sum_{i=1}^{3} \boldsymbol{p}_i^k \hat{N}_i(\xi,\eta) = (\boldsymbol{p}_2^k - \boldsymbol{p}_1^k)\xi + (\boldsymbol{p}_3^k - \boldsymbol{p}_1^k)\eta + \boldsymbol{p}_1^k, \qquad (101)$$

where $\hat{N}_i,\ i = 1,2,3,$ are the nodal shape functions on $\hat{T}$

$$\hat{N}_1(\xi,\eta) = 1 - \xi - \eta, \qquad (102)$$

$$\hat{N}_2(\xi,\eta) = \xi, \qquad (103)$$

$$\hat{N}_3(\xi,\eta) = \eta. \qquad (104)$$

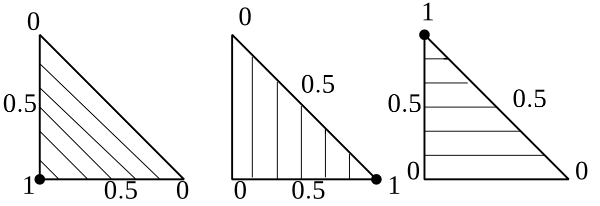


Figure: Linear nodal shape functions on a reference triangle.

- Using mapping $\mathcal{F}_k$ we can write

$$\int_{T_k} u(x,y)\,dx\,dy = \int_{\hat{T}} u(\mathcal{F}_k(\xi,\eta))\,|\det(J_{\mathcal{F}_k})|\,d\xi\,d\eta \tag{105}$$

where $J_{\mathcal{F}_k}$ is the Jacobian of $\mathcal{F}_k$, given as

$$J_{\mathcal{F}_k} = \left[\frac{\partial \mathcal{F}_k}{\partial_\xi}, \frac{\partial \mathcal{F}_k}{\partial_\eta}\right] = \left[\boldsymbol{p}_2^k - \boldsymbol{p}_1^k, \boldsymbol{p}_3^k - \boldsymbol{p}_1^k\right]. \tag{106}$$

- With this formula the integration on T_k is reduced to integration on $\hat{T}$ and can be evaluated numerically using 2D integration points and weights, ξ_p, η_p and ω_p, defined on the reference element $\hat{T}$

$$\int_{T_k} u(x,y)\,dx\,dy \approx |\det(J_{\mathcal{F}_k})| \sum_{p=1}^{P} \omega_p u(\mathcal{F}_k(\xi_p,\eta_p)). \tag{107}$$

- Note $|\det(J_{\mathcal{F}_k})|$ is two times area of the triangle T_k.

- Shape functions on T can be defined using mapping $\mathcal{F}_k$ as

$$N_i^k := \hat{N}_i(\mathcal{F}^{-1}(\boldsymbol{r})), \tag{108}$$

i.e.,

$$N_i^k(\boldsymbol{r}) = \hat{N}_i(\mathcal{F}_k^{-1}(\boldsymbol{r})) = \hat{N}_i(\hat{\boldsymbol{r}}), \tag{109}$$

where $\mathcal{F}_k^{-1} : T_k \mapsto \hat{T}$ is the inverse of $\mathcal{F}_k$, $\boldsymbol{r} = (x, y)$ is a point in T_k and $\hat{\boldsymbol{r}} = (\xi, \eta)$ a point in $\hat{T}_k$.

- Thus, we have

$$\begin{aligned} \texttt{alok1}(i,j) &= \int_{T_k} N_i^k(\boldsymbol{r})\, N_j^k(\boldsymbol{r})\, d\boldsymbol{r} = |\det(J_{\mathcal{F}_k})| \int_{\hat{T}} \hat{N}_i(\hat{\boldsymbol{r}})\, \hat{N}_j(\hat{\boldsymbol{r}})\, d\hat{\boldsymbol{r}}, & (110)\\ \texttt{blok1}(i) &= \int_{T_k} N_i^k(\boldsymbol{r})\, f(\boldsymbol{r})\, d\boldsymbol{r} = |\det(J_{\mathcal{F}_k})| \int_{\hat{T}} \hat{N}_i(\hat{\boldsymbol{r}})\, f(\mathcal{F}_k(\hat{\boldsymbol{r}}))\, d\hat{\boldsymbol{r}}. & (111) \end{aligned}$$

- The gradients of the nodal shape functions are more complicated.
- Using the chain rule we get (in 2D)

$$\frac{\partial \hat{N}_i(\hat{\boldsymbol{r}})}{\partial \xi} = \frac{\partial N_i^k(\boldsymbol{r})}{\partial x}\frac{\partial \mathcal{F}_x}{\partial \xi} + \frac{\partial N_i^k(\boldsymbol{r})}{\partial y}\frac{\partial \mathcal{F}_y}{\partial \xi}, \tag{112}$$

$$\frac{\partial \hat{N}_i(\hat{\boldsymbol{r}})}{\partial \eta} = \frac{\partial N_i^k(\boldsymbol{r})}{\partial x}\frac{\partial \mathcal{F}_x}{\partial \eta} + \frac{\partial N_i^k(\boldsymbol{r})}{\partial y}\frac{\partial \mathcal{F}_y}{\partial \eta}. \tag{113}$$

- This can be expressed shortly as

$$\hat{\nabla}\hat{N}_i(\hat{\boldsymbol{r}}) = J_{\mathcal{F}_k}^T \nabla N_i^k(\boldsymbol{r}), \tag{114}$$

where $J_{\mathcal{F}_k}^T$ is the transpose of the Jacobian matrix of mapping $\mathcal{F}_k$.

- The Jacobian matrix is given as

$$J_{\mathcal{F}_k}(\hat{\boldsymbol{r}}) = \begin{bmatrix} \dfrac{\partial F_x}{\partial \xi} & , & \dfrac{\partial F_x}{\partial \eta} \\ \dfrac{\partial F_y}{\partial \xi} & , & \dfrac{\partial F_y}{\partial \eta} \end{bmatrix} = \left[\boldsymbol{p}_2^k - \boldsymbol{p}_1^k \ , \ \boldsymbol{p}_3^k - \boldsymbol{p}_1^k\right] . \qquad (115)$$

- Computing the inverse of $J_{\mathcal{F}_k}^T$ we get

$$\nabla N_i^k(\boldsymbol{r}) = (J_{\mathcal{F}_k}^T)^{-1}\hat{\nabla}\hat{N}_i(\hat{\boldsymbol{r}}) \qquad (116)$$

- Matrix elements with gradients of the shape functions are

$$\begin{aligned} \texttt{alok2}(i,j) &= \int_{T_k} \nabla N_i^k \cdot \nabla N_j^k \, d\boldsymbol{r} \\ &= \left|\det(J_{\mathcal{F}_k})\right| \int_{\hat{T}} \left((J_{\mathcal{F}_k}^T)^{-1}\,\hat{\nabla}\hat{N}_i \cdot (J_{\mathcal{F}_k}^T)^{-1}\,\hat{\nabla}\hat{N}_j \right) d\hat{\boldsymbol{r}}. \qquad (117) \end{aligned}$$

- 3D is more or less a straightforward extension (3 coordinates, 4 shape functions per tetra, etc.).

NUMERICAL EXAMPLE

Capacitance Calculation in Electrostatics

- Maxwell's equations for the static electric field $\boldsymbol{E}$ read

$$\begin{aligned} \nabla \times \boldsymbol{E}(\boldsymbol{r}) &= 0, && (118) \\ \nabla \cdot (\varepsilon(\boldsymbol{r})\boldsymbol{E}(\boldsymbol{r})) &= \rho_s(\boldsymbol{r}), && (119) \end{aligned}$$

  where ρ_s is the charge density and ε is the electric permitttivity.
- Since $\nabla \times \nabla u = 0$ for an arbitrary sufficiently differentiable function u, static electric field $\boldsymbol{E}$ can be expressed using a scalar potential ϕ

$$\boldsymbol{E}(\boldsymbol{r}) = -\nabla\phi(\boldsymbol{r}). \tag{120}$$

- Using Maxwell's equation $\nabla \cdot (\varepsilon\boldsymbol{E}) = \rho_s$ potential ϕ satisfies

$$\nabla \cdot (\varepsilon(\boldsymbol{r})\nabla\phi(\boldsymbol{r})) = -\rho_s(\boldsymbol{r}). \tag{121}$$

- If ε is constant, equation (121) reduces to the Laplace equation.

$$\Delta\phi(\boldsymbol{r}) = -\frac{\rho_s(\boldsymbol{r})}{\varepsilon}, \tag{122}$$

- Thus, in electrostatics we may consider (generalized) Laplace (Poisson) equation for a scalar function.

Consider a homogeneous PE filled ($\varepsilon = 2.3\varepsilon_0$) circular RG-58/U coaxial cable with inner radius $a = 0.81$mm and outer radius $b = 2.9$mm. Assume that the voltage on the inner conductor is +5V and on the outer conductor the voltage is 0V. Since the structure is uniform and homogeneous it is sufficient to find the scalar potential on a 2D cross section of the cable by solving the Dirichlet boundary value problem for Laplace equation

$$
\begin{aligned}
2.3\varepsilon_0\nabla^2\phi(\boldsymbol{r}) &= 0, \quad \boldsymbol{r} \in \Omega, && (123)\\
\phi(\boldsymbol{r}) &= 5, \quad \boldsymbol{r} \in \Gamma_a, && (124)\\
\phi(\boldsymbol{r}) &= 0, \quad \boldsymbol{r} \in \Gamma_b. && (125)
\end{aligned}
$$

Here Ω is the 2D cross section of the medium between the inner and outer conductors, Γ_a is the boundary of the inner conductor, Γ_b is the boundary of the outer conductor, and ε_0 is a known constant.

NUMERICAL EXAMPLE

Capacitance Calculation in Electrostatics

Find the solution using (2D) FEM and piece-wise linear nodal functions as described before.

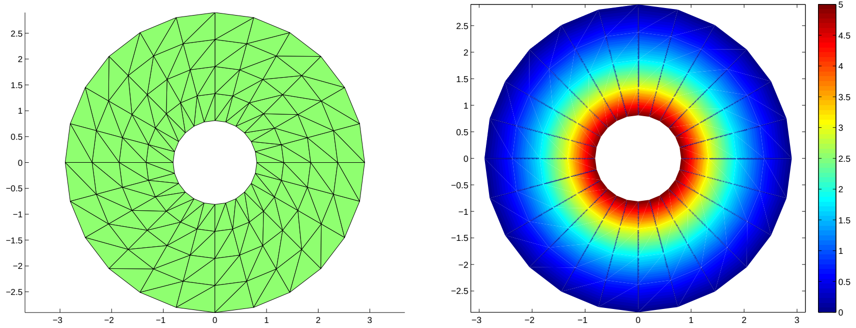


Figure: Triangular mesh on the 2D cross section of a coaxial cable (left) and the solution, i.e., the electrostatic potential (right).

NUMERICAL EXAMPLE

Capacitance Calculation in Electrostatics

Capacitance C tells how much a structure can store an electrical charge

$$C = \frac{Q}{U}, \tag{126}$$

where U is the potential difference between the conductors, and Q is a net charge. This can be expressed using electrostatic energy W stored in the structure

$$C = \frac{2}{U^2} W, \tag{127}$$

where the energy is given by ($\boldsymbol{D} = \varepsilon \boldsymbol{E}$)

$$W = \frac{1}{2} \int_\Omega \boldsymbol{E}(\boldsymbol{r}) \cdot \boldsymbol{D}(\boldsymbol{r})\, d\boldsymbol{r}. \tag{128}$$

Using electrostatic potential ϕ, energy can be further written as

$$W = \frac{1}{2} \int_\Omega \varepsilon(\boldsymbol{r}) \nabla \phi(\boldsymbol{r}) \cdot \nabla \phi(\boldsymbol{r})\, d\boldsymbol{r}. \tag{129}$$

Thus, capacitance can be expressed as

$$C = \frac{1}{U^2} \int_{\Omega} \varepsilon(\boldsymbol{r}) \nabla \phi(\boldsymbol{r}) \cdot \nabla \phi(\boldsymbol{r}) \, d\boldsymbol{r}. \tag{130}$$

This can be calculated by using the electrostatic FEM system matrix, i.e., the matrix with elements (without boundary conditions)

$$A_{m,n} = \int_{\mathrm{spt}(u_m) \cap \mathrm{spt}(u_n)} \varepsilon(\boldsymbol{r}) \nabla u_m(\boldsymbol{r}) \cdot \nabla u_n(\boldsymbol{r}) \, d\boldsymbol{r}. \tag{131}$$

On circular domains, both the potential and capacitance have analytical solutions

$$\phi_{\mathrm{ana}}(r) = \frac{\phi_b - \phi_a}{\log(b/a)} \log(r/a) + \phi_a, \tag{132}$$

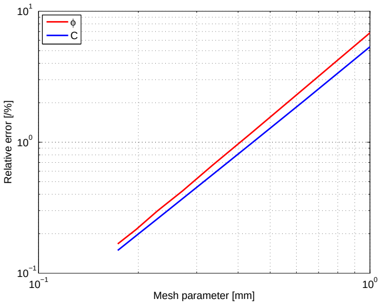
$$C_{\mathrm{ana}} = 2\pi\varepsilon / \log(b/a). \tag{133}$$

NUMERICAL EXAMPLE

Capacitance Calculation in Electrostatics

Compute Relative Root Mean Square (RMS) error of the numerical solution

$$\frac{\left\|\phi_{\text{num}} - \phi_{\text{ana}}\right\|}{\left\|\phi_{\text{ana}}\right\|} := \frac{\sqrt{\sum_{m=1}^{M} |\phi_{\text{num}}(\boldsymbol{r}) - \phi_{\text{ana}}(\boldsymbol{r})|^2/M}}{\sqrt{\sum_{m=1}^{M} |\phi_{\text{ana}}(\boldsymbol{r})|^2/M}}. \qquad (134)$$



Solutions of both ϕ and C seem to converge with rate $O(h^2)$. General rule is that FEM approximations of "smooth" functions converge with rate $O(h^{2p})$, where p is the order of the polynomial approximation. Higher order approximations usually give better accuracy with less number of dof, but implementations become (much) more complicated.

SUMMARY

FEM for finding approximate solutions of boundary value problems arising from partial differential equation-based mathematical modeling of physical phenomena can be summarized as:

1. Develop a BVP for a PDE with information of the domain and boundary.
2. Derive weak formulation of the BVP with appropriate boundary conditions.
3. Generate the mesh, i.e., divide domain Ω into a finite number of simple elements.
4. Find appropriate conforming and unisolvent discrete FE spaces.
5. Compute the matrix elements and assemble the matrix.
6. Solve the matrix equation.
7. Compute wanted parameters.