Chapter 2
The Finite Element (FE) Method

The finite element (FE) method has become the standard numerical calculation scheme for the computer simulation of physical systems [1–3]. The advantages of this method can be summarized as follows:

- **Numerical efficiency**: The discretization of the calculation domain with finite elements yields matrices that are in most cases sparse and symmetric. Therefore, the system matrix, which is obtained after spatial and time discretization, is sparse and symmetric, too. Both the storage of the system matrix and the solution of the algebraic system of equations can be performed in a very efficient way.
- **Treatment of nonlinearities**: The modeling of nonlinear material behavior is well established for the FE method (e.g., nonlinear curves, hysteresis).
- **Complex geometry**: By the use of the FE method, any complex domain can be discretized by triangular elements in 2D and by tetrahedra in 3D.
- **Analysis possibilities**: The FE method is suited for static, transient, harmonic as well as eigenfrequency analysis.

The two essential disadvantages of the FE method are given by

- **Discretization**: The effort for the discretization of the simulation domain is quite high, since in 2D the whole cross section and in 3D the whole volume has to be subdivided into finite elements.
- **Open domain problems**: Models that need the treatment of an open boundary, e.g., the simulation of radiation characteristics of an ultrasound array, lead in the general case to errors due to the limitation of the simulation domain. One of several approaches to overcome this problem is the use of absorbing boundary conditions, perfectly matched layer (PML) techniques or so-called *infinite elements* (see Sect. 5.5).

The general approach of the FE method is shown in Fig. 2.1. Starting from the partial differential equation (PDE) with given boundary conditions, we multiply it by appropriate test functions and integrate over the whole simulation domain. Performing an integration by parts, we arrive at the variational formulation, also

© Springer-Verlag Berlin Heidelberg 2015

M. Kaltenbacher, *Numerical Simulation of Mechatronic Sensors and Actuators*,
DOI 10.1007/978-3-642-40170-1_2

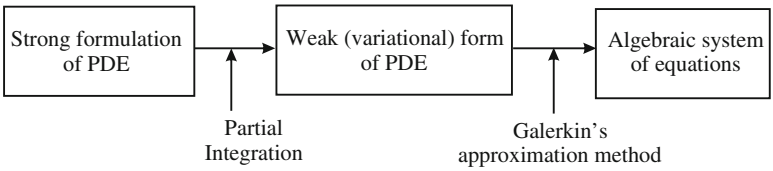
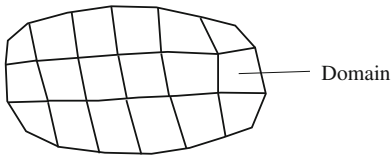


Fig. 2.1 From the strong formulation to the algebraic system of equation

Fig. 2.2 FE method: discretization of the domain with quadrilateral finite elements



Finite element

1–4 Finite element nodes

called the weak formulation. Applying Galerkin's approximation method using finite elements (FE) results in the algebraic system of equations.

As already mentioned, the use of the FE method requires the discretization of the whole domain (see Fig. 2.2). For the discretization triangular as well as quadrilateral finite elements are used in 2D and tetrahedral as well as hexahedron finite elements in 3D. The physical quantity of interest (e.g., temperature, mechanical displacement, etc.) is approximated by so-called shape functions and the solution of the algebraic equation yields the physical quantity in the discretization points, the so-called finite element nodes, for Lagrangian finite elements and along the edges for Nédélec finite elements.

2.1 Finite Element Formulation

In the following, all steps—from the strong formulation of the partial differential equation (PDE) to the algebraic equation—will be briefly described by means of the following simple PDE with the searched for quantity $u(\boldsymbol{r}, t)$, the known source term $f(\boldsymbol{r}, t)$ at each time t of the interval $(0, T)$, and the corresponding initial and boundary conditions.

$$\begin{aligned} \text{Given: } f &: \Omega \times (0, T) \to \mathbb{R} \\ u_0 &: \Omega \to \mathbb{R} \end{aligned}$$

$$\text{Find: } \quad u : \bar{\Omega} \times [0, T] \to \mathbb{R}$$

$$\begin{aligned} \frac{\partial u}{\partial t} &= \nabla \cdot \nabla u + f \\ u &= u_{\mathrm{e}} \text{ on } \Gamma_{\mathrm{e}} \ \times \ (0, T) \\ \frac{\partial u}{\partial \boldsymbol{n}} &= u_{\mathrm{n}} \text{ on } \Gamma_{\mathrm{n}} \ \times \ (0, T) \\ u(\boldsymbol{r}, 0) &= u_0 \,, \ \boldsymbol{r} \in \Omega \,. \end{aligned} \tag{2.1}$$

In this so-called *strong formulation* of the initial-boundary value problem $\mathbb{R}$ denotes the set of real numbers, $\bar{\Omega}$ the simulation domain, Ω the simulation domain without the boundary $\Gamma = \Gamma_{\mathrm{e}} \cup \Gamma_{\mathrm{n}}$, Γ_{e} the boundary with prescribed Dirichlet boundary condition, and Γ_{n} the boundary with prescribed Neumann boundary condition. Now, let us introduce for any $t \in [0, T]$ the space $\mathbf{T}_t$

$$\mathbf{T}_{\mathrm{t}} = \{u(\cdot, t) \mid u(\cdot, t) \in H^1(\Omega), u(\boldsymbol{r}, t) = u_{\mathrm{e}}(\boldsymbol{r}, t) \text{ on } \Gamma_{\mathrm{e}}\} \,, \tag{2.2}$$

and $\mathbf{G}$, the space of so-called test functions, as

$$\mathbf{G} = \{w \mid w \in H^1(\Omega), w = 0 \text{ on } \Gamma_{\mathrm{e}}\}, \tag{2.3}$$

with H^1 the standard Sobolev space (see Appendix D). It has to be noted that the spaces $\mathbf{T}_t$, $t \in [0, T]$ vary with time, whereas the space $\mathbf{G}$ is time-independent.

In the first step, we multiply the partial differential equation with an arbitrary *test function* w and perform an integration over the whole domain Ω

$$\int_{\Omega} w \left(\frac{\partial u}{\partial t} - \nabla \cdot \nabla u - f \right) \, \mathrm{d}\Omega = 0.$$

Applying Green's first integration theorem to the above equation results in

$$\int_{\Omega} w \frac{\partial u}{\partial t} \, \mathrm{d}\Omega + \int_{\Omega} \nabla w \cdot \nabla u \, \mathrm{d}\Omega = \int_{\Omega} w f \, \mathrm{d}\Omega + \int_{\Gamma_{\mathrm{n}}} w \frac{\partial u}{\partial \boldsymbol{n}} \, \mathrm{d}\Gamma. \tag{2.4}$$

Thus, the *weak formulation* (often also called variational formulation) for the initial-boundary problem is as follows:

Given: $f \quad : \Omega \times (0, T) \to \mathbb{R}$
$u_0 \quad : \Omega \to \mathbb{R}$

Find: $u(t) \in \mathbf{T}_t$ such that for all $w \in \mathbf{G}$ and $t \in [0, T]$

$$\int_\Omega w\frac{\partial u}{\partial t}\,\mathrm{d}\Omega + \int_\Omega \nabla w\cdot\nabla u\,\mathrm{d}\Omega = \int_\Omega wf\,\mathrm{d}\Omega + \int_{\Gamma_\mathrm{n}} wu_\mathrm{n}\,\mathrm{d}\Gamma \qquad (2.5)$$
$$u = u_\mathrm{e} \text{ on } \Gamma_\mathrm{e} \times (0,T)$$
$$\int_\Omega wu(0)\,\mathrm{d}\Omega = \int_\Omega wu_0\,\mathrm{d}\Omega.$$

Since the Neumann boundary condition is now incorporated into the equation of the weak form, it is also called *natural*. The Dirichlet boundary condition on u still has to be explicitly forced, and is therefore called *essential*. Formally it can be proven that the two formulations according to (2.1) and (2.5) are mathematically equivalent, provided u is sufficiently smooth [4].

To discretize (2.5), which is still infinite dimensional, we now apply the approximation according to *Galerkin's* method. Let us define the finite dimensional spaces $\mathbf{T}_\mathrm{t}^h$ and $\mathbf{G}^h$ according to

$$\mathbf{T}_\mathrm{t}^h \subset \mathbf{T} \quad \mathbf{G}^h \subset \mathbf{G}.$$

Therefore, we perform the domain discretization (see Fig. 2.2), and approximate the searched for quantity $u(t)$ as well as the test function w by

$$u(t) \approx u^h(t) \qquad w \approx w^h, \qquad (2.6)$$

with h the discretization parameter (defining the mesh size). Furthermore, we decompose $u^h(t)$ into the searched for value $v^h(t)$ and the known Dirichlet values $u_\mathrm{e}^h(t)$. For v^h, w^h, and u_e^h we choose the following ansatz

$$v^h(t) = \sum_{a=1}^{n_\mathrm{eq}} N_a(\boldsymbol{r})v_a(t) \qquad (2.7)$$

$$w^h = \sum_{a=1}^{n_\mathrm{eq}} N_a(\boldsymbol{r})c_a \qquad (2.8)$$

$$u_\mathrm{e}^h(t) = \sum_{a=1}^{n_\mathrm{e}} N_a(\boldsymbol{r})u_{\mathrm{e}a}(t), \qquad (2.9)$$

where $N_a(\boldsymbol{r})$ denotes appropriate shape functions (often also called interpolation or basis functions), n_eq the number of unknowns, which is equal to the number of finite element nodes with no Dirichlet boundary condition, and n_e the number of finite element nodes with Dirichlet boundary condition. Substituting (2.7)–(2.9) into (2.5) results in

$$\int_\Omega \left(\sum_{a=1}^{n_{\mathrm{eq}}} N_a c_a \frac{\partial}{\partial t} \sum_{b=1}^{n_{\mathrm{eq}}} N_b v_b\right) \mathrm{d}\Omega + \int_\Omega \nabla\left(\sum_{a=1}^{n_{\mathrm{eq}}} N_a c_a\right) \cdot \nabla\left(\sum_{b=1}^{n_{\mathrm{eq}}} N_b v_b\right) \mathrm{d}\Omega$$
$$+ \int_\Omega \left(\sum_{a=1}^{n_{\mathrm{eq}}} N_a c_a \frac{\partial}{\partial t} \sum_{b=1}^{n_{\mathrm{e}}} N_b u_{eb}\right) \mathrm{d}\Omega + \int_\Omega \nabla\left(\sum_{a=1}^{n_{\mathrm{eq}}} N_a c_a\right) \cdot \left(\nabla \sum_{b=1}^{n_{\mathrm{e}}} N_b u_{eb}\right) \mathrm{d}\Omega$$
$$= \int_\Omega \sum_{a=1}^{n_{\mathrm{eq}}} N_a c_a f(\boldsymbol{r}_a)\ \mathrm{d}\Omega + \int_{\Gamma_{\mathrm{n}}} \sum_{a=1}^{n_{\mathrm{eq}}} N_a c_a u_{\mathrm{n}}(\boldsymbol{r}_a)\ \mathrm{d}\Gamma. \tag{2.10}$$

Now, since we can put the sums before the integrals and having in mind that ∇ just operates on the shape functions $N(\boldsymbol{r})$ (c_a as well as u_b are constants with respect to the space variables), we may write (2.10) for the 2D plane case as follows

$$\sum_{a=1}^{n_{\mathrm{eq}}} c_a \Bigg\{ \quad \sum_{b=1}^{n_{\mathrm{eq}}} \Bigg(\left[\int_\Omega N_a N_b \,\mathrm{d}\Omega\right] \frac{\partial v_b}{\partial t}$$
$$+ \left[\int_\Omega \left(\frac{\partial N_a}{\partial x}\frac{\partial N_b}{\partial x} + \frac{\partial N_a}{\partial y}\frac{\partial N_b}{\partial y}\right) \mathrm{d}\Omega\right] v_b \Bigg)$$
$$- \int_\Omega N_a f \,\mathrm{d}\Omega - \int_{\Gamma_{\mathrm{n}}} N_a u_{\mathrm{n}} \,\mathrm{d}\Gamma$$
$$+ \sum_{b=1}^{n_{\mathrm{e}}} \left(\int_\Omega \left(\frac{\partial N_a}{\partial x}\frac{\partial N_b}{\partial x} + \frac{\partial N_a}{\partial y}\frac{\partial N_b}{\partial y}\right) \mathrm{d}\Gamma\right) u_{eb}$$
$$+ \sum_{b=1}^{n_{\mathrm{e}}} \left(\int_\Omega N_a N_b \,\mathrm{d}\Omega\right) \frac{\partial u_{eb}}{\partial t} \Bigg\} = 0.$$

Since the equation has to be fulfilled for all coefficients c_a, we obtain the defining equations for the searched for finite element node values v_b: for each a ($a = 1, .., n_{\mathrm{eq}}$) we have to solve an equation as follows

$$\sum_{b=1}^{n_{\mathrm{eq}}} \left(\left[\int_\Omega N_a N_b \,\mathrm{d}\Omega\right] \frac{\partial v_b}{\partial t} + \left[\int_\Omega \left(\frac{\partial N_a}{\partial x}\frac{\partial N_b}{\partial x} + \frac{\partial N_a}{\partial y}\frac{\partial N_b}{\partial y}\right) \mathrm{d}\Omega\right] v_b \right)$$
$$- \int_\Omega N_a f \,\mathrm{d}\Omega - \int_{\Gamma_{\mathrm{n}}} N_a u_{\mathrm{n}} \,\mathrm{d}\Gamma$$
$$+ \sum_{b=1}^{n_{\mathrm{e}}} \left[\int_\Omega \left(\frac{\partial N_a}{\partial x}\frac{\partial N_b}{\partial x} + \frac{\partial N_a}{\partial y}\frac{\partial N_b}{\partial y}\right) \mathrm{d}\Omega\right] u_{eb}$$
$$+ \sum_{b=1}^{n_{\mathrm{e}}} \left[\int_\Omega N_a N_b \,\mathrm{d}\Omega\right] \frac{\partial u_{eb}}{\partial t} = 0. \tag{2.11}$$

Thus, the *semi-discrete Galerkin formulation* can be written in matrix form as follows

$$\mathbf{M}\underline{\dot{v}} + \mathbf{K}\underline{v} = \underline{f}, \tag{2.12}$$

with $\underline{\dot{v}} = \partial \underline{v}/\partial t$, $\underline{v}$ the nodal unknowns and $\underline{f}$ the right-hand side vector.

- Mass matrix $\mathbf{M}$:

$$\begin{aligned} \mathbf{M} &= [M_{ab}] \\ M_{ab} &= \int_{\Omega} N_a N_b \, \mathrm{d}\Omega \\ 1 &\leq a, b \leq n_{\mathrm{eq}} \end{aligned} \tag{2.13}$$

- Stiffness Matrix $\mathbf{K}$:

$$\begin{aligned} \mathbf{K} &= [K_{ab}] \\ K_{ab} &= \int_{\Omega} \left(\frac{\partial N_a}{\partial x}\frac{\partial N_b}{\partial x} + \frac{\partial N_a}{\partial y}\frac{\partial N_b}{\partial y} \right) \mathrm{d}\Omega \\ 1 &\leq a, b \leq n_{\mathrm{eq}} \end{aligned} \tag{2.14}$$

- Right-hand side $\underline{f}$:

$$\begin{aligned} \underline{f} &= [f_a] \\ f_a &= \int_{\Omega} N_a f \, \mathrm{d}\Omega + \int_{\Gamma_{\mathrm{n}}} N_a u_{\mathrm{n}} \, \mathrm{d}\Gamma \\ &\quad - \sum_{b=1}^{n_{\mathrm{e}}} \left[\int_{\Omega} \left(\frac{\partial N_a}{\partial x}\frac{\partial N_b}{\partial x} + \frac{\partial N_a}{\partial y}\frac{\partial N_b}{\partial y} \right) \mathrm{d}\Omega \right] u_{\mathrm{e}b} \\ &\quad - \sum_{b=1}^{n_{\mathrm{e}}} \left[\int_{\Omega} N_a N_b \, \mathrm{d}\Omega \right] \frac{\partial u_{\mathrm{e}b}}{\partial t} \end{aligned} \tag{2.15}$$

$$\begin{aligned} 1 &\leq a \leq n_{\mathrm{eq}} \\ 1 &\leq b \leq n_{\mathrm{e}} \end{aligned} \tag{2.16}$$

This example was supposed to illustrate the main steps of the FE method. Note that the mass and stiffness matrix may take different forms depending on the physical phenomena they model and on the material parameters (see Chaps. 3, 5, and 6). The resulting equation (2.12) is still infinite dimensional due to the time dependence. Therefore, in Sect. 2.5 we will discuss time-discretization schemes, in order to arrive at the algebraic system of equations. Before, we will provide a detailed description of the FE method by means of applying it to a 1D problem, followed by discussing all steps necessary for computer implementation.

2.2 Finite Element Method for a 1D Problem

In order to illustrate the main idea of the FE method, we will consider the following 1D differential equation

$$\begin{aligned} -\frac{\partial^2 u}{\partial x^2} + c\,u &= f(x) \\ u(a) &= u_a \\ u(b) &= u_b, \end{aligned} \tag{2.17}$$

where $[a, b]$ defines the computational domain. As described in Sect. 2.1, the first step is to derive the weak form of (2.17). For this purpose, we choose an appropriate test function v, multiply (2.17) by this test function and integrate over the whole domain

$$\int_a^b v \left(-\frac{\partial^2 u}{\partial x^2} + c\,u - f(x) \right) \mathrm{dx}. \tag{2.18}$$

For the first term in (2.18) we perform an integration by parts

$$\int_a^b v \frac{\partial^2 u}{\partial x^2}\ \mathrm{dx} = v \left.\frac{\partial u}{\partial x}\right|_a^b - \int_a^b \frac{\partial v}{\partial x} \frac{\partial u}{\partial x}\ \mathrm{dx}.$$

Provided that the test function $v(x)$ vanishes on the Dirichlet boundary (first restriction on the test function, second one will be the existence of a first-order derivative in the weak sense, see Appendix D), we obtain for (2.18)

$$\int_a^b \left(\frac{\partial v}{\partial x} \frac{\partial u}{\partial x} + cvu \right) \mathrm{dx} = \int_a^b vf\ \mathrm{dx}. \tag{2.19}$$

Therewith, the weak (variational) formulation reads as follows:

Given:
$f, c : [a, b] \to \mathbb{R}$

Find: $u \in V = \{u \in H^1(a, b) | u(a) = u_a\,,\ \ u(b) = u_b\}$ such that for all $v \in W = \{v \in H^1(a, b) | v(a) = v(b) = 0\}$

$$a(u, v) =< f, v > \tag{2.20}$$

with

$$a(u,v) = \int_a^b \left(\frac{\partial v}{\partial x} \frac{\partial u}{\partial x} + cvu \right) \mathrm{dx}$$

$$< f, v > = \int_a^b vf \ \mathrm{dx}.$$

In (2.20) $a(u, v)$ is called a bilinear form and $< f, v >$ an inner product in the specified functional space.

In the next step, we divide the computational domain into cells, so-called finite elements. Therewith, in our case, we divide the interval $[a, b]$ into a set of smaller intervals $[x_{i-1}, x_i]$, $i = 1, \ldots, M$ such that the following properties are fulfilled

- Ascending order of node positions

$$x_{i-1} < x_i \ \text{ for } \ i = 1, \ldots, M$$

- Complete covering of the domain

$$[a, b] = \bigcup_{i=1}^{M} [x_{i-1}, x_i] \quad x_0 = a, \quad x_M = b$$

- No intersection of intervals

$$[x_{i-1}, x_i] \cap [x_{j-1}, x_j] = 0 \ \text{ for } \ i \neq j$$

For simplicity, we choose an equidistant discretization, so that we obtain (see Fig. 2.3)

$$x_i = a + ih \quad h = \frac{b-a}{M} \quad i = 0, \ldots, M.$$

The unknown quantity $u(x)$ is now approximated by a linear combination of finite functions with local support, which means that these functions are just different from zero in a 'small' interval (see Fig. 2.4). Such a choice is given e.g., by piecewise linear hat-functions, defined as follows (see Fig. 2.5)

$$N_i(x) = \begin{cases} 0 & a \leq x \leq x_{i-1} \\ \frac{x - x_{i-1}}{h} & x_{i-1} < x \leq x_i \\ \frac{x_{i+1} - x}{h} & x_i < x \leq x_{i+1} \\ 0 & x_{i+1} < x \leq b \end{cases} \tag{2.21}$$

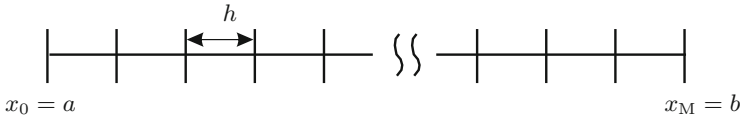


Fig. 2.3 Subdivision of the computational domain into finite elements

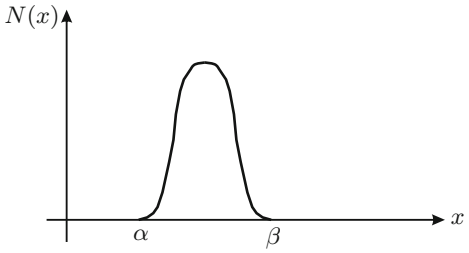


Fig. 2.4 Finite element function with local support: $\text{supp}N(x) = [\alpha, \beta]$

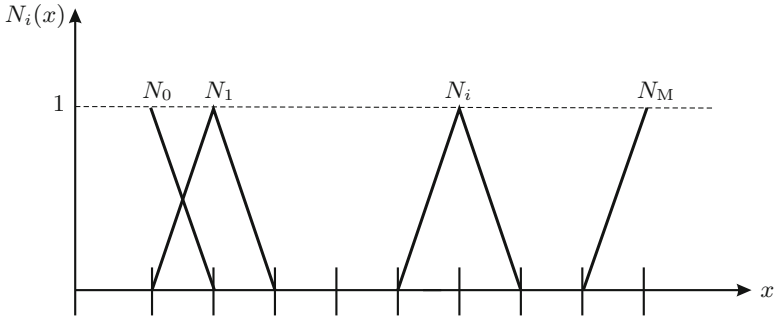


Fig. 2.5 Piecewise, linear hat-functions

Our chosen ansatz (shape) functions fulfill the delta-property

$$N_i(x_j) = \delta_{ij} = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases} \tag{2.22}$$

and the approximation of the unknown $u(x)$ is given by

$$u(x) \approx u^h(x) = \sum_{i=1}^{M-1} N_i(x)u_i + N_0(x)u_a + N_M(x)u_b \tag{2.23}$$

$$u^h(x = x_i) = u_i. \tag{2.24}$$

The discretized weak (variational) formulation reads as

Given:
$f, c : [a, b] \to \mathbb{R}$

Find: $u^h \in V_h = \{u^h(x) = \sum_{i=1}^{M-1} N_i(x)u_i + N_0 u_a + N_M u_b\}$ such that for all $v^h \in W_h = \{v^h(x) = \sum_{i=1}^{M-1} N_i(x)v_i\}$

$$a(u^h, v^h) = < f, v^h > \tag{2.25}$$

with

$$a(u^h, v^h) = \int_a^b \left(\frac{\partial v^h}{\partial x} \frac{\partial u^h}{\partial x} + c v^h u^h \right) \mathrm{dx}$$

$$< f, v^h > = \int_a^b v^h f \ \mathrm{dx}.$$

For the finite dimensional functional spaces V^h, W^h we have the property $V^h \subset V$, $W^h \subset W$.

Now, we have to set up the algebraic system of equations in order to obtain the unknowns u_i at all the finite element nodes within our grid. Using the approximation according to (2.23) for u as well as the test function v results in

$$\int_a^b \frac{\partial}{\partial x} \left(\sum_{i=1}^{M-1} N_i(x)v_i \right) \frac{\partial}{\partial x} \left(\sum_{j=1}^{M-1} N_j(x)u_j + N_0 u_a + N_M u_b \right) \mathrm{dx}$$
$$+ \int_a^b c \left(\sum_{i=1}^{M-1} N_i(x)v_i \right) \left(\sum_{j=1}^{M-1} N_j(x)u_j + N_0 u_a + N_M u_b \right) \mathrm{dx}$$
$$- \int_a^b \left(\sum_{i=1}^{M-1} N_i(x)v_i \right) f \ \mathrm{dx} = 0.$$

Considering that we can interchange the integrals and the sums, and that all v_i as well as u_j are constants (no function of x), we obtain

$$\sum_{i=1}^{M-1} v_i \left(\sum_{j=1}^{M-1} u_j \int_a^b \left(\frac{\partial N_i}{\partial x} \frac{\partial N_j}{\partial x} + c N_i N_j \right) \mathrm{dx} + \int_a^b \left(\frac{\partial N_i}{\partial x} \left(\frac{\partial N_0}{\partial x} u_a + \frac{\partial N_M}{\partial x} u_b \right) + c N_i (N_0 u_a + N_M u_b) \right) \mathrm{dx} - \int_a^b N_i \, f \, \mathrm{dx} \right) = 0.$$

Letting v_i with $i = 1, \ldots, M-1$ run through all unit vectors in $\mathbb{R}^M$, we obtain for each i an equation

$$\sum_{j=1}^{M-1} S_{ij} \, u_j = f_i \qquad i = 1, \ldots, M-1$$

$$\mathbf{S}\underline{u} = \underline{f} \tag{2.26}$$

with

$$S_{ij} = \int_a^b \left(\frac{\partial N_i}{\partial x} \frac{\partial N_j}{\partial x} + c N_i N_j \right) \mathrm{dx} \tag{2.27}$$

$$f_i = \int_a^b N_i \, f \, \mathrm{dx} - \int_a^b \frac{\partial N_i}{\partial x} \left(\frac{\partial N_0}{\partial x} u_a + \frac{\partial N_M}{\partial x} u_b \right) \mathrm{dx} - \int_a^b c N_i (N_0 u_a + N_M u_b) \, \mathrm{dx}. \tag{2.28}$$

According to the properties of our chosen ansatz functions (see (2.22)), we get the following pattern for our system matrix $\mathbf{S}$ (see Fig. 2.6)

$$\mathbf{S} = \begin{pmatrix} * & * & 0 & 0 & \cdots & 0 & \cdots & 0 \\ * & * & * & 0 & \cdots & 0 & \cdots & 0 \\ 0 & * & * & * & \cdots & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & \cdot & \cdot & \cdot & \cdots & * & * & * \\ 0 & \cdot & \cdot & \cdot & \cdots & 0 & * & * \end{pmatrix} \qquad S_{ij} = \begin{cases} 0 & j \notin \{i-1, i, i+1\} \\ * & j \in \{i-1, i, i+1\} \end{cases}$$

$*$...nonzero entry

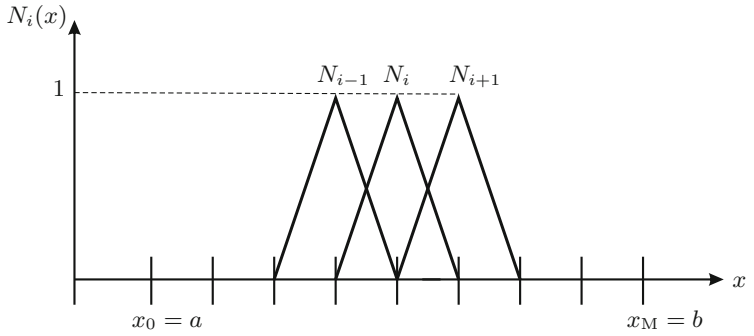


Fig. 2.6 Shape functions for nodes x_{i-1}, x_i and x_{i+1}

Now, let us compute the nonzero entries of **S** by evaluating (2.27) and using (2.21)

$$
\begin{aligned}
S_{i,i-1} &= \int_{x_{i-1}}^{x_i} \left(\frac{\partial N_i}{\partial x} \frac{\partial N_{i-1}}{\partial x} + c N_i N_{i-1} \right) \, \mathrm{dx} \\
&= \int_{x_{i-1}}^{x_i} \left(\left(\frac{1}{h} \right) \left(\frac{-1}{h} \right) + c \left(\frac{x - x_{i-1}}{h} \right) \left(\frac{x_i - x}{h} \right) \right) \, \mathrm{dx} \\
&= \frac{-1}{h} + \frac{ch}{6} \\
S_{i,i} &= \int_{x_{i-1}}^{x_{i+1}} \left(\frac{\partial N_i}{\partial x} \frac{\partial N_i}{\partial x} + c N_i N_i \right) \, \mathrm{dx} \\
&= \int_{x_{i-1}}^{x_i} \left(\left(\frac{1}{h} \right) \left(\frac{1}{h} \right) + c \left(\frac{x - x_{i-1}}{h} \right) \left(\frac{x - x_{i-1}}{h} \right) \right) \, \mathrm{dx} \\
&\quad + \int_{x_i}^{x_{i+1}} \left(\left(-\frac{1}{h} \right) \left(-\frac{1}{h} \right) + c \left(\frac{x_{i+1} - x}{h} \right) \left(\frac{x_{i+1} - x}{h} \right) \right) \, \mathrm{dx} \\
&= \frac{2}{h} + \frac{2ch}{3} \\
S_{i-1,i} &= S_{i,i-1} = \frac{-1}{h} + \frac{ch}{6}.
\end{aligned}
$$

For simplicity, we set our source term $f(x)$ equal to 1 over the whole computational domain and assume $u_a = u_b = 0$, which results in

$$f_i = \int_{x_{i-1}}^{x_{i+1}} N_i(x)\ \mathrm{dx}$$
$$= \int_{x_{i-1}}^{x_i} \frac{x - x_{i-1}}{h}\ \mathrm{dx} + \int_{x_i}^{x_{i+1}} \frac{x_{i+1} - x}{h}\ \mathrm{dx} = h.$$

Let us choose $L = 10$ and $M = 5$, so that our mesh size h is 2. For the parameter c we choose once the value 0 and once 0.5, and we set the boundary values u_a as well as u_b to zero. Substituting these values, results in

$$\mathbf{K}_{c=0} = \begin{pmatrix} 1 & -\frac{1}{2} & 0 & 0 \\ -\frac{1}{2} & 1 & -\frac{1}{2} & 0 \\ 0 & -\frac{1}{2} & 1 & -\frac{1}{2} \\ 0 & 0 & -\frac{1}{2} & 1 \end{pmatrix} \quad \underline{f} = \begin{pmatrix} 2 \\ 2 \\ 2 \\ 2 \end{pmatrix}$$

$$\mathbf{K}_{c=0.5} = \begin{pmatrix} \frac{5}{3} & -\frac{1}{3} & 0 & 0 \\ -\frac{1}{3} & \frac{5}{3} & -\frac{1}{3} & 0 \\ 0 & -\frac{1}{3} & \frac{5}{3} & -\frac{1}{3} \\ 0 & 0 & -\frac{1}{3} & \frac{5}{3} \end{pmatrix}$$

Solving the two algebraic systems of equations results in the solutions displayed in Fig. 2.7. It is worth mentioning that the FE solution corresponding to $c = 0$, which solves the 1D Poisson equation (Laplace operator, see (2.17)), is exact at the FE nodes. However, the case $c = 0.5$ already shows an error at the FE nodes. An a priori error estimate for the discretization error will be discussed in Sect. 2.8.

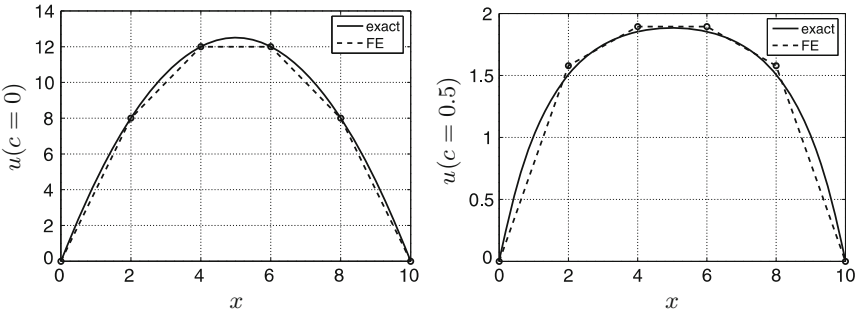


Fig. 2.7 Solutions for the cases $c = 0$ and $c = 0.5$

2.3 Nodal (Lagrangian) Finite Elements

As previously discussed, the FE method subdivides the simulation domain into small elements (e.g., triangles, tetrahedra, etc.) and the unknown quantities are approximated by interpolation functions that have local support. After spatial and time discretization, we end up with an algebraic system of equations. Now our task is to discuss the computation of the matrices (stiffness, mass, etc.) as well as the right-hand side suitable for a computer implementation.

The first important step is to rewrite the integration over the whole domain (see e.g., (2.14)) as a sum of integrations over the element domains, e.g., for the stiffness matrix

$$\begin{aligned} K_{ab} &= \int_{\Omega} \left(\frac{\partial N_a}{\partial x} \frac{\partial N_b}{\partial x} + \frac{\partial N_a}{\partial y} \frac{\partial N_b}{\partial y} \right) \mathrm{d}\Omega \\ &= \sum_{e=1}^{n_\mathrm{e}} \int_{\Omega^e} \left(\frac{\partial N_a}{\partial x} \frac{\partial N_b}{\partial x} + \frac{\partial N_a}{\partial y} \frac{\partial N_b}{\partial y} \right) \mathrm{d}\Omega, \end{aligned} \tag{2.29}$$

with n_e the number of finite elements within the mesh. Therefore, we can introduce the so-called element stiffness (mass, etc.) matrix and obtain

$$\begin{aligned} \boldsymbol{K} &= \bigwedge_{e=1}^{n_\mathrm{e}} \boldsymbol{k}^e \qquad \boldsymbol{k}^e = [k_{pq}] \\ k_{pq} &= \int_{\Omega^e} \left(\frac{\partial N_p}{\partial x} \frac{\partial N_q}{\partial x} + \frac{\partial N_p}{\partial y} \frac{\partial N_q}{\partial y} \right) \mathrm{d}\Omega \\ & \quad 1 \le p, q \le n_\mathrm{en}, \end{aligned}$$

with n_en the number of element nodes and $\bigwedge$ the assembly operator (for the assembling procedure see Sect. 2.4).

In the second step, we shall briefly discuss the computation of the element matrices and the right-hand side. For this task, we need to compute the interpolation functions, their derivatives, and perform numerical integration. The easiest and most general strategy is to introduce transfer functions for the different geometric elements (quadrilateral, tetrahedral, etc.) to their parent elements (see Sect. 2.3). For these parent elements, we have to develop appropriate shape functions as well as numerical integration schemes. In addition, since we need the derivatives of the interpolation functions with respect to the global coordinates x, y, z (also called global derivatives), we have to develop a procedure for performing this task with the help of the transformation as well as local shape functions of the geometric elements (see Sect. 2.3.8). In the case that we choose for the transformation from the local to the global coordinate system

$$\boldsymbol{x}(\boldsymbol{\xi}) = \sum_{i=1}^{n_{\text{en}}} N_i(\boldsymbol{\xi}) \boldsymbol{x}_i$$

the same interpolation functions N_i as for the unknown quantity

$$u^h(\boldsymbol{\xi}) = \sum_{i=1}^{n_{\text{en}}} N_i(\boldsymbol{\xi}) u_i^h$$

we call these finite elements *isoparametric*.

2.3.1 Basic Properties

Let us assume that we want to solve a partial differential equation with the scalar unknown u on the domain Ω with Dirichlet boundary Γ_{e} and Neumann boundary Γ_{n} as displayed in Fig. 2.8. After performing the domain discretization—in this case with triangular finite elements—we obtain 12 finite elements and 11 finite element nodes (see Fig. 2.9). The numbers in the parenthesis are the equation numbers, and as can be seen, for Dirichlet nodes this number is zero (compare with the decomposition of u into the unknown v and known values u_{e}, see Sect. 2.1). Figure 2.10 displays the interpolation function $N_5(\boldsymbol{r})$ for Eq. (5) (finite element node 9). As can be seen, the interpolation function has a *local support*, which means that it has the value 1 at

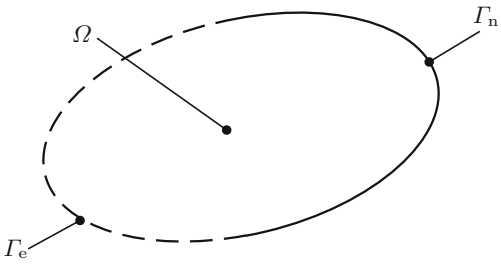


Fig. 2.8 Domain to be discretized

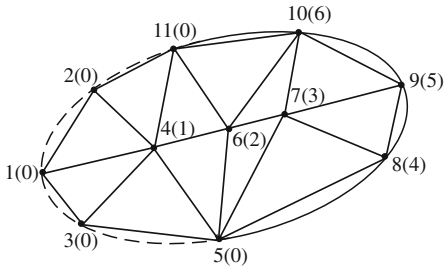


Fig. 2.9 Discretization of the domain with finite elements

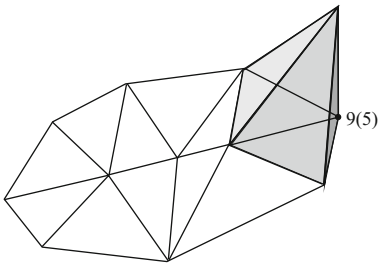


Fig. 2.10 Shape (basis) function for the unknown in FE node 9

the node, decreases to zero approaching the neighboring nodes and is zero outside the neighboring elements. Therefore, the ansatz according to (2.7)–(2.9) is allowed, since, e.g., at any node b it exhibits exactly the value v_b

$$v^h(t)|_{\boldsymbol{r}=\boldsymbol{r}_b} = \sum_{a=1}^{n_{\text{eq}}} N_a(\boldsymbol{r}_b) v_a(t) = v_b(t).$$

Thereby, the interpolation function N_a (also named shape or FE basis function) fulfilles the delta-property

$$N_a(\boldsymbol{r}_b) = \begin{cases} 1 \text{ for } a = b \\ 0 \; else \end{cases} \tag{2.30}$$

In addition, it is now clear that the mass, stiffness as well as effective system matrices show a sparse profile, since the integrals defining their entries (see (2.13) and (2.14)) are nonzero only if the supports of the interpolation functions N_a and N_b overlap, which only happens if the functions belong to neighboring nodes. For our simple example we obtain the following matrix structure

$$\begin{pmatrix} a_{11} & a_{12} & 0 & 0 & 0 & 0 \\ a_{21} & a_{22} & a_{23} & 0 & 0 & a_{26} \\ 0 & a_{32} & a_{33} & a_{34} & a_{35} & a_{36} \\ 0 & 0 & a_{43} & a_{44} & a_{45} & 0 \\ 0 & 0 & a_{53} & a_{54} & a_{55} & a_{56} \\ 0 & a_{62} & a_{63} & 0 & a_{65} & a_{66} \end{pmatrix}.$$

The main properties to be fulfilled by any nodal finite element are:

1. Smoothness on each element interior Ω^e
2. Continuity across each element boundary Γ^e
3. Completeness

Let us assume that we have to evaluate finite element matrices with partial derivatives of order m in the weak formulation. Then, properties 1 and 2 demand for any nodal finite element basis functions, which are m times differentiable in Ω^e and $(m-1)$

times differentiable over the boundary Γ^e. For example, for $m = 1$ any finite element with linear interpolation functions (C^0 finite element) fulfills property 1 and 2. In general we call finite elements satisfying property 1 and 2 *conforming*, or *compatible*.

Now, let us illustrate the property *completeness*. We assume the following ansatz for the unknown quantity u

$$u^h(x) = \sum_{a=1}^{n_{\text{en}}} N_a u_a^e \ ,$$

with n_{en} the number of nodes for the finite element e, and the property

$$u^h(\boldsymbol{x}_a^e) = u_a^e.$$

The finite element is said to be complete if

$$u_a^e = c_0 + c_1 x_a^e + c_2 y_a^e + c_3 z_a^e \tag{2.31}$$

implies

$$u^h(\boldsymbol{x}) = c_0 + c_1 x + c_2 y + c_3 z, \tag{2.32}$$

with arbitrary constants $c_0 \ \ldots \ c_3$. This means the element interpolation functions are capable of exactly representing an arbitrary linear polynomial. Therewith, it is guaranteed that by reducing the mesh size h^e, the approximated solution converges towards the exact solution.

2.3.2 Quadrilateral Element in $\mathbb{R}^2$

For the computation of the element matrices, it is convenient to transform each finite element to its reference element, where the numerical integration can be performed easily. Therefore, we introduce the bijective map F_{Ω_e} from the reference element $\hat{\Omega}$ to the global finite element with domain Ω_e, which is an element of the computational grid with index e

$$F_e : \hat{\Omega} \ \rightarrow \Omega_e. \tag{2.33}$$

Let us investigate this transformation for the bilinear quadrilateral element in two space dimension as displayed in Fig. 2.11. The local coordinates

$$\boldsymbol{\xi} = \begin{pmatrix} \xi \\ \eta \end{pmatrix} \tag{2.34}$$

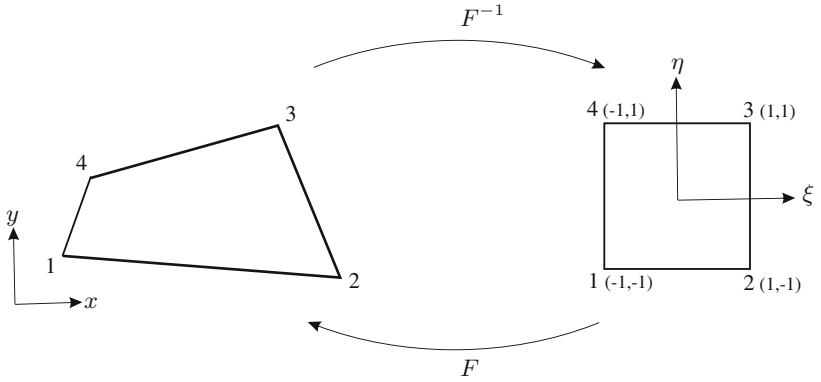


Fig. 2.11 Bilinear quadrilateral element: global and local domain

are related to the global coordinates

$$\boldsymbol{x} = \begin{pmatrix} x \\ y \end{pmatrix} \tag{2.35}$$

via the following transformation

$$\boldsymbol{x}(\boldsymbol{\xi}) = \begin{cases} x(\xi, \eta) = \sum_{i=1}^{4} N_i(\xi, \eta) x_i^e \\ y(\xi, \eta) = \sum_{i=1}^{4} N_i(\xi, \eta) y_i^e \end{cases} \tag{2.36}$$

Now, in order to compute explicitly the basis functions N_i, we choose the following bilinear ansatz

$$x(\xi, \eta) = \alpha_0 + \alpha_1 \xi + \alpha_2 \eta + \alpha_3 \xi\eta \tag{2.37}$$
$$y(\xi, \eta) = \beta_0 + \beta_1 \xi + \beta_2 \eta + \beta_3 \xi\eta. \tag{2.38}$$

The shape functions have to be constructed in such a way that the relations

$$x(\xi_i, \eta_i) = x_i^e$$
$$y(\xi_i, \eta_i) = y_i^e$$

are fulfilled. Since the local coordinates take only the following values

node i	ξ_i	η_i
1	-1	-1
2	1	-1
3	1	1
4	-1	1

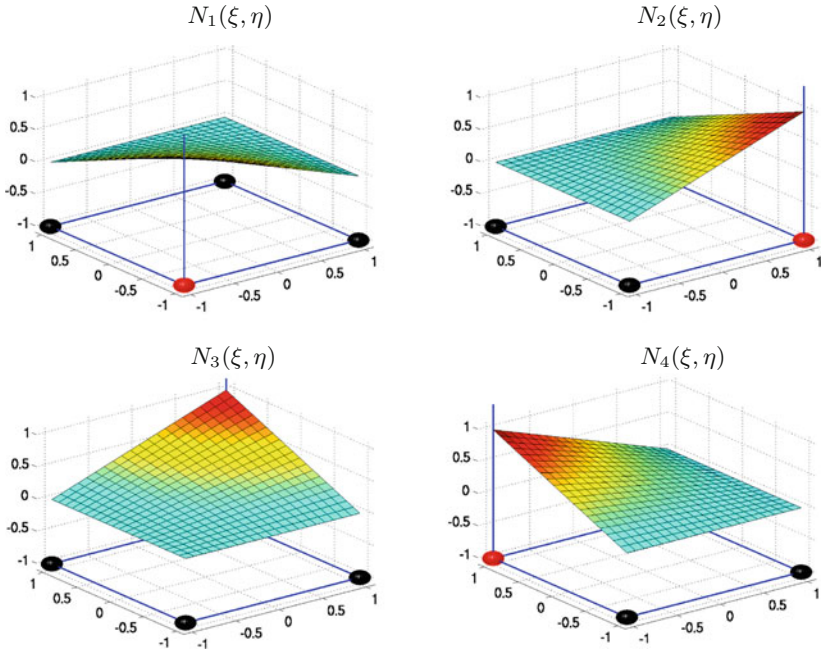


Fig. 2.12 The four bilinear shape functions of a quadrilateral element

we obtain 8 equations for the eight unknowns α_0 .. β_3. Using the solution for α_i and β_i in (2.37) and (2.38) and comparing the coefficients with (2.36), we arrive at the following explicit form of the shape function for node i (see Fig. 2.12)

$$N_i(\boldsymbol{\xi}) = N_i(\xi,\eta) = \frac{1}{4}(1+\xi_i\xi)(1+\eta_i\eta)\,. \tag{2.39}$$

Now, let us investigate the three mentioned properties for the quadrilateral element.

1. *Smoothness* on each element interior Ω^e:
   The shape functions N_i define smooth functions, if each interior angle of the quadrilateral is less then 180°.
2. *Continuity* across each element boundary Γ^e:
   Figure 2.12 displays the shape function N_i for node i defined by (ξ_i, η_i), and it is easy to see that
   $$N_i(\xi_i,\eta_i) = \delta_{ij} \tag{2.40}$$
   is fulfilled. Along the boundary, e.g., $\eta = -1$, we obtain
   $$N_i(\xi,-1) = \frac{1+\xi_i\xi}{2}\,,$$

which is exactly the shape function for the 1D case. Since this shape function is typically the same for all edges, the quadrilateral element fulfills the continuity condition.

3. *Completeness*:

$$\begin{aligned} u^h &= \sum_{i=1}^{n_{\text{en}}} N_i(\xi,\eta) u_i^e \\ &= \sum_{i=1}^{n_{\text{en}}} N_i(\xi,\eta)(c_0 + c_1 x_i^e + c_2 y_i^e) \\ &= \left(\sum_{i=1}^{n_{\text{en}}} N_i(\xi,\eta)\right) c_0 + \underbrace{\left(\sum_{i=1}^{n_{\text{en}}} N_i(\xi,\eta) x_i^e\right)}_{x(\xi,\eta)} c_1 + \underbrace{\left(\sum_{i=1}^{n_{\text{en}}} N_i(\xi,\eta) y_i^e\right)}_{y(\xi,\eta)} c_2 \end{aligned}$$

Summing up all four shape functions results in

$$\begin{aligned} \sum_{i=1}^{n_{\text{en}}} N_i(\xi,\eta) &= \frac{1}{4}[(1-\xi)(1-\eta) + (1+\xi)(1-\eta) \\ &\quad +(1+\xi)(1+\eta) + (1-\xi)(1+\eta)] \\ &= 1\,, \end{aligned}$$

which proves the completeness.

2.3.3 Triangular Element in $\mathbb{R}^2$

The linear triangular element is defined by its three nodes as displayed in Fig. 2.13. The local coordinates are as follows

node i	ξ_i	η_i
1	0	0
2	1	0
3	0	1

Similar to the quadrilateral element (see above) we obtain the local shape functions given by

$$\begin{aligned} N_1 &= 1 - \xi - \eta \\ N_2 &= \xi \\ N_3 &= \eta. \end{aligned}$$

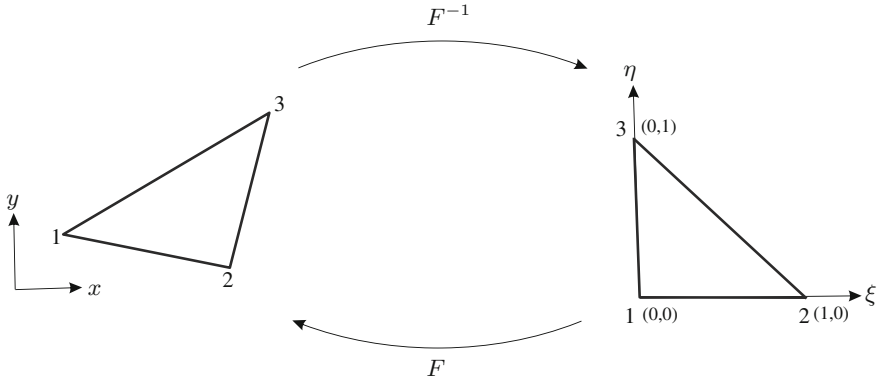


Fig. 2.13 Triangular element: global and local domain

2.3.4 Tetrahedron Element in $\mathbb{R}^3$

The linear tetrahedron element is defined by its four coordinates as shown in Fig. 2.14.

node i	ξ_i	η_i	ζ_i
1	0	0	0
2	1	0	0
3	0	1	0
4	0	0	1

Let us compute the transformation that maps any arbitrary tetrahedral element in the global (x, y, z)-domain to a parent tetrahedron in the local (ξ, η, ζ)-domain by choosing the following linear ansatz

$$x_i = \alpha_0 + \alpha_1\xi_i + \alpha_2\eta_i + \alpha_3\zeta_i \tag{2.41}$$
$$y_i = \beta_0 + \beta_1\xi_i + \beta_2\eta_i + \beta_3\zeta_i \tag{2.42}$$
$$z_i = \gamma_0 + \gamma_1\xi_i + \gamma_2\eta_i + \gamma_3\zeta_i\,. \tag{2.43}$$

We know that the transformation has to satisfy the following relations at the four nodes of a tetrahedron element

$$\boldsymbol{x}(\boldsymbol{\xi}) = \boldsymbol{x}_a^e \quad a = 1, \ \ldots\ , 4.$$

Therefore, we obtain (see Fig. 2.14)

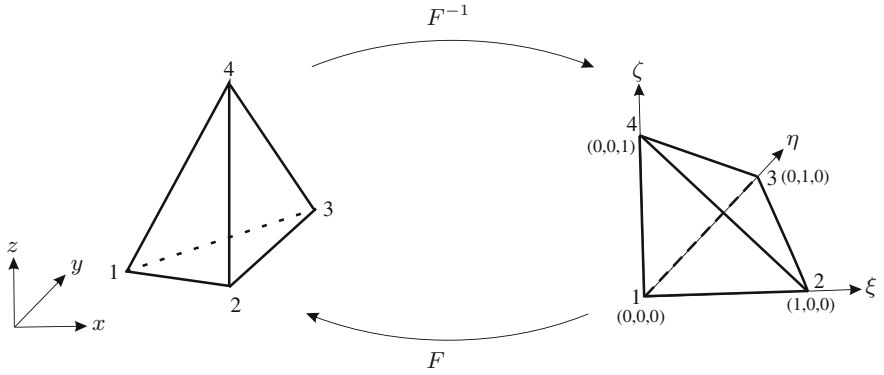


Fig. 2.14 Tetrahedron element: global and local domain

$$x_1 = \alpha_0 \quad y_1 = \beta_0 \tag{2.44}$$

$$x_2 = \alpha_0 + \alpha_1 \quad y_2 = \beta_0 + \beta_1 \tag{2.45}$$

$$x_3 = \alpha_0 + \alpha_2 \quad y_3 = \beta_0 + \beta_2 \tag{2.46}$$

$$x_4 = \alpha_0 + \alpha_3 \quad y_4 = \beta_0 + \beta_3. \tag{2.47}$$

Solving the above system of equations, we arrive at a general expression for the shape function N_i as a function of the local coordinates

$$N_1 = 1 - \xi - \eta - \zeta$$

$$N_2 = \xi$$

$$N_3 = \eta$$

$$N_4 = \zeta.$$

2.3.5 Hexahedron Element in $\mathbb{R}^3$

In many 3D applications hexahedron elements are used for the domain discretization, due to their good approximation property. Figure 2.15 displays the hexahedron element in its global and local coordinate system.

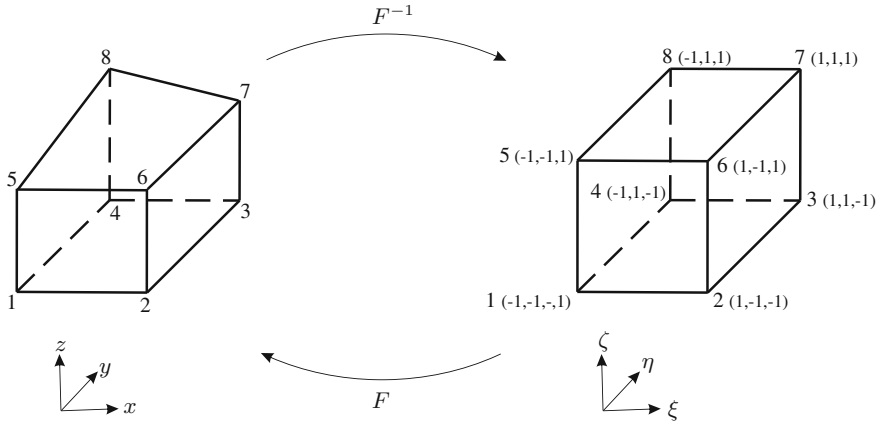


Fig. 2.15 Hexahedron element: global and local domain

node i	ξ_i	η_i	ζ_i
1	−1	−1	−1
2	1	−1	−1
3	1	1	−1
4	−1	1	−1
5	−1	−1	1
6	1	−1	1
7	1	1	1
8	−1	1	1

For the element a trilinear mapping is applied between the global (defined by $\boldsymbol{x}$) and the local (defined by $\boldsymbol{\xi}$) element domain [4]

$$\boldsymbol{x}(\boldsymbol{\xi}) = \alpha_0 + \alpha_1\xi + \alpha_2\eta + \alpha_3\zeta + \alpha_4\xi\eta + \alpha_5\eta\zeta + \alpha_6\xi\zeta + \alpha_7\xi\eta\zeta. \tag{2.48}$$

The coefficients α_i are determined by the relations (see Fig. 2.15)

$$\boldsymbol{x}(\boldsymbol{\xi}_a) = \boldsymbol{x}_a^e \quad a = 1, \ldots, 8, \tag{2.49}$$

which results in

$$N_a(\boldsymbol{\xi}) = \frac{1}{8}(1 + \xi_a\xi)(1 + \eta_a\eta)(1 + \zeta_a\zeta). \tag{2.50}$$

Using a simple degeneration technique [4], one can obtain a pyramid, a wedge as well as a tetrahedron element from a hexahedron one.

2.3.6 Wedge Element in $\mathbb{R}^3$

The trilinear wedge element is defined by its six nodes as displayed in Fig. 2.16. The local coordinates are as follows

node i	ξ_i	η_i	ζ_i
1	0	0	-1
2	1	0	-1
3	0	1	-1
4	0	0	1
5	1	0	1
6	0	1	1

Thereby, the local shape functions compute as follows

$$N_1 = \frac{1}{2}(1-\zeta)(1-\xi-\eta)$$
$$N_2 = \frac{1}{2}(1-\zeta)\,\xi$$
$$N_3 = \frac{1}{2}(1-\zeta)\,\eta$$

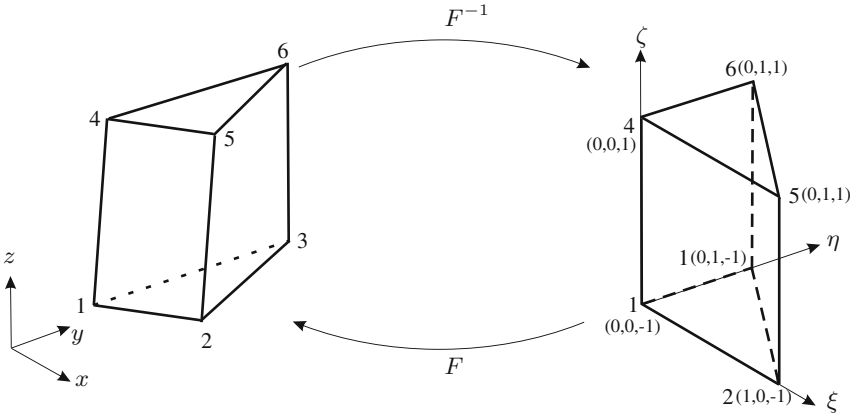


Fig. 2.16 Wedge element: global and local domain

$$N_4 = \frac{1}{2}(1+\zeta)(1-\xi-\eta)$$
$$N_5 = \frac{1}{2}(1+\zeta)\,\xi$$
$$N_6 = \frac{1}{2}(1+\zeta)\,\eta.$$

2.3.7 *Pyramidal Element in* $\mathbb{R}^3$

The first order pyramidal element is defined by its five nodes as displayed in Fig. 2.17. The local coordinates are defines as follows

node i	ξ_i	η_i	ζ_i
1	1	1	0
2	1	−1	0
3	−1	−1	0
4	−1	1	0
5	0	0	1

This type of element is of great importance, when one performs a meshing of an inner domain with hexahedral elements and an outer domain with tetrahedrals. Then, pyramidal elements are necessary as transition elements.

To obtain consistent basis functions for a pyramidal element is not trivial. Using the classical approach of a polynomial ansatz is not working. According to [5] the main

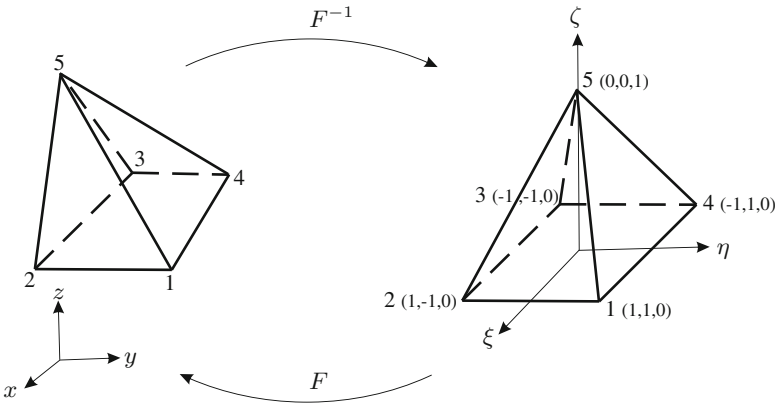


Fig. 2.17 Pyramidal element: global and local domain

idea is to construct basis functions by adding a rational expression to the polynomial functions. The use of the rational term allows to achieve a non-singular Jacobian and continuity between triangular faces. Thereby, the basis functions compute as

$$
\begin{aligned}
N_1 &= \frac{1}{4}\left((1+\xi)(1+\eta) - \zeta + \frac{\xi\eta\zeta}{1-\zeta}\right)\\
N_2 &= \frac{1}{4}\left((1+\xi)(1-\eta) - \zeta + \frac{\xi\eta\zeta}{1-\zeta}\right)\\
N_3 &= \frac{1}{4}\left((1-\xi)(1-\eta) - \zeta + \frac{\xi\eta\zeta}{1-\zeta}\right)\\
N_4 &= \frac{1}{4}\left((1-\xi)(1+\eta) - \zeta + \frac{\xi\eta\zeta}{1-\zeta}\right)\\
N_5 &= \zeta.
\end{aligned}
$$

It has to be noted that the rational term at $\zeta = 1$ is zero, since both ξ and η are zero.

2.3.8 Global/Local Derivatives

For the computation we need to evaluate derivatives of the shape functions with respect to the global coordinate system (see e.g., (2.14)). Since the shape functions N_i depend on the local coordinates (ξ, η, ζ), we may write

$$
\begin{aligned}
N_{a,x} &= N_{a,\xi}\xi_{,x} + N_{a,\eta}\eta_{,x} + N_{a,\zeta}\zeta_{,x}\\
N_{a,y} &= N_{a,\xi}\xi_{,y} + N_{a,\eta}\eta_{,y} + N_{a,\zeta}\zeta_{,y}\\
N_{a,z} &= N_{a,\xi}\xi_{,z} + N_{a,\eta}\eta_{,z} + N_{a,\zeta}\zeta_{,z},
\end{aligned}
$$

with the notation, e.g., $\xi_{,x} = \partial\xi/\partial x$. In matrix form, we obtain

$$
\begin{pmatrix} N_{a,x} \\ N_{a,y} \\ N_{a,z} \end{pmatrix} = \begin{bmatrix} \xi_{,x} & \eta_{,x} & \zeta_{,x} \\ \xi_{,y} & \eta_{,y} & \eta_{,y} \\ \xi_{,z} & \eta_{,z} & \zeta_{,z} \end{bmatrix} \begin{pmatrix} N_{a,\xi} \\ N_{a,\eta} \\ N_{a,\zeta} \end{pmatrix}. \tag{2.51}
$$

Now, we do not have the explicit form of the derivatives of the local coordinates with respect to the global coordinates. However, we can express this relation as follows

$$
\begin{pmatrix} N_{a,\xi} \\ N_{a,\eta} \\ N_{a,\zeta} \end{pmatrix} = \begin{bmatrix} x_{,\xi} & y_{,\xi} & z_{,\xi} \\ x_{,\eta} & y_{,\eta} & z_{,\eta} \\ x_{,\zeta} & y_{,\zeta} & z_{,\zeta} \end{bmatrix} \begin{pmatrix} N_{a,x} \\ N_{a,y} \\ N_{a,z} \end{pmatrix}. \tag{2.52}
$$

Comparing (2.51) and (2.52) we arrive at

$$\begin{bmatrix} \xi_{,x} & \eta_{,x} & \zeta_{,x} \\ \xi_{y} & \eta_{,y} & \eta_{,y} \\ \xi_{,z} & \eta_{,z} & \zeta_{,z} \end{bmatrix} = \underbrace{\begin{bmatrix} x_{,\xi} & y_{,\xi} & z_{,\xi} \\ x_{,\eta} & y_{,\eta} & z_{,\eta} \\ x_{,\zeta} & y_{,\zeta} & z_{,\zeta} \end{bmatrix}^{-1}}_{(\mathcal{J}^T)^{-1}=\mathcal{J}^{-T}}, \tag{2.53}$$

with $\mathcal{J}$ the Jacobi matrix. The computation of $\mathcal{J}$ can be performed by the transformation between the global and local coordinate systems

$$\begin{aligned} x(\xi,\eta,\zeta) &= \sum_{a=1}^{n_{\text{en}}} N_a(\xi,\eta,\zeta) x_a^e \\ y(\xi,\eta,\zeta) &= \sum_{a=1}^{n_{\text{en}}} N_a(\xi,\eta,\zeta) y_a^e \\ z(\xi,\eta,\zeta) &= \sum_{a=1}^{n_{\text{en}}} N_a(\xi,\eta,\zeta) z_a^e. \end{aligned}$$

Therefore, the explicit expression for the Jacobian reads as

$$\mathcal{J} = \begin{bmatrix} x_{,\xi} & x_{,\eta} & x_{,\zeta} \\ y_{,\xi} & y_{,\eta} & y_{,\zeta} \\ z_{,\xi} & z_{,\eta} & z_{,\zeta} \end{bmatrix} = \begin{bmatrix} \sum_{a=1}^{n_{\text{en}}} N_{a,\xi} x_a^e & \sum_{a=1}^{n_{\text{en}}} N_{a,\eta} x_a^e & \sum_{a=1}^{n_{\text{en}}} N_{a,\zeta} x_a^e \\ \sum_{a=1}^{n_{\text{en}}} N_{a,\xi} y_a^e & \sum_{a=1}^{n_{\text{en}}} N_{a,\eta} y_a^e & \sum_{a=1}^{n_{\text{en}}} N_{a,\zeta} y_a^e \\ \sum_{a=1}^{n_{\text{en}}} N_{a,\xi} z_a^e & \sum_{a=1}^{n_{\text{en}}} N_{a,\eta} z_a^e & \sum_{a=1}^{n_{\text{en}}} N_{a,\zeta} z_a^e \end{bmatrix}. \tag{2.54}$$

The algorithm can be summarized as follows (n_{int} denotes the number of integration points, see next section):

for $l := 1, n_{\text{int}}$
  Determine: $W_l, \tilde{\xi}_l, \tilde{\eta}_l, \tilde{\zeta}_l$
  for $a := 1, n_{\text{en}}$
    Calculate: N_a , $N_{a,\xi}$, $N_{a,\eta}$, $N_{a,\zeta}$ at $(\tilde{\xi}_l, \tilde{\eta}_l, \tilde{\zeta}_l)$
  end

  Compute Jacobi matrix, determinant and its inverse
  Compute global derivatives $N_{a,x}$, $N_{a,y}$, $N_{a,z}$ at $(\tilde{\xi}_l, \tilde{\eta}_l, \tilde{\zeta}_l)$

end

2.3.9 Numerical Integration

For the computation of the element matrices as well as element right-hand sides we have to numerically evaluate an integral of the form

$$\int_{\Omega^e} f(\boldsymbol{x})\,\mathrm{d}\Omega. \tag{2.55}$$

Since we perform a transformation of each finite element to its parent element, (2.55) changes, e.g., for a hexahedron, to

$$\int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1} f(\boldsymbol{x}(\boldsymbol{\xi}))|\mathcal{J}|\,\mathrm{d}\xi\,\mathrm{d}\eta\,\mathrm{d}\zeta, \tag{2.56}$$

with $|\mathcal{J}|$ the Jacobi determinant (see Sect. 2.3.8). In the 1D case a *Gaussian quadrature formula* is optimal, since by using n_{int} integration points, we achieve an accuracy of order $2n_{\mathrm{int}}$ (see e.g., [4])

$$\int_{-1}^{1} g(\xi)\,\mathrm{d}\xi = \sum_{l=1}^{n_{\mathrm{int}}} g(\tilde{\xi}_l)\,W_l + E \tag{2.57}$$

$\tilde{\xi}_l$ ... zero positions of Legendre polynomial with order n_{int}
W_l ... weighting factor for integration point l
E ... error.

For our 3D case, we can write

$$\begin{aligned}\int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1} f(\boldsymbol{x}(\boldsymbol{\xi}))|\mathcal{J}|\,\mathrm{d}\xi\,\mathrm{d}\eta\,\mathrm{d}\zeta &= \sum_{l^1=1}^{n^1_{\mathrm{int}}}\sum_{l^2=1}^{n^2_{\mathrm{int}}}\sum_{l^3=1}^{n^3_{\mathrm{int}}} g(\tilde{\xi}_{l^1},\tilde{\eta}_{l^2},\tilde{\zeta}_{l^3})\,W_{l^1}W_{l^2}W_{l^3} + E \\ &= \sum_{l=1}^{n_{\mathrm{int}}} g(\tilde{\xi}_l,\tilde{\eta}_l,\tilde{\zeta}_l)\,W_l + E.\end{aligned} \tag{2.58}$$

In the following, we give for each discussed geometric element the integration points as well as the weighting factors.

- Quadrilateral elements (Gaussian quadrature, 2nd order):

l	ξ_l	η_l	W_l
1	−0.57735026919	−0.57735026919	1.0
2	0.57735026919	−0.57735026919	1.0
3	0.57735026919	0.57735026919	1.0
4	−0.57735026919	0.57735026919	1.0

- Triangular elements (Gaussian quadrature, 2nd order):

l	ξ_l	η_l	W_l
1	0.166 666 67	0.166 666 67	0.166 666 67
2	0.666 666 67	0.166 666 67	0.166 666 67
3	0.166 666 67	0.666 666 67	0.166 666 67

- Tetrahedron elements (Gaussian quadrature, 2nd order):

l	ξ_l	η_l	ζ_l	W_l
1	0.585 410	0.138 196	0.138 196	0.041 666 7
2	0.138 196	0.585 410	0.138 196	0.041 666 7
3	0.138 196	0.138 196	0.585 410	0.041 666 7
4	0.138 196	0.138 196	0.138 196	0.041 666 7

- Hexahedron elements (Gaussian quadrature, 2nd order):

l	ξ_l	η_l	ζ_l	W_l
1	−0.57735026919	−0.57735026919	−0.57735026919	1.0
2	0.57735026919	−0.57735026919	−0.57735026919	1.0
3	0.57735026919	0.57735026919	−0.57735026919	1.0
4	−0.57735026919	0.57735026919	−0.57735026919	1.0
5	−0.57735026919	−0.57735026919	0.57735026919	1.0
6	0.57735026919	−0.57735026919	0.57735026919	1.0
7	0.57735026919	0.57735026919	0.57735026919	1.0
8	−0.57735026919	0.57735026919	0.57735026919	1.0

- Wedge elements (Gaussian quadrature, 2nd order):

l	ξ_l	η_l	ζ_l	W_l
1	0.16666666667	0.16666666667	−0.57735026919	0.16666666667
2	0.66666666667	0.16666666667	−0.57735026919	0.16666666667
3	0.16666666667	0.66666666667	−0.57735026919	0.16666666667
4	0.16666666667	0.16666666667	0.57735026919	0.16666666667
5	0.66666666667	0.16666666667	0.57735026919	0.16666666667
6	0.16666666667	0.66666666667	0.57735026919	0.16666666667

- Pyramidal elements (Gaussian quadrature, 2nd order):

l	ξ_l	η_l	ζ_l	W_l
1	0.0	0.0	0.69370598373	0.21333333333
2	−0.4879500365	−0.4879500365	0.16548457453	0.28000
3	0.4879500365	−0.4879500365	0.16548457453	0.28000
4	0.4879500365	0.4879500365	0.16548457453	0.28000
5	−0.4879500365	0.4879500365	0.16548457453	0.28000

2.4 Finite Element Procedure

In the previous sections we discussed the basis function of different geometric finite elements, the global/local derivatives and the numerical integration. For a general definition of the computation of the element matrices as well as right-hand side, we use the concept of mapping (see Sect. 2.3.2). Therefore, the transformation to the integral can be written as

$$\int_{\Omega_e} f(\boldsymbol{x})\ \mathrm{d}\Omega = \int_{\hat{\Omega}} |\mathcal{J}_e| \hat{f}(\boldsymbol{\xi})\, \mathrm{d}\hat{\Omega}. \tag{2.59}$$

In (2.59) $\mathcal{J}_e$ is the Jacobian of the grid element with index e. The FE basis functions are defined on the local element and do not need any further mapping. However, e.g. the nabla operator, which contains the derivatives with respect to the global coordinates, transforms with the Jacobian $\mathcal{J}_e$ by

$$\nabla \rightarrow \mathcal{J}_e^{-T}\, \hat{\nabla}. \tag{2.60}$$

Here, $\hat{\nabla}$ denotes the nabla-operator with respect to the local coordinates. Summarizing, we can write the bilinear form of a simple Laplace-problem (see (2.29)) as follows

$$\int_{\Omega_e} \nabla N_a \cdot \nabla N_b\ \mathrm{d}\Omega = \int_{\hat{\Omega}} |\mathcal{J}_e| \mathcal{J}_e^{-T} \hat{\nabla} N_a \cdot \mathcal{J}_e^{-T} \hat{\nabla} N_b\ \mathrm{d}\hat{\Omega}. \tag{2.61}$$

Finally, we address the still-open question of the assembly procedure. In the first step we introduce the *nodal equation* array *NE*, which relates the global equation number P to the global node number A.

$$NE(A) = \begin{cases} P & : \text{if the quantity is unknown at } A \\ 0 & : \text{if the quantity is known at } A \\ & \quad (\text{e.g., Dirichlet boundary condition}) \end{cases}$$

A ... global node number

P ... global equation number.

Since the whole simulation domain is discretized with finite elements, and we first compute the element matrices (right-hand side) and then assemble it to the global system matrix (right-hand side), we need information given by the following *information element node* array *IEN*

$$IEN(a, e) = A$$

a ... local element node number

e ... element number

A ... global node number.

Assuming that we solve a scalar PDE, we have just one unknown per FE node, and the local node number coincides with the local equation number. Combining the *NE* array with the *IEN* array results in the *equation* array *EQ*

$$EQ(a, e) = NE\,(IEN(a, e)) = P\ .$$

The *EQ* array connects the element node number (element equation number) a of element e with the global equation number P. The following simple example, displayed in Fig. 2.18, will demonstrate all the steps that have to be performed for the assembly process. The FE mesh consists of two finite elements with given Dirichlet boundary conditions u_1, u_2 at node 1 and 2. Let us write the algebraic system of equations for some discretized PDE on this domain (e.g., the Poisson equation) in the following general form

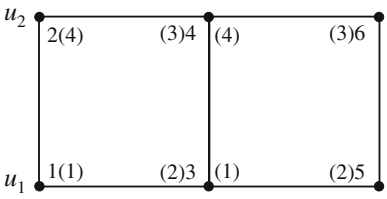


Fig. 2.18 Example: global node numbers and in parenthesis the local node numbers

$$\begin{bmatrix} K_{11} & K_{12} & K_{13} & K_{14} & K_{15} & K_{16} \\ K_{21} & K_{22} & K_{23} & K_{24} & K_{25} & K_{26} \\ K_{31} & K_{32} & K_{33} & K_{34} & K_{35} & K_{36} \\ K_{41} & K_{42} & K_{43} & K_{44} & K_{45} & K_{46} \\ K_{51} & K_{52} & K_{53} & K_{54} & K_{55} & K_{56} \\ K_{61} & K_{62} & K_{63} & K_{64} & K_{65} & K_{66} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \\ f_6 \end{bmatrix}. \tag{2.62}$$

Since we know u_1 and u_2 we can rewrite (2.62) as

$$\begin{bmatrix} K_{33} & K_{34} & K_{35} & K_{36} \\ K_{43} & K_{44} & K_{45} & K_{46} \\ K_{53} & K_{54} & K_{55} & K_{56} \\ K_{63} & K_{64} & K_{65} & K_{66} \end{bmatrix} \begin{bmatrix} u_3 \\ u_4 \\ u_5 \\ u_6 \end{bmatrix} = \begin{bmatrix} f_3 - K_{31}u_1 - K_{32}u_2 \\ f_4 - K_{41}u_1 - K_{42}u_2 \\ f_5 - K_{51}u_1 - K_{52}u_2 \\ f_6 - K_{61}u_1 - K_{62}u_2 \end{bmatrix}. \tag{2.63}$$

The nodal equation array *NE* is given by

1	2	3	4	5	6
0*	0*	1	2	3	4

$*$: at this node we already know the quantity (Dirichlet boundary condition)

and using it for setting up the algebraic system, we recognize that we obtain a similar system as in (2.63) with four unknowns. Since there is no connection between the nodes 1, 2 and 5, 6 (see Fig. 2.18), the values of K_{51}, K_{52}, K_{61}, and K_{62} are zero.

The number of nodes for the linear quadrilateral element is $n_{\text{en}} = 4$ and the unknown quantity is a scalar. Therefore, we obtain the following element matrices as well as right-hand sides for the two finite elements

$$\mathbf{k}^1 = \begin{bmatrix} k_{11}^1 & k_{12}^1 & k_{13}^1 & k_{14}^1 \\ \{k_{21}^1\} & (k_{22}^1) & (k_{23}^1) & \{k_{24}^1\} \\ \{k_{31}^1\} & (k_{32}^1) & (k_{33}^1) & \{k_{34}^1\} \\ k_{41}^1 & k_{42}^1 & k_{43}^1 & k_{44}^1 \end{bmatrix} \quad \underline{f}^1 = \begin{bmatrix} f_1^1 \\ < f_2^1 > \\ < f_3^1 > \\ f_4^1 \end{bmatrix}$$

$$\mathbf{k}^2 = \begin{bmatrix} (k_{11}^2) & (k_{12}^2) & (k_{13}^2) & (k_{14}^2) \\ (k_{21}^2) & (k_{22}^2) & (k_{23}^2) & (k_{24}^2) \\ (k_{31}^2) & (k_{32}^2) & (k_{33}^2) & (k_{34}^2) \\ (k_{41}^2) & (k_{42}^2) & (k_{43}^2) & (k_{44}^2) \end{bmatrix} \quad \underline{f}^2 = \begin{bmatrix} < f_1^2 > \\ < f_2^2 > \\ < f_3^2 > \\ < f_4^2 > \end{bmatrix}$$

() ... contributes to the global stiffness matrix $\mathbf{K}$
$< >$ ... contributes to the global right-hand side $\underline{f}$
{} ... contributes to the local right-hand side $\underline{f}^e$.

Table 2.1 Information element node (IEN) and global equation array (EQ) for our exampled displayed in Fig. 2.18

IEN		el.nr.		EQ		el.nr.	
		1	**2**			**1**	**2**
Local	**1**	1	3	Local	**1**	0	1
node	**2**	3	5	node	**2**	1	3
number	**3**	4	6	number	**3**	2	4
	4	2	4		**4**	0	2

The entries of the right-hand side $\underline{f}^1$ for element 1 compute as

$$\underline{f}^1 = \begin{bmatrix} f_1^1 \\ f_2^1 - k_{21}^1 u_1 - k_{24}^1 u_2 \\ f_3^1 - k_{31}^1 u_1 - k_{34}^1 u_2 \\ f_4^1 \end{bmatrix}.$$

Combining the *NE* array with the element node array *IEN*, we arrive at the *EQ* array (Table 2.1).

Assembling:

$$\begin{aligned}
&EQ(1,1) = 0 \\
&EQ(2,1) = 1 \quad \begin{cases} K_{11} \leftarrow K_{11} + k_{22}^1 \\ K_{12} \leftarrow K_{12} + k_{23}^1 \end{cases} \quad f_1 \leftarrow f_2^1 \\
&EQ(3,1) = 2 \quad \begin{cases} K_{22} \leftarrow K_{22} + k_{33}^1 \\ K_{21} \leftarrow K_{21} + k_{32}^1 \end{cases} \quad f_2 \leftarrow f_3^1 \\
&EQ(4,1) = 0 \\
&EQ(1,2) = 1 \quad \begin{cases} K_{11} \leftarrow K_{11} + k_{11}^2 \\ K_{13} \leftarrow K_{13} + k_{12}^2 \\ K_{14} \leftarrow K_{14} + k_{13}^2 \\ K_{12} \leftarrow K_{12} + k_{14}^2 \end{cases} \quad f_1 \leftarrow f_1^2 \\
&EQ(2,2) = 3 \quad \begin{cases} K_{33} \leftarrow K_{33} + k_{22}^2 \\ K_{31} \leftarrow K_{31} + k_{21}^2 \\ K_{34} \leftarrow K_{34} + k_{23}^2 \\ K_{32} \leftarrow K_{32} + k_{24}^2 \end{cases} \quad f_3 \leftarrow f_2^2 \\
&EQ(3,2) = 4 \quad \begin{cases} K_{44} \leftarrow K_{44} + k_{33}^2 \\ K_{41} \leftarrow K_{41} + k_{31}^2 \\ K_{43} \leftarrow K_{43} + k_{32}^2 \\ K_{42} \leftarrow K_{42} + k_{34}^2 \end{cases} \quad f_4 \leftarrow f_3^2
\end{aligned}$$

$$EQ(4,2) = 2 \begin{cases} K_{22} \leftarrow K_{22} + k_{44}^2 \\ K_{21} \leftarrow K_{21} + k_{41}^2 \\ K_{23} \leftarrow K_{23} + k_{42}^2 \\ K_{24} \leftarrow K_{24} + k_{43}^2 \end{cases} \quad f_2 \leftarrow f_4^2.$$

Thus, we obtain the following algebraic system of equations

$$\mathbf{K} = \begin{bmatrix} k_{22}^1 + k_{11}^2 & k_{23}^1 + k_{14}^2 & k_{12}^2 & k_{13}^2 \\ k_{32}^1 + k_{41}^2 & k_{33}^1 + k_{44}^2 & k_{42}^2 & k_{43}^2 \\ k_{21}^2 & k_{24}^2 & k_{22}^2 & k_{23}^2 \\ k_{31}^2 & k_{34}^2 & k_{32}^2 & k_{33}^2 \end{bmatrix} \quad \underline{f} = \begin{bmatrix} f_2^1 + f_1^2 \\ f_3^1 + f_4^2 \\ f_2^2 \\ f_2^3 \end{bmatrix}.$$

Comparing the result with Fig. 2.18, we can set up the following rules

- **Diagonal entries**:
  K_{11} is the entry of $\mathbf{K}$ due to node 3. Since this node belongs to element 1 as well as 2, the entry in $\mathbf{K}$ will be a sum of the contributions from element 1 and element 2. The entry K_{33} is due to global node 5, and therefore, only element 2 will contribute to it.

- **Off-diagonal entries**:
  The same situation as described for the diagonal entries occurs.

Figure 2.19 summarizes the whole assembly procedure, and the following pseudo-code demonstrates the computer implementation.

```
//loop over all finite elements
for e := 1, n_e
    //loop over all element nodes (equations)
    for a := 1, n_en
        eq1 := EQ(a, e)
        if (eq1 > 0)
            //loop over all element nodes (equations)
            for b := 1, n_en
                eq2 := EQ(b, e)
                if (eq2 > 0)
                    K(eq1, eq2) := K(eq1, eq2) + k^e(a, b)
                end
            end
        end
    end
end
```

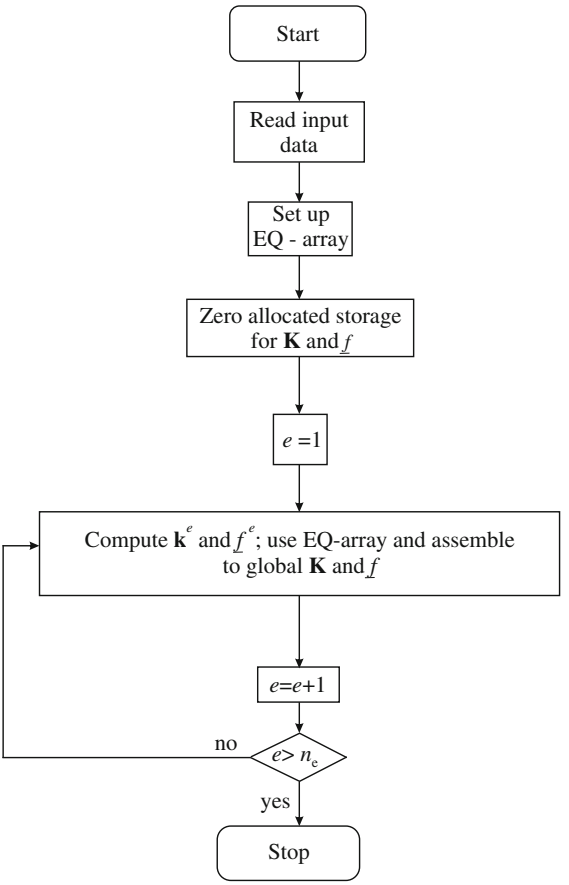


Fig. 2.19 FE procedure

2.5 Time Discretization

In the following we will describe single time step discretization algorithms for parabolic (e.g., electromagnetic field in the eddy current case) as well as for hyperbolic (mechanical as well as acoustic field) partial differential equations.

2.5.1 Parabolic Differential Equation

Let us consider the following *semi-discrete Galerkin formulation* (discrete in space and continuous in time)

$$\mathbf{M}\dot{\underline{u}}(t) + \mathbf{K}\underline{u}(t) = \underline{f}(t), \tag{2.64}$$

Table 2.2 Finite difference schemes

γ_P	Method
0.0	Forward difference; forward Euler
0.5	Trapezoidal rule; Crank-Nicolson
1.0	Backward difference; backward Euler

with $\mathbf{M}$ the mass matrix, $\mathbf{K}$ the stiffness matrix, $\underline{f}$ the right-hand side, $\underline{u}$ the vector of unknowns at the finite element nodes and $\dot{\underline{u}}$ its derivative with respect to time. In a first step, let us subdivide the time interval $[0, T]$ into M sub-intervals, which are defined as follows

$$[0, T] = \bigcup_{n=1}^{M} [t_{n-1}, t_n] \ , \quad t_0 = 0 < t_1 < t_2 < .... < t_n < .. < t_M = T.$$

For simplicity, we assume the time step size $\Delta t = t_n - t_{n-1} = T/M$ to be constant over the whole time interval of interest. In a second step, we approximate the time derivative of the unknown $\underline{u}(t)$ by the forward time difference scheme

$$\dot{\underline{u}}(t) \approx \frac{\underline{u}(t_{n+1}) - \underline{u}(t_n)}{\Delta t} = \frac{\underline{u}_{n+1} - \underline{u}_n}{\Delta t}. \tag{2.65}$$

In the third step, we have to decide at which point in time we evaluate the elliptic part $\mathbf{K}\underline{u}(t)$ and the right-hand side $\underline{f}(t)$. Applying a convex linear combination with the time integration parameter γ_P to these terms, will result in the following system of equations

$$\mathbf{M}\,\frac{\underline{u}(t_{n+1}) - \underline{u}(t_n)}{\Delta t} + \gamma_P \mathbf{K}\underline{u}_{n+1} + (1-\gamma_P)\mathbf{K}\underline{u}_n = \gamma_P \underline{f}_{n+1} + (1-\gamma_P)\underline{f}_n. \tag{2.66}$$

Table 2.2 is listing the different time discretization methods depending on the value of γ_P.

Rearranging the terms in (2.66), we arrive at

$$(\mathbf{M} + \gamma_P \Delta t \mathbf{K})\,\underline{u}_{n+1} = \Delta t \left(\gamma_P \underline{f}_{n+1} + (1-\gamma_P)\underline{f}_n\right) + (\mathbf{M} - (1-\gamma_P)\Delta t \mathbf{K})\,\underline{u}_n. \tag{2.67}$$

We obtain the same system of algebraic equations, when we apply the general trapezoidal difference scheme [4], which is defined as follows

$$\underline{u}_{n+1} = \underline{u}_n + \Delta t \ \left((1-\gamma_P)\dot{\underline{u}}_n + \gamma_P \dot{\underline{u}}_{n+1}\right). \tag{2.68}$$

From (2.68) we can compute $\dot{\underline{u}}_{n+1}$

$$\dot{\underline{u}}_{n+1} = \frac{\underline{u}_{n+1} - \underline{u}_n}{\gamma_P \Delta t} - \frac{1-\gamma_P}{\gamma_P}\,\dot{\underline{u}}_n$$

and substituting it into the time discrete version of (2.64) at $t = t_{n+1}$ leads to

$$(\mathbf{M} + \gamma_P \Delta t \mathbf{K})\, \underline{u}_{n+1} = \gamma_P \Delta t\, \underline{f}_{n+1} + \mathbf{M}\left(\underline{u}_n + (1-\gamma_P)\Delta t \underline{\dot{u}}_n\right). \tag{2.69}$$

To see that (2.69) and (2.67) are equivalent, we use the relation

$$\mathbf{M}\underline{\dot{u}}_n + \mathbf{K}\underline{u}_n = \underline{f}_n$$

and rewrite the right-hand side of (2.67) as follows

$$\begin{aligned}
&\Delta t \left(\gamma_P \underline{f}_{n+1} + (1-\gamma_P)\underline{f}_n\right) + (\mathbf{M} - (1-\gamma_P)\Delta t \mathbf{K})\, \underline{u}_n \\
&\quad = \gamma_P \Delta t\, \underline{f}_{n+1} + \mathbf{M}\underline{u}_n + (1-\gamma_P)\Delta t \underbrace{\left(\underline{f}_n - \mathbf{K}\underline{u}_n\right)}_{\mathbf{M}\underline{\dot{u}}_n} \\
&\quad = \gamma_P \Delta t\, \underline{f}_{n+1} + \mathbf{M}\left(\underline{u}_n + (1-\gamma_P)\Delta t \underline{\dot{u}}_n\right).
\end{aligned}$$

Applying the general trapezoidal difference method, we can write the solution process as a *predictor-corrector* algorithm, where we distinguish between an *effective mass* and *effective stiffness* formulation.

1. *Effective Mass Formulation*

   - Perform predictor step:

$$\tilde{\underline{u}} = \underline{u}_n + (1-\gamma_P)\Delta t\, \underline{\dot{u}}_n. \tag{2.70}$$

   - Solve algebraic system of equations:

$$\begin{aligned}
\boldsymbol{M}^* \underline{\dot{u}}_{n+1} &= \underline{f}_{n+1} - \boldsymbol{K}\tilde{\underline{u}} \\
\boldsymbol{M}^* &= \boldsymbol{M} + \gamma_P \Delta t\, \boldsymbol{K}.
\end{aligned} \tag{2.71}$$

   - Perform corrector step:

$$\underline{u}_{n+1} = \tilde{\underline{u}} + \gamma_P \Delta t\, \underline{\dot{u}}_{n+1}. \tag{2.72}$$

2. *Effective Stiffness Formulation*

   - Perform predictor step:

$$\tilde{\underline{u}} = \underline{u}_n + (1-\gamma_P)\Delta t\, \underline{\dot{u}}_n. \tag{2.73}$$

- Solve algebraic system of equations:

$$\boldsymbol{K}^* \underline{u}_{n+1} = \underline{f}_{n+1} + \frac{1}{\gamma_P \Delta t} \boldsymbol{M} \tilde{\underline{u}} \tag{2.74}$$

$$\boldsymbol{K}^* = \boldsymbol{K} + \frac{1}{\gamma_P \Delta t} \boldsymbol{M}.$$

- Perform corrector step:

$$\dot{\underline{u}}_{n+1} = \frac{\underline{u}_{n+1} - \tilde{\underline{u}}}{\gamma_P \Delta t}. \tag{2.75}$$

According to the choice of the integration parameter γ_P, one distinguishes between an *explicit* and *implicit* algorithm.

1. *Explicit Algorithm*: $\gamma_P = 0$
   For this case, $\mathbf{M}^* = \mathbf{M}$ and lumping of the mass matrix might be considered (see e.g., [4]). Thus, the system matrix of the algebraic system is a diagonal matrix, and the only time-consuming part of the solution process are the matrix vector multiplications on the right-hand side. The disadvantage of this algorithm is clearly the lack of stability, which implies restrictions on the time step value Δt, depending on the material parameters and the quality of the mesh. In addition, one has to consider that the mass matrix must be regular. This is, for example, not the case within electromagnetic computations, where the entries in the mass matrix are zero for regions with zero electrical conductivity (e.g., air). A solution to this problem may be the application of a mixed explicit/implicit time discretization [4].
2. *Implicit Algorithm*: $0 < \gamma_P \leq 1$
   In this case, $\mathbf{M}^*$ computes as the sum of the mass matrix $\mathbf{M}$ and the stiffness matrix $\mathbf{K}$ multiplied by $\gamma_P \Delta t$. Now $\mathbf{M}^*$ is a sparse matrix and the algebraic system of equations has to be solved by any direct or iterative solver. For $\gamma_P = 0.5$ the time discretization is called the *Crank-Nicolson* scheme (second-order accurate), and for $\gamma_P \geq 0.5$ the algorithm is absolute stable (independently of Δt).

In order to investigate in the accuracy of the time discretization scheme, we will expand $\underline{u}(t)$ at time $t = t_{k+\gamma_P}$ in a Taylor series

$$\underline{u}(t) = \underline{u}(t_{k+\gamma_P}) + \dot{\underline{u}}(t_{k+\gamma_P})\,(t - t_{k+\gamma_P}) + \ddot{\underline{u}}(t_{k+\gamma_P})\,\frac{(t - t_{k+\gamma_P})^2}{2} + O\left((t - t_{k+\gamma_P})^3\right). \tag{2.76}$$

By considering the relation $t_{k+\gamma_P} = t_k + \gamma_P \Delta t$ we obtain for the evaluation of (2.76) at $t = t_k$

$$\underline{u}(t_k) = \underline{u}(t_{k+\gamma_\mathrm{P}}) + \dot{\underline{u}}(t_{k+\gamma_\mathrm{P}})\,(-\gamma_\mathrm{P}\Delta t) + \ddot{\underline{u}}(t_{k+\gamma_\mathrm{P}})\,\frac{(-\gamma_\mathrm{P}\Delta t)^2}{2} + O\left((-\gamma_\mathrm{P}\Delta t)^3\right)$$

as well as at $t = t_{k+1}$

$$\underline{u}(t_{k+1}) = \underline{u}(t_{k+\gamma_\mathrm{P}}) + \dot{\underline{u}}(t_{k+\gamma_\mathrm{P}})\,((1-\gamma_\mathrm{P})\Delta t) + \ddot{\underline{u}}(t_{k+\gamma_\mathrm{P}})\,\frac{((1-\gamma_\mathrm{P})\Delta t)^2}{2} + O\left(((1-\gamma_\mathrm{P})\Delta t)^3\right).$$

Therewith, we obtain for the difference of $\underline{u}(t_{k+1})$ and $\underline{u}(t_k)$, which is used for approximating the time derivative (see (2.65)), the following expression

$$\underline{u}(t_{k+1}) - \underline{u}(t_k) = \dot{\underline{u}}\,\Delta t + \ddot{\underline{u}}\,\frac{\Delta t^2 - 2\gamma_\mathrm{P}\Delta t^2}{2} + O(\Delta t^3). \tag{2.77}$$

Now, (2.77) clearly shows that only for $\gamma_\mathrm{P} = 0.5$ (Crank-Nicolson), we arrive at a second order time discretization scheme, and for all other choices the scheme is first-order accurate.

2.5.2 Hyperbolic Differential Equation

For the hyperbolic case, we arrive after the spatial discretization at the following system of second-order ordinary differential equations (still continuous in time)

$$\mathbf{M}\ddot{\underline{u}}(t) + \mathbf{K}\underline{u}(t) = \underline{f}(t). \tag{2.78}$$

Now, we will apply a finite difference scheme of second order to approximate $\ddot{\underline{u}}(t)$

$$\ddot{\underline{u}}(t) \approx \frac{\underline{u}(t_{n+1}) - 2\underline{u}(t_n) + \underline{u}(t_{n-1})}{\Delta t^2} = \frac{\underline{u}_{n+1} - 2\underline{u}_n + \underline{u}_{n-1}}{\Delta t^2} \tag{2.79}$$

with Δt the time step size. Similar to the parabolic case, we substitute the elliptic part $\boldsymbol{K}\underline{u}(t)$ and the right-hand side $\underline{f}(t)$ by a convex, linear combination at $t = t_{n-1}$, t_n and t_{n+1} weighted by the integration parameter α_H. Therewith, we obtain the following algebraic system of equations

$$\boldsymbol{M}\,\frac{\underline{u}_{n+1} - 2\underline{u}_n + \underline{u}_{n-1}}{\Delta t^2} + \alpha_\mathrm{H}\boldsymbol{K}\underline{u}_{n+1} + (1-2\alpha_\mathrm{H})\boldsymbol{K}\underline{u}_n + \alpha_\mathrm{H}\boldsymbol{K}\underline{u}_{n-1} = \alpha_\mathrm{H}\underline{f}_{n+1} + (1-2\alpha_\mathrm{H})\underline{f}_n + \alpha_\mathrm{H}\underline{f}_{n-1}, \tag{2.80}$$

which can be rewritten as

$$\begin{aligned}(\boldsymbol{M}+\alpha_{\mathrm{H}}\Delta t^2\boldsymbol{K})\underline{u}_{n+1} &= \alpha_{\mathrm{H}}\Delta t^2\underline{f}_{n+1}+(1-2\alpha_{\mathrm{H}})\Delta t^2\underline{f}_n+\alpha_{\mathrm{H}}\Delta t^2\underline{f}_{n-1}\\ &\quad+(2\boldsymbol{M}-(1-2\alpha_{\mathrm{H}})\Delta t^2\boldsymbol{K})\underline{u}_n\\ &\quad-(\boldsymbol{M}+\alpha_{\mathrm{H}}\Delta t^2\boldsymbol{K})\underline{u}_{n-1}.\end{aligned} \tag{2.81}$$

Choosing $\alpha_{\mathrm{H}}=0$ and transforming $\boldsymbol{M}$ to a diagonal matrix (mass lumping, see e.g., [4]), results in an explicit time discretization scheme, which does not need any algebraic solver. However, such schemes will not be unconditionally stable. A stability analysis will exhibit the so-called CFL (Courant-Friedrich-Levi) condition [6]

$$\Delta t<\frac{2}{\sqrt{\lambda_{\max}(\boldsymbol{M}^{-1}\boldsymbol{K})}}. \tag{2.82}$$

In (2.82) $\lambda_{\max}$ denotes the largest eigenvalue of the matrix $(\boldsymbol{M}^{-1}\boldsymbol{K})$. Since this value is of the order $O(h^{-2})$, we should choose the time step size Δt and the mesh size h of the same order. For $\alpha_{\mathrm{H}}=1/4$ the above scheme is unconditional stable.

For practical applications, especially if an additional damping matrix $\boldsymbol{C}$ is present, the *Newmark* schemes are mainly used. Let us start at the *semi-discrete Galerkin formulation*

$$\boldsymbol{M}\ddot{\underline{u}}_{n+1}+\boldsymbol{C}\dot{\underline{u}}_{n+1}+\boldsymbol{K}\underline{u}_{n+1}=\underline{f}_{n+1}, \tag{2.83}$$

with $\boldsymbol{M}$ the mass matrix, $\boldsymbol{C}$ the damping matrix (e.g., $\boldsymbol{C}=\alpha_{\mathrm{M}}\boldsymbol{M}+\alpha_{\mathrm{K}}\boldsymbol{K}$, see Sect. 3.7.2), $\boldsymbol{K}$ the stiffness matrix, $\underline{f}$ the right-hand side, $\underline{u}$ the vector of unknowns at the finite element nodes and $\dot{\underline{u}}$ as well as $\ddot{\underline{u}}$ its first and second derivative with respect to time. Thus, we have (see e.g., [4])

$$\underline{u}_{n+1}=\underline{u}_n+\Delta t\,\dot{\underline{u}}_n+\frac{\Delta t^2}{2}\left((1-2\beta_{\mathrm{H}})\ddot{\underline{u}}_n+2\beta_{\mathrm{H}}\ddot{\underline{u}}_{n+1}\right) \tag{2.84}$$

$$\dot{\underline{u}}_{n+1}=\dot{\underline{u}}_n+\Delta t\left((1-\gamma_{\mathrm{H}})\ddot{\underline{u}}_n+\gamma_{\mathrm{H}}\ddot{\underline{u}}_{n+1}\right). \tag{2.85}$$

In (2.84) and (2.85) n denotes the time step counter, Δt the time step value and β_{H}, γ_{H} the integration parameters. Substituting $\underline{u}_{n+1}$ and $\dot{\underline{u}}_{n+1}$ according to (2.84) and (2.85) into (2.83) leads to the following algebraic system of equations

$$\begin{aligned}\boldsymbol{M}^*\ddot{\underline{u}}_{n+1} &= \underline{f}_{n+1}-\boldsymbol{C}\left(\dot{\underline{u}}_n+(1-\gamma_{\mathrm{H}})\Delta t\,\ddot{\underline{u}}_n\right)\\ &\quad-\boldsymbol{K}\left(\underline{u}_n+\Delta t\,\dot{\underline{u}}_n+\frac{\Delta t^2}{2}(1-2\beta_{\mathrm{H}})\ddot{\underline{u}}_n\right)\\ \boldsymbol{M}^* &= \boldsymbol{M}+\gamma_{\mathrm{H}}\Delta t\,\boldsymbol{C}+\beta_{\mathrm{H}}\Delta t^2\,\boldsymbol{K}.\end{aligned} \tag{2.86}$$

According to the choice of the integration parameters β_{H} and γ_{H}, one distinguishes similarly to the parabolic case between an *explicit* and *implicit* algorithm.

1. *Explicit Algorithm*: $\beta_H = 0$ and $\boldsymbol{C} = \alpha_M \boldsymbol{M}$
   Choosing for γ_H the value 0.5 we achieve a second-order accurate scheme. Similar to the parabolic case, it makes sense to lump the system matrix, so that again the time-consuming part of the solution process is the matrix vector multiplications on the right-hand side. Of course, the stability depends on the time step value Δt, the material parameters and the quality of the mesh. However, this kind of algorithm is used quite often in acoustic computations, especially when the discretization is performed by a mapped mesh.
2. *Implicit Algorithm*: $\beta_H \neq 0$, $\gamma_H \neq 0$
   In this case, $\boldsymbol{M}^*$ computes as the sum of the mass matrix $\boldsymbol{M}$, the damping matrix $\boldsymbol{C}$ and the stiffness matrix $\boldsymbol{K}$ with appropriate integration factors. The matrix $\boldsymbol{M}^*$ is now sparse and the algebraic system of equations has to be solved by any direct or iterative solver. For $\beta_H = 0.25$ and $\gamma_H = 0.5$ the time discretization is second-order accurate with respect to time, if $\boldsymbol{C}$ vanishes. To keep the second-order accuracy even in the damped case, one has to extend the Newmark scheme to the Hilbert-Hughes-Taylor scheme (see [4]).

Writing the solution process for one time step as a *predictor-corrector* algorithm we arrive at the *effective mass* as well as *effective stiffness* formulations.

1. *Effective Mass Formulation*

   - Perform predictor step:

$$\tilde{\underline{u}} = \underline{u}_n + \Delta t\, \underline{\dot{u}}_n + (1 - 2\beta_H)\frac{\Delta t^2}{2}\underline{\ddot{u}}_n \tag{2.87}$$

$$\tilde{\underline{\dot{u}}} = \underline{\dot{u}}_n + \Delta t\, (1 - \gamma_H)\underline{\ddot{u}}_n. \tag{2.88}$$

   - Solve algebraic system of equations:

$$\boldsymbol{M}^* \underline{\ddot{u}}_{n+1} = \underline{f}_{n+1} - \boldsymbol{K}\tilde{\underline{u}} - \boldsymbol{C}\tilde{\underline{\dot{u}}} \tag{2.89}$$

$$\boldsymbol{M}^* = \boldsymbol{M} + \gamma_H \Delta t\, \boldsymbol{C} + \beta \Delta t^2\, \boldsymbol{K}.$$

   - Perform corrector step:

$$\underline{u}_{n+1} = \tilde{\underline{u}} + \beta_H \Delta t^2\, \underline{\ddot{u}}_{n+1} \tag{2.90}$$

$$\underline{\dot{u}}_{n+1} = \tilde{\underline{\dot{u}}} + \gamma_H \Delta t\, \underline{\ddot{u}}_{n+1}. \tag{2.91}$$

2. *Effective Stiffness Formulation*
   According to (2.84) and (2.85) we can express $\underline{\ddot{u}}_{n+1}$ and $\underline{\dot{u}}_{n+1}$ as follows

$$\underline{\ddot{u}}_{n+1} = \frac{\underline{u}_{n+1} - \tilde{\underline{u}}}{\beta_H \Delta t^2} \tag{2.92}$$

$$\underline{\dot{u}}_{n+1} = \tilde{\underline{\dot{u}}} + \gamma_H \Delta t\, \underline{\ddot{u}}_{n+1} = \tilde{\underline{\dot{u}}} + \frac{\gamma_H}{\beta_H \Delta t}\, (\underline{u}_{n+1} - \tilde{\underline{u}}). \tag{2.93}$$

Therefore, we obtain

- Perform predictor step:

$$\tilde{\underline{u}} = \underline{u}_n + \Delta t\, \dot{\underline{u}}_n + (1 - 2\beta_{\mathrm{H}}) \frac{\Delta t^2}{2} \ddot{\underline{u}}_n \tag{2.94}$$

$$\tilde{\dot{\underline{u}}} = \dot{\underline{u}}_n + \Delta t\, (1 - \gamma_{\mathrm{H}}) \ddot{\underline{u}}_n. \tag{2.95}$$

- Solve algebraic system of equations:

$$\boldsymbol{K}^* \underline{u}_{n+1} = \underline{f}_{n+1} - \boldsymbol{C} \tilde{\underline{\dot{u}}} + \left(\frac{1}{\beta_{\mathrm{H}} \Delta t^2} \boldsymbol{M} + \frac{\gamma_{\mathrm{H}}}{\beta_{\mathrm{H}} \Delta t} \boldsymbol{C} \right) \tilde{\underline{u}} \tag{2.96}$$

$$\boldsymbol{K}^* = \boldsymbol{K} + \frac{\gamma_{\mathrm{H}}}{\beta_{\mathrm{H}} \Delta t} \boldsymbol{C} + \frac{1}{\beta_{\mathrm{H}} \Delta t^2} \boldsymbol{M}.$$

- Perform corrector step:

$$\ddot{\underline{u}}_{n+1} = \frac{\underline{u}_{n+1} - \tilde{\underline{u}}}{\beta_{\mathrm{H}} \Delta t^2} \tag{2.97}$$

$$\dot{\underline{u}}_{n+1} = \tilde{\dot{\underline{u}}} + \gamma_{\mathrm{H}} \Delta t \ddot{\underline{u}}_{n+1}. \tag{2.98}$$

2.6 Integration over Surfaces

Very often one has to evaluate an integral over a 3D surface or along a contour in 2D. In the first case the integration is performed within a 2D finite element in a 3D space and in the second case within a 1D finite element in a 2D space.

Let us assume a scalar function $f(x, y, z)$ as the integrand of a surface integral. According to [7], we can write

$$\int_{\Gamma} f(x, y, z)\ \mathrm{d}\Gamma = \int \int f(x(\xi, \eta), y(\xi, \eta), z(\xi, \eta))\ |\boldsymbol{x}_\xi(\xi, \eta) \times \boldsymbol{x}_\eta(\xi, \eta)|\ \mathrm{d}\xi\ \mathrm{d}\eta \tag{2.99}$$

with

$$\boldsymbol{x}_\xi = \begin{pmatrix} \frac{\partial x}{\partial \xi} \\ \frac{\partial y}{\partial \xi} \\ \frac{\partial z}{\partial \xi} \end{pmatrix} \qquad \boldsymbol{x}_\eta = \begin{pmatrix} \frac{\partial x}{\partial \eta} \\ \frac{\partial y}{\partial \eta} \\ \frac{\partial z}{\partial \eta} \end{pmatrix}.$$

For the second case of a contour integral over a scalar function $f(x, y)$, we obtain

$$\int_{C} f(x, y)\ \mathrm{d}s = \int f(x(\xi), y(\xi))\ |\boldsymbol{x}_\xi(\xi)|\ \mathrm{d}\xi, \tag{2.100}$$

with

$$\boldsymbol{x}_\xi = \begin{pmatrix} \frac{\partial x}{\partial \xi} \\ \frac{\partial y}{\partial \xi} \end{pmatrix}.$$

Therefore, with slight modifications we can evaluate such integrals by performing an integration in the local domain and instead of the Jacobian determinant we have to compute the expressions given in (2.99) and (2.100).

2.7 Edge Nédélec Finite Elements

Edge finite elements belong to the family of vector finite elements (shape functions are vectors) and assign the degrees of freedom to the edges rather than to the nodes of the element. These types of elements were first introduced by Whitney (see e.g., [8]). Their importance in electromagnetics were realized by Nédélec (see e.g., [9]), who constructed edge elements on quadrilateral and tetrahedron elements. Many important studies followed for the further development of different electromagnetic field problems (see e.g., [10–14]).

Within edge elements, a physical vector quantity $\boldsymbol{A}$ (e.g., the magnetic vector potential) is approximated by the following ansatz

$$\boldsymbol{A} \approx \boldsymbol{A}^h = \sum_{k=1}^{n_\mathrm{e}} A_k^e \boldsymbol{N}_k. \tag{2.101}$$

In (2.101) n_e defines the number of edges in the finite element mesh, $\boldsymbol{N}_k$ the edge shape function associated with the k-th edge, and A_k^e the corresponding degree of freedom, namely the line integral of the physical vector quantity along the k-th edge

$$A_k^e = \int_k \boldsymbol{A} \cdot \mathrm{d}\mathbf{s}. \tag{2.102}$$

For edge shape functions $\boldsymbol{N}_k$ of lowest order, the following conditions have to be fulfilled [15]:

1. The tangential component of $\boldsymbol{N}_k$ along the edge k has to be constant.
2. The tangential component of $\boldsymbol{N}_k$ along edges $l \neq k$ is zero.
3. The divergence of $\boldsymbol{N}_k$ is zero inside the element.

Since, in this work, Nédélec finite elements are exclusively used for 3D magnetic field computation, we will restrict ourselves to this case using tetrahedron elements. Let us consider the following vector ansatz function $\boldsymbol{N}_1$ (∇ defines here the derivatives with respect to the global coordinates $\boldsymbol{x}$)

$$\boldsymbol{N}_1 = N_1 \nabla N_2 - N_2 \nabla N_1 \tag{2.103}$$

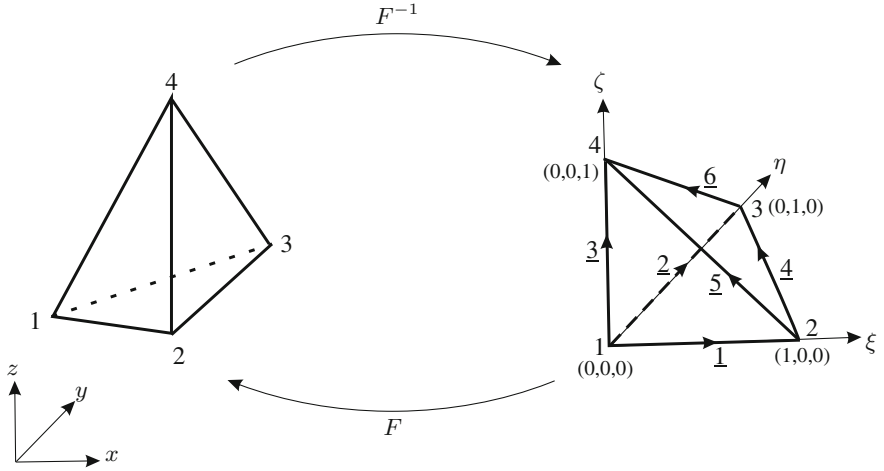


Fig. 2.20 Tetrahedron element: global and local domain with orientation of the edges

along edge 1 defined by the nodes 1 and 2 (see Fig. 2.20) and N_1 and N_2 the nodal interpolation functions (see Sect. 2.3.4) in node 1 and 2, respectively. To check condition 1, we compute, e.g., the tangential component of $\boldsymbol{N}_1$ along edge 1

$$\boldsymbol{N}_1 \cdot \boldsymbol{t}_1 = N_1 \boldsymbol{t}_1 \cdot \nabla N_2 - N_2 \boldsymbol{t}_1 \cdot \nabla N_1.$$

It has to be mentioned that we have to compute the derivatives with respect to the global coordinate system (see Sect. 2.3.8). Since $\boldsymbol{t}_1 \cdot \nabla N_1 = -1/(2l_1)$ and $\boldsymbol{t}_1 \cdot \nabla N_2 = 1/(2l_1)$ (l_1 denotes the length of edge 1), we obtain

$$\boldsymbol{N}_1 \cdot \boldsymbol{t}_1 = \frac{1}{l_1}.$$

Condition 2 is also fulfilled, since N_1 vanishes along the edges (4,5,6), N_2 along the edges (2,3,6) and $\boldsymbol{t}_2 \cdot \nabla N_1$, $\boldsymbol{t}_3 \cdot \nabla N_1$, $\boldsymbol{t}_4 \cdot \nabla N_2$, $\boldsymbol{t}_5 \cdot \nabla N_2$ are zero. Therewith, shape function $\boldsymbol{N}_1$ has no tangential component along the edges (2,3,4,5,6).

Condition number 3 states that the divergence of $\boldsymbol{N}_1$ has to vanish inside the element. Applying the divergence to $\boldsymbol{N}_1$ results in

$$\begin{aligned} \nabla \cdot (N_1 \nabla N_2 - N_2 \nabla N_1) &= N_1 \boldsymbol{\Delta} N_2 + \nabla N_1 \cdot \nabla N_2 \\ &\quad - N_2 \boldsymbol{\Delta} N_1 - \nabla N_2 \cdot \nabla N_1 \\ &= N_1 \boldsymbol{\Delta} N_2 - N_2 \boldsymbol{\Delta} N_1. \end{aligned}$$

Since the interpolation functions N_1 as well as N_2 are linear functions, the value of $\nabla \cdot \boldsymbol{N}_1$ is zero.

To obtain for the tangential components of $\boldsymbol{N}_k$ the value 1, we have to scale the vector function with the length of the corresponding edge length, which results in the following edge interpolation functions for a linear tetrahedron

$$\begin{aligned}
\boldsymbol{N}_1 &= (N_1 \nabla N_2 - N_2 \nabla N_1)\, l_1 \\
\boldsymbol{N}_2 &= (N_1 \nabla N_3 - N_3 \nabla N_1)\, l_2 \\
\boldsymbol{N}_3 &= (N_1 \nabla N_4 - N_4 \nabla N_1)\, l_3 \\
\boldsymbol{N}_4 &= (N_2 \nabla N_3 - N_3 \nabla N_2)\, l_4 \\
\boldsymbol{N}_5 &= (N_4 \nabla N_2 - N_2 \nabla N_4)\, l_5 \\
\boldsymbol{N}_6 &= (N_3 \nabla N_4 - N_4 \nabla N_3)\, l_6.
\end{aligned}$$

These FE basis functions defines the lowest order Nédélec functions for a tetrahedron, which are of polynomial order $p = 0$, since they have a constant tangential component along each edge. Higher order elements are discussed in Sect. 6.7.6.

2.8 Discretization Error

When applying the FE method, it is of great importance to have some knowledge of the discretization error. Since the error analysis is a quite sophisticated task, we will just provide a short overview containing important results. For a detailed discussion on this topic we refer to [6].

For our purpose of error analysis let us consider the following variational form:

Given:
$$f, c : \Omega \to \mathbb{R}$$

Find: $u \in V_g$ such that for all $v \in V_0$

$$a(u, v) = < f, v > \tag{2.104}$$

with

$$\begin{aligned}
a(u, v) &= \int_{\Omega} \nabla v \cdot \nabla u \, \mathrm{d}\Omega + \int_{\Omega} c\, v\, u \, \mathrm{d}\Omega \\
< f, v > &= \int_{\Omega} v f \, \mathrm{d}\Omega \\
V_g &= \{u \in H^1 | u = g \text{ on } \Gamma\} \\
V_0 &= \{v \in H^1 | v = 0 \text{ on } \Gamma\}.
\end{aligned}$$

Applying the FE method to our problem, results in

Given:
$$f, c : \Omega \to \mathbb{R}$$

Find: $u^h \in V_g^h$ such that for all $v^h \in V_0^h$

$$a(u^h, v^h) = < f, v^h > \tag{2.105}$$

with

$$a(u^h, v^h) = \int_\Omega \nabla v^h \cdot \nabla u^h \, \mathrm{d}\Omega + \int_\Omega c\, v^h\, u^h \, \mathrm{d}\Omega$$
$$< f, v^h > = \int_\Omega v^h f \, \mathrm{d}\Omega$$

with $V_g^h \subset V_g$ and $V_0^h \subset V_0$.
In general, we distinguish between *a priori* and *a posteriori* error estimates:

- The a priori error estimate is expressed as

$$||u - u^h|| \leq C_1(u)\, h^\alpha \tag{2.106}$$

  with $\alpha > 0$ and $C_1(u)$ being a positive constant. In (2.106) u denotes the exact solution, u^h our solution obtained by the FE method, h the discretization parameter (e.g., longest edge in the FE mesh) and $||\cdot||$ an adequate norm. The constant α depends on the smoothness of the solution u and the polynomial degree of the chosen FE basis (shape) functions. In general, this constant is not known.
- The a posteriori error estimate reads as

$$||u - u^h|| \leq C_2(u^h, h). \tag{2.107}$$

  As for the a priori error estimate, u denotes the exact solution, u^h our solution obtained by the FE method, h the discretization parameter and $||\cdot||$ an adequate norm. The constant C_2 depends on the FE solution and the discretization parameter h. Providing an optimal error estimator, it is possible to obtain C_2 or at least sharp bounds for C_2 (see e.g., [16]).

In the following, we will concentrate on a priori estimates, and we assume that our bilinear form as defined in (2.105) is V_0-elliptic and V_0-bounded (for a proof see e.g., [6]). V_0-ellipticity means that there exists a constant $C_1 > 0$ so that

$$a(v, v) \geq C_1 ||v||_{H^1}^2$$

for all $v \in V_0$. V_0-boundedness states that there exists a constant $C_2 > 0$, such that

$$|a(u, v)| \leq C_2 ||u||_{H^1} \, ||v||_{H^1}$$

for all $u, v \in V_0$.

In addition to the norms defined in Appendix D, we introduce the energy norm of the error $e = (u - u^h)$ associated with the bilinear form $a(u, v)$

$$||u - u^h||_a = \sqrt{a(u - u^h, u - u^h)} = \sqrt{a(e, e)}.$$

Since $V_0^h \subset V_0$, we may rewrite (2.104) by

$$a(u, v^h) \; = < f, v^h >$$

for all $v^h \in V_0^h$. Now, let us subtract from this equation the discrete weak form (see (2.105)). This operation results in

$$a(u - u^h, v^h) = 0$$

for all $v^h \in V_0^h$, which is known as the *Galerkin orthogonality* of the error. This result implies that in a certain sense u^h is the best approximation of u in V_g^h.

The most important theorems for the a priori estimation of the discretization error in $H^1(\Omega)$ are *Céa*'s lemma (see Theorem 1) and the Bramble-Hilbert lemma [17]. The lemma of Céa allows us to transform the estimation of the discretization error to an estimation of the approximation error [6].

Theorem 1 *Let the bilinear form $a(\cdot, \cdot)$ be V_0-elliptic and V_0-bounded. Then, the error estimation reads as*

$$||u - u^h||_{H^1} \leq \frac{C_1}{C_2} \inf_{w^h \in V_g^h} ||u - w^h||_{H^1}. \tag{2.108}$$

Since the approximation error

$$\inf_{w^h \in V_g^h} ||u - w^h||_{H^1}$$

can be estimated from above by the interpolant (see Fig. 2.21)

$$\Pi^h(u) = \sum_{i=0}^{M} N_i(x) \, u_i \; \subset V_g^h,$$

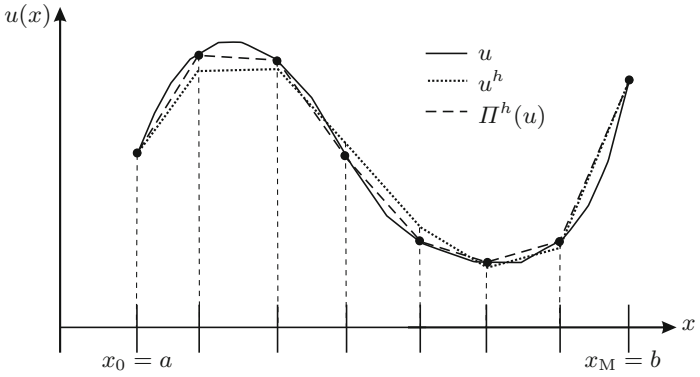


Fig. 2.21 Exact solution u, interpolant $\Pi^h(u)$ and the FE solution u^h

we may rewrite (2.108) by

$$||u - u^h||_{H^1} \leq \frac{C_1}{C_2} \inf_{w^h \in V_g^h} ||u - w^h||_{H^1} \leq \frac{C_1}{C_2} ||u - \Pi^h(u)||_{H^1}.$$

According to this important result and using the Bramble-Hilbert lemma [17], we will obtain practical a priori error estimates. For our FE discretization we will use piecewise, continuous polynomial basis functions of order m, and can state the following theorem [4]:

Theorem 2 *Let $a(\cdot,\cdot)$ be a V_0-elliptic and V_0-bounded bilinear form. Then, for all $u \in H^r$ with $r \geq 0$, there exists a discrete solution $u^h \in V_g^h$ and a constant C indepent of u and the discretization parameter h, such that*

$$||u - u^h||_{H^m} \leq C\, h^{\alpha} ||u||_{H^r}, \tag{2.109}$$

where $\alpha = \min(k+1-m, r-m)$ is the rate of convergence. Thereby, m is the order of the highest derivative in the bilinear form $a(\cdot,\cdot)$, and k is the order of the FE basis functions. For lower H^s norms, $0 \leq s \leq m$, an error estimate can be developed by the Aubin-Nitsche method (see, e.g., [18]) resulting in

$$||u - u^h||_{H^s} \leq C\, h^{\beta} ||u||_{H^r} \tag{2.110}$$

with C being independent of u and h, and $\beta = \min(k+1-s, 2(k+1-m))$.

For our considered bilinear form (see (2.104)), the exact solution u is in H^2 ($r = 2$), and the order of the highest derivative in the bilinear form is $m = 1$. Using linear FE basis functions for which $k = 1$, we arrive at the following practical estimates using (2.109) and (2.110)

$$||u - u^h||_{H^1} \leq C\, h |u|_{H^2} = O(h) \tag{2.111}$$
$$||u - u^h||_{L_2} \leq \tilde{C}\, h^2 |u|_{H^2} = O(h^2) \tag{2.112}$$

with $C, \tilde{C}$ being constants and independent of the discretization. However, problems with discontinuous jumps in material parameters (e.g., magnetic permeability at an iron–air interface) usually do not permit solutions that are smooth enough to be in H^2. In such a case, the bounds deteriorate to

$$||u - u^h||_{H^1} \leq C\, h^{\alpha} |u|_{H^2} = O(h^{\alpha}) \tag{2.113}$$
$$||u - u^h||_{L_2} \leq \tilde{C}\, h^{2\alpha} |u|_{H^2} = O(h^{2\alpha}) \tag{2.114}$$

with $0 < \alpha < 1$. The value of α depends on the smoothness (regularity) of the solution u.

2.9 Finite Elements of Higher Order

As explained in the previous sections, the standard FE method utilizes an isoparametric approximation, whereby the geometry and the unknown physical field are interpolated using polynomials of fixed low order (typically 1st or 2nd order). This approach is commonly referred to as h-FEM, as a further increase in accuracy is accomplished by refining the mesh and thus decreasing the mesh size h.

In contrast, the p-version method achieves convergence by successively increasing the polynomial degree (commonly referred to as p), while keeping the spatial discretization fixed, as first introduced by Babuška and Szabó in [19]. It could be proven that in case of smooth problems exponential convergence can be achieved [20]. A distinct feature of the p-version is the usage of hierarchical polynomials, which keeps the polynomials of lower order when increasing p [21]. This is different from the classical Lagrange-type elements as the basis functions are not defined in terms of *nodal* basis functions anymore, which need distinct spatial supporting points, but originate from general hierarchical 1D polynomials (e.g. Legendre et al. [22]) which are then composed to shape functions associated with nodes, edges, faces and the interior of the elements. In other words, the basis functions are not *interpolatory* anymore, and in order to evaluate the field variable, one has to perform true interpolation involving all shape functions.

To illustrate the convergence of linear finite elements, we consider the Helmholtz equation (wave equation in the frequency domain, see Sect. 5.4.5)

$$\boldsymbol{\Delta}\, \hat{p} + k^2 \hat{p} = 0 \tag{2.115}$$

with $\hat{p}$ the acoustic pressure in the frequency domain and k the wave number. Please note that this PDE has the same structure as our model equation in Sect. 2.2. As an example, we consider the simple case of an acoustic wave in a channel with given

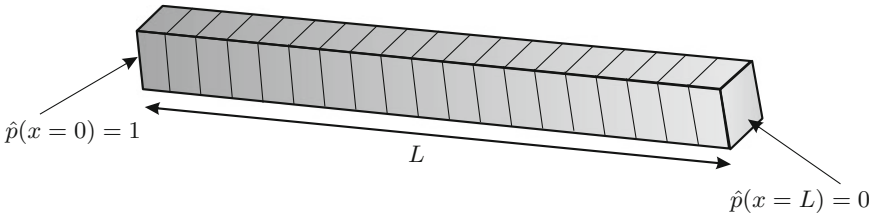


Fig. 2.22 Acoustic wave in a channel with given Dirichlet boundary conditions

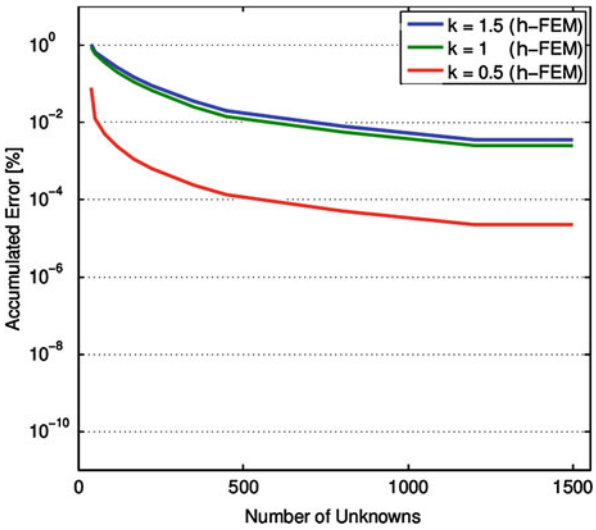


Fig. 2.23 Reduction of numerical error by h-refinement

Dirichlet boundary conditions on the left and right end (see Fig. 2.22). We use three different wave numbers k and preform for the computations a successively reduction of the element mesh size h. For comparison we compute the accumulated error E_{a}

$$E_{\mathrm{a}} = \sqrt{\frac{\sum_{i=1}^{n_{\mathrm{n}}} \left(\hat{p}_i^h - \hat{p}_i\right)^2}{\sum_{i=1}^{n_{\mathrm{n}}} \hat{p}_i^2}} \tag{2.116}$$

with $\hat{p}_i^h$ the FE solution at node i, $\hat{p}_i$ the analytical solution at node i, and n_{n} the number of finite element nodes. The analytic solution $\hat{p}$ computes by

$$\hat{p}(x) = \cos(kx) - \frac{\cos(kL)}{\sin(kL)} \sin(kx). \tag{2.117}$$

In Fig. 2.23 the obtained accumulated error is displayed. It can be seen that the numerical error decays only slowly even for the finest grid featuring approximately

150 elements per wavelength for $k = 0.5$. To increase the rate of convergence we will now introduce two higher order polynomials which can be utilized within FEM. For both approaches, an exponential convergence rate can be achieved as demonstrated for our channel example in Sect. 5.4.5 together with the discretization error analysis.

2.9.1 Legendre Polynomials and Hierarchical Finite Elements

In the FE context, any analytical function u gets approximated by a *finite* dimensional subset of interpolation functions, defined on a finite element mesh. In element local coordinates, this reads as

$$u(\boldsymbol{\xi}) \approx u^h(\boldsymbol{\xi}) = \sum_{i=1}^{n_{eq}} N_i(\boldsymbol{\xi})\, u_i \ , \tag{2.118}$$

where $u^h(\boldsymbol{\xi})$ is the approximated function, N_i the ansatz functions (FE basis functions), u_i the unknown physical quantity for equation i, and n_{eq} the number of equations.

In the case of standard Lagrangian elements, N_i are defined by the corner coordinates and u_i is the related physical quantity at node i. The ansatz functions of first order on the unit domain $\hat{\Omega} \in [-1, 1]$ are defined as follows

$$N_1(\xi) = \frac{1-\xi}{2} \qquad N_2(\xi) = \frac{1+\xi}{2} \ . \tag{2.119}$$

However, one disadvantage of the Lagrangian basis is that for each polynomial degree, one needs a new set of shape functions (see Fig. 2.24a), which prevents the use of different approximation orders within one FE-mesh.

In contrast, hierarchic ansatz functions are defined in such a way that every basis of order p is contained in the set of functions of order $p + 1$ (see Fig. 2.24b). We make use of the Legendre based interpolation functions as given by [20]

$$\begin{aligned} N_1(\xi) &= \frac{1}{2}(1-\xi) \qquad N_2(\xi) = \frac{1}{2}(1+\xi) \\ N_i(\xi) &= \phi_{i-1}(\xi), \quad i = 3, \ldots, p+1 \ , \end{aligned}$$

where ϕ_i denotes the integrated Legendre polynomials L_i

$$\phi_i(\xi) = \int_{-1}^{\xi} L_{i-1}(x)\, dx \tag{2.120}$$

$$L_i(x) = \frac{1}{2^i !} \frac{d^i}{dx^i}(x^2 - 1)^i \ . \tag{2.121}$$

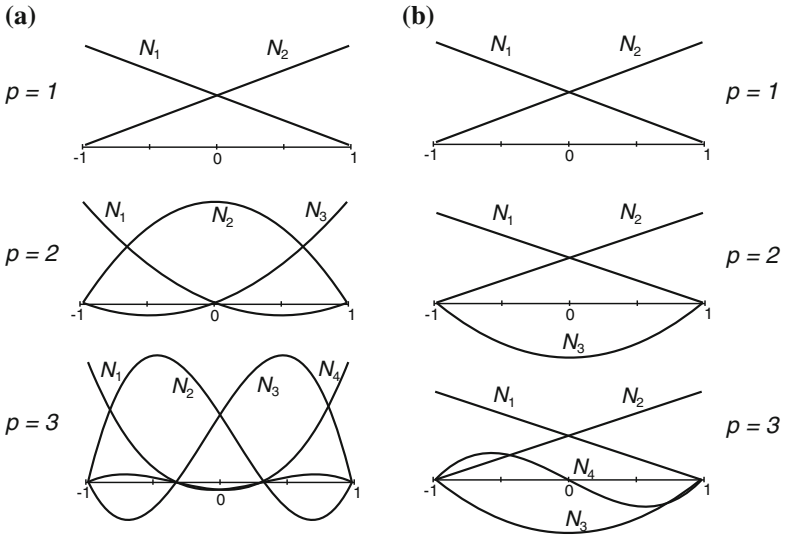


Fig. 2.24 Lagrange and Legendre based ansatz functions up to order 3. **a** Lagrange functions. **b** Legendre functions

These polynomials are mutual orthogonal with respect to the H^1 semi-norm (see (D.11))

$$\int_{-1}^{1} \phi'_i(\xi)\phi'_j(\xi)\, \mathrm{d}\xi = 0 \quad \text{for } i \neq j. \tag{2.122}$$

Furthermore, the following recursive formula allows a fast evaluation for order $i \geq 2$ [23]

$$\begin{aligned} \phi_1(\xi) &= \xi \\ \phi_2(\xi) &= \frac{1}{2}\left(\xi^2 - 1\right) \\ (i+1)\phi_{i+1}(\xi) &= (2i-1)\xi\phi_i(\xi) - (i-2)\phi_{i-1}(\xi) \quad \text{for } i \geq 2. \end{aligned} \tag{2.123}$$

Note that these functions give only a non-zero value within the interval. Therefore they are also called *internal modes* or *bubble modes*.

2.9.1.1 Quadrilateral Element

The definition of shape functions for quadrilateral elements is established by means of a 2D tensor product. The reference quadrilateral is depicted in Fig. 2.25, which

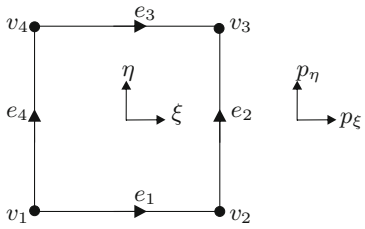


Fig. 2.25 Reference quadrilateral on the unit domain $\hat{\Omega} \in [-1, 1] \times [-1, 1]$

is connected to the two spatial polynomial degrees (p_ξ, p_η). For the quadrilateral element, the basis function can be splitted into the following types:

1. *Vertex modes*: The 4 nodal modes are the standard bilinear functions known from the first order Lagrange element (see Sect. 2.3.2)

$$N_{v_i}^{1,1} = \frac{1}{4}(1 + \xi_i \xi)(1 + \eta_i \eta), \; i = 1, \ldots, 4 \,, \tag{2.124}$$

   where ξ_i, η_i denote the local coordinates of the reference element vertices in $[-1, 1]$. Figure 2.26a displays the shape function for vertex v_3.
2. *Edge modes*: The edge modes are defined individually for each edge and their number depends on the approximation order in the direction of the edge. The modes corresponding to edge 1, e.g. are

$$N_{e_1}^{p_\xi,1} = \frac{1}{2}\phi_{p_\xi}(\xi)(1 - \eta) \,, \tag{2.125}$$

   where ϕ_{p_ξ} denote the related one-dimensional integrated Legendre basis function as given in (2.120).
3. *Internal mode*: The internal modes are only defined within the element interior and vanish at its boundaries (see Fig. 2.26c). They are defined as

$$N_{\text{int}}^{p_\xi, p_\eta} = \phi_{p_\xi}(\xi)\phi_{p_\eta}(\eta) \,. \tag{2.126}$$

In contrast to the one-dimensional case, in 2D the notion of a polynomial degree involves now the spatial tuple (p_ξ, p_η), spanning the global polynomial space, as sketched in Fig. 2.27. The single entries can be identified again as vertex $(p_\xi = 1, p_\eta = 1)$, edge and interior degrees/face degrees of freedom (see Fig. 2.28).
The set of shape functions can now be divided into the following spaces [24]:

- **Trunk Space** $W_{\text{tr}}^p(\Omega)$: This space is spanned by the following set of monomials:
  - $\xi^i \eta^j \quad i, j = 0, 1, \ldots, p; \quad i + j = 0, 1, \ldots, p$
  - $\xi\eta \quad \text{for} \quad p = 1$
  - $\xi^p \eta, \xi\eta^p \quad \text{for} \quad p \geq 2.$

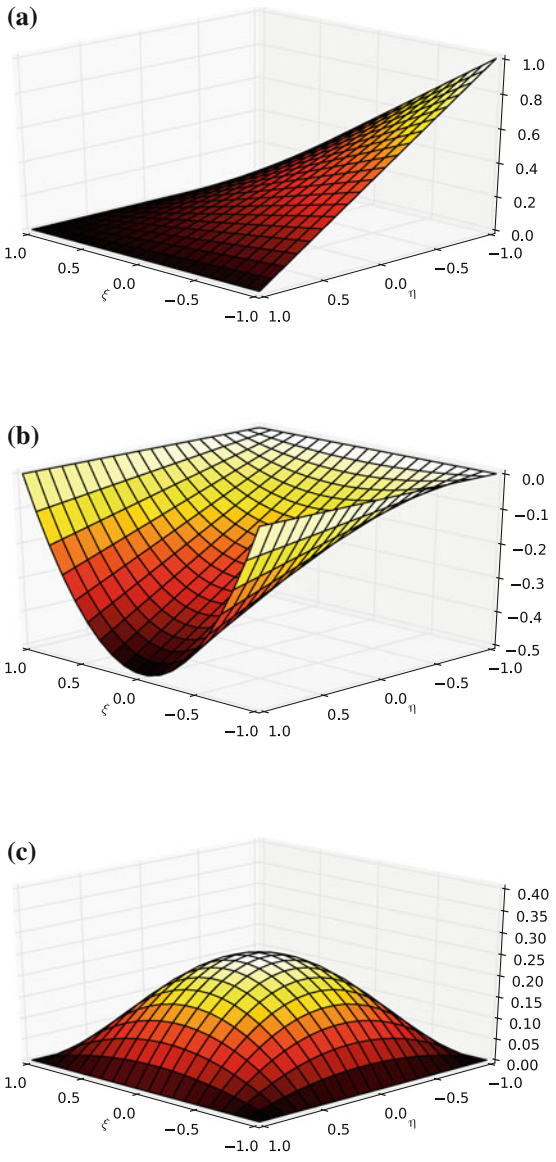


Fig. 2.26 Vortex, edge and bubble functions for quadrilateral element. **a** Basis function for vertex v_3. **b** Edge function of order 2 for edge e_1. **c** Interior shape function of order $p = 2$

This set of functions is depicted in Fig. 2.29 and the dotted line marks the functions for the trunk space W_{tr}^3. The corresponding shape functions are shown in Fig. 2.30.

- **Tensor Product Space** $W_{\text{tp}}^p(\Omega)$: This space is spanned by all monomials $\xi^i \eta^j$ with $i = 0, 1, \ldots, p$ and $j = 0, 1, \ldots, p$. As an example, we display the polynomials for the tensor product space W_{tr}^3 in Fig. 2.31 and the corresponding shape functions in Fig. 2.32.

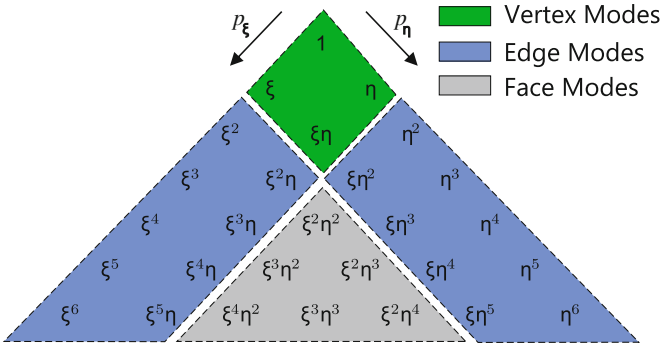


Fig. 2.27 2D polynomials and type of shape functions

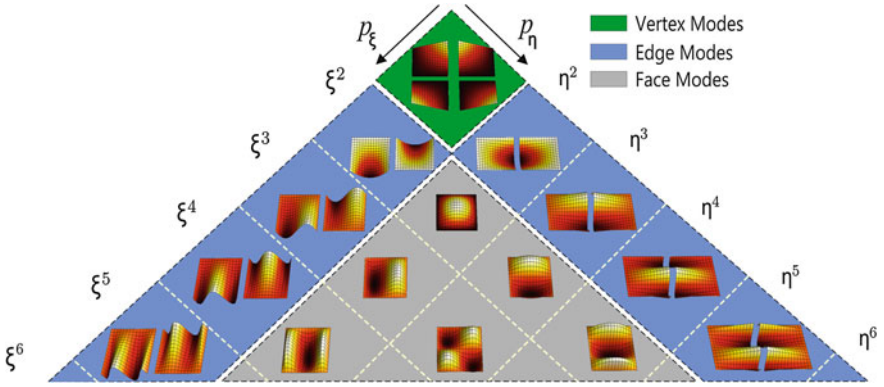
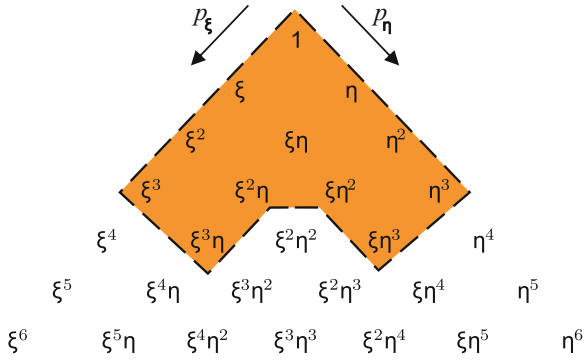


Fig. 2.28 Shape functions according to the polynomials of Fig. 2.27 (scaled for display reasons)

Fig. 2.29 Trunk space with $p = 3$



It is obvious that due to the higher number of internal modes, the tensor product space supplies more FE basis functions. As the internal degrees of freedoms are purely local to the element, they can be eliminated by static condensation. This procedure increases the computational time on the element level. However, it dratsically

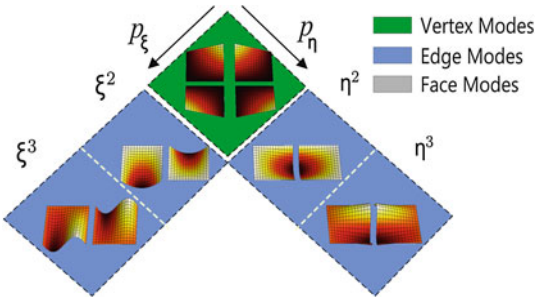


Fig. 2.30 Shape functions of trunk space of order 3 (scaled for display reasons)

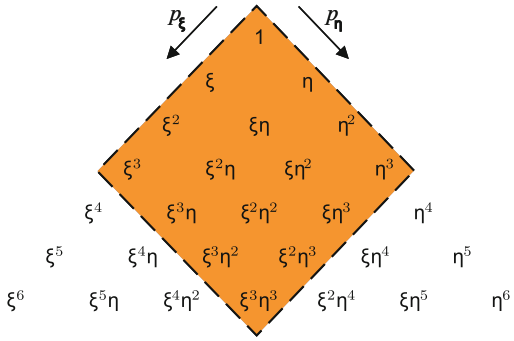


Fig. 2.31 Tensor product space with $p_\xi = p_\eta = 3$

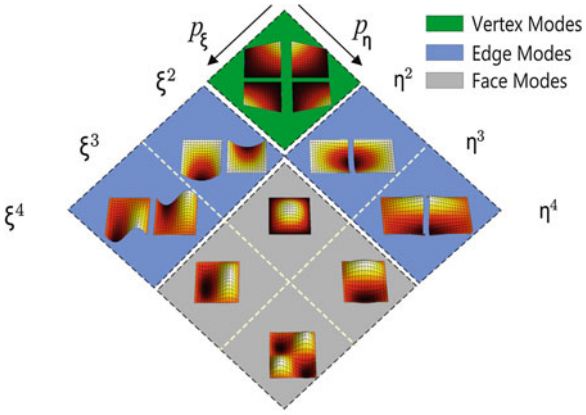


Fig. 2.32 Shape functions of tensor product space of order 3 (scaled for display reasons)

improves the overall performance due to the reduced size of the global system matrix and in addition due to the drastic decrease in the condition number (see Chap. 13), which is of high importance when applying iterative solvers.

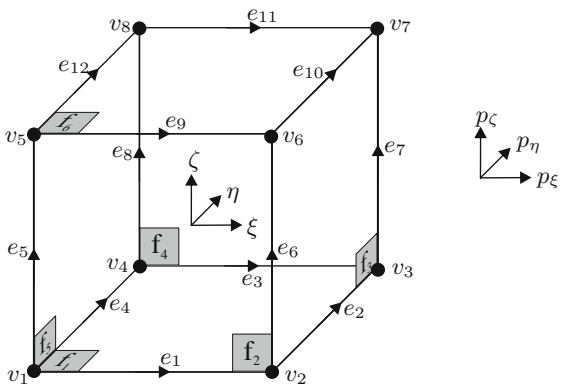


Fig. 2.33 Reference hexahedron with directional polynomial degrees p_ξ, p_η, p_ζ

2.9.1.2 Hexahedral Element

To build the ansatz functions in 2D and 3D, we simply take the Cartesian product of the one-dimensional basis functions. As we have three element local directions ξ, η, ζ, we can accordingly choose three different approximation orders p_ξ, p_η, p_ζ. For the hexahedron the resulting basis functions can be distinguished into the following types (see Fig. 2.33):

1. *Vertex modes*: The 8 vertex modes are the standard trilinear functions known from the first order Lagrange element (see Sect. 2.3.5)

$$N_{v_i}^{1,1,1} = \frac{1}{8}(1+\xi_i\xi)(1+\eta_i\eta)(1+\zeta_i\zeta),\ i = 1,\ldots,8\ , \tag{2.127}$$

   where ξ_i, η_i, ζ_i denote the local coordinates of the reference element vertices in $[-1, 1]$.
2. *Edge modes*: The edge modes are defined individually for each edge and their number depends on the approximation order in the direction of the edge. The modes corresponding to edge 1, e.g. are

$$N_{e_1}^{p_\xi,1,1} = \frac{1}{4}\phi_{p_\xi}(\xi)(1-\eta)(1-\zeta)\ , \tag{2.128}$$

   where ϕ_{p_ξ} denotes the related one-dimensional integrated Legendre basis function as given in (2.120).
3. *Face modes*: Also the face modes are defined for each face separately, e.g. for face 1

$$N_{f_1}^{p_\xi,p_\eta,1} = \frac{1}{2}(1-\zeta)\phi_{p_\xi}(\xi)\phi_{p_\eta}(\eta)\ . \tag{2.129}$$

4. *Internal modes*: The internal modes are only defined within the elements interior and vanish at its boundaries. They are defined as

$$N_{\text{int}}^{p_\xi, p_\eta, p_\zeta} = \phi_{p_\xi}(\xi)\phi_{p_\eta}(\eta)\phi_{p_\zeta}(\zeta) \ . \tag{2.130}$$

The number of nodal functions is always 8, whereas the number of edge, face and internal modes depends on the choice of p_ξ, p_η and p_ζ. In order to guarantee a C^0-continuous approximation, the inter-element orientation of edge and surface functions have to be considered.

2.9.1.3 Enforcing Conformity (Minimum Rule)

In the general 1D case, the polynomial order of each element can be chosen individually, as only nodal and interior/bubble modes are present. In the 2D/3D case however, also edge and face degrees of freedoms are involved. In order to guarantee a conforming approach, the polynomial degrees of edges/faces between neighboring elements have to be set accordingly as demonstrated by the example in Fig. 2.34.

2.9.1.4 Duffy Transformation

By viewing a triangle as a collapsed quadrilateral, one can also construct tensorial-type basis for triangles in a similar way as for quadrilateral elements [25]. Thereby, the *Duffy transformation* from a quadrilateral to a triangle reads as

$$\mathcal{D} : (u, v) \to (\xi, \eta) \quad \begin{aligned} \xi &= \tfrac{1}{4}(1+u)(1-v) \\ \eta &= \tfrac{1}{2}(1+v). \end{aligned} \tag{2.131}$$

By using the inverse of the Duffy transformation, we can parameterize the triangle as follows

$$\begin{aligned} u &= 2\frac{\xi}{1-\eta} - 1 = \frac{N_2 - N_1}{N_1 + N_2} \in [-1, 1] \\ v &= 2\eta - 1 = 2N_3 - 1 = 1 - 2N_1 - 2N_2 \in [-1, 1] \end{aligned} \tag{2.132}$$

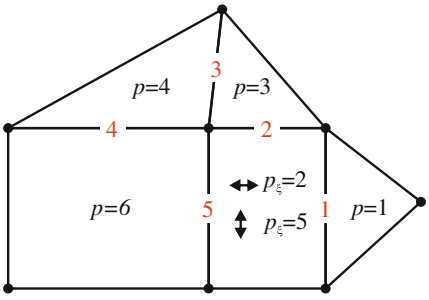


Fig. 2.34 Application of minimum rule results in the displayed edge degrees

with N_1, N_2, N_3 the basis functions as defined in Sect. 2.3.3. In a similar way we can define the Duffy transformation for tetrahedral elements to use tensorial basis

$$\mathcal{D} : (u, v, w) \rightarrow (\xi, \eta, \zeta) \quad \text{with} \quad \begin{aligned} \xi &= \tfrac{1}{8}(1+u)(1-v)(1-w) \\ \eta &= \tfrac{1}{4}(1+v)(1-w) \\ \zeta &= \tfrac{1}{2}(1+w). \end{aligned} \tag{2.133}$$

Here, the inverse map reads as

$$\begin{aligned} u &= 2\frac{\xi}{1-\eta-\zeta} - 1 = \frac{N_2 - N_1}{N_1 + N_2} \in [-1, 1] \\ v &= 2\frac{\eta}{1-\zeta} - 1 = \frac{2N_3 - (1-N_4)}{1-N_4} = \frac{N_3 - (N_1+N_2)}{N_1+N_2+N_3} \in [-1, 1] \\ w &= 2\zeta - 1 = 2N_4 - 1 \in [-1, 1] \end{aligned} \tag{2.134}$$

with N_1, N_2, N_3, N_4 the basis functions as defined in Sect. 2.3.4. In [23] this approach has been used in order to construct higher order FE basis functions for all types of finite elements.

2.9.2 Lagrange Polynomials and Spectral Elements

Spectral methods are high order techniques that are used to solve PDEs either in their strong or weak formulation. Common to all methods is the approximation of unknowns by a high order orthogonal polynomial expansion. In the context of FEM a higher order approximation is mostly limited to order three or four to balance computational effort and desired accuracy. In the context of spectral methods approximation orders up to one thousand are utilized. Examples for such methods are the *Fourier Collocation Method* or the *Fourier Galerkin Method* [26]. Although these methods feature excellent convergence properties their applicability towards complex geometries is difficult due to their global nature, the incorporation of discontinuities in e.g. material parameters is not straight forward and the computational costs increase when high polynomial degrees are needed to resolve all features of the problem under investigation. In contrast to these global or *single domain methods*, the spectral element method can be introduced for classical FEM [27]. The idea is to combine the accuracy and computational advantages of spectral methods with the geometrical flexibility of the finite element method. Within these methods one needs to find a trade-off between polynomial order and the number of finite elements. The error within a spectral element method will vary with

$$E \sim \frac{e^{-\alpha p}}{{n_e}^{p+1}} \quad , \tag{2.135}$$

in which we denote the polynomial order by p and the number of elements by n_e [26]. For the definition of the spectral element method one needs to define the interpolating polynomial including its supporting points and a quadrature rule with abscissas at these supporting points with adequate accuracy for the order of the interpolation polynomial.

Within the Galerkin spectral element method one chooses Lagrange polynomials of order p as interpolating polynomials. These polynomials feature the delta property and extensions towards quadrilateral and hexahedral elements are straight forward. Giving p supporting points at locations ξ_j, distributed on the interval $[\xi_a, \xi_{a+1}]$ one can associate to each supporting a polynomial as

$$N_j^p(\xi) = \prod_{i=0, i\neq j}^{p} \frac{\xi - \xi_i}{\xi_j - \xi_i} \quad . \tag{2.136}$$

We can already see, that for a higher degree of approximation, the complete set of basis functions changes (see Fig. 2.24). The hierarchical approach as done with the integrated Legendre polynomials is obviously no longer possible.

The actual choice of supporting points for Lagrange polynomials is a critical one as we will show in the following. Without further consideration one could choose equidistant points on the given interval which will give rise to instabilities also known as *Runges phenomenon*. Let's approximate the functional

$$f(\xi) = \frac{1}{1 + 10\xi^2} \tag{2.137}$$

on the interval $[-1, 1]$ by sixth and fourteenth order Lagrange polynomials with different set of supporting points. The results are shown in Fig. 2.35. It can be seen that for the equidistant set of supporting points, the interpolated solution does not converge and shows heavy oscillations at the ends of the interval. In fact the interpolation error tends to infinity if the order is increased further. In contrast to this, the solution obtained by choosing the so called *Legendre-Gauss-Lobatto* (LGL) nodes approximates the exact solution smoothly with increasing polynomial order. The *Legendre-Gauss-Lobatto* points are located at the roots of the derivative of the Legendre polynomials.

As already pointed out, the extension of Lagrange shape functions to quadrilateral and hexahedral elements is given by the tensor product. All nine second order shape functions on the quadrilateral element are displayed in Fig. 2.36 and some of the third order functions in Fig. 2.37.

Another crucial point in the spectral element method is the choice of quadrature rule to use. In the case of LGL nodes the natural method would be the *Gauss-Lobatto* quadrature rule which is exact for polynomials of order $2p - 1$ which is lower than the optimal Gaussian quadrature but still sufficient. Summarizing, the abscissas and weight of the *Gauss-Lobatto* rule are given by [28]

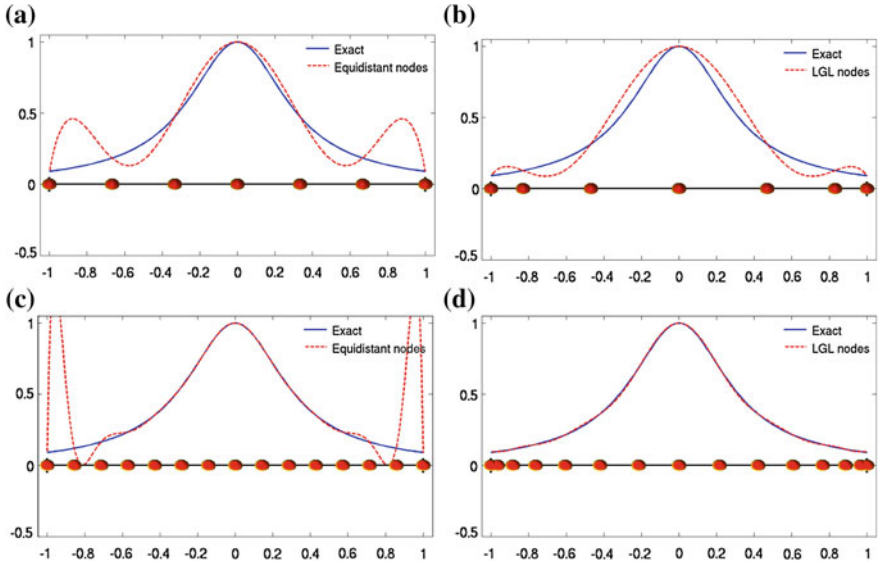


Fig. 2.35 Interpolation results for (2.137) by using Lagrange polynomials. Locations of supporting points are marked by *red circles*. **a** 6th order: Using equidistant supporting points. **b** 6th order: Using non-equidistant Legendre Gauss Lobatto (LGL) nodes. **c** 14th order: Heavy oscillations at the ends of the interval. **d** 14th order: Smooth approximation of exact curve

$$\xi_j = +1, -1 \text{ and roots of } L'_p(\xi) \;;\; w_j = \frac{2}{p(p+1)\left(L_p(\xi_j)\right)^2} \tag{2.138}$$

with L_p the Legendre polynom of order p (see (2.121)). The actual nodes and weights for arbitrary order of approximation can be found using a Newton method [29]. Explicitly computed values can also be found in various text books such as [28].

The big advantage of spectral element methods for time domain computations is that it gives rise to exact diagonal mass matrices which enables the use of explicit time stepping schemes. Although, these are limited in their time step size by the *Courant-Friedrichs-Lewy* (CFL) condition [30], they are preferable due to their low computational cost in each time step especially for problems with a high number of unknowns.

2.10 Flexible Discretization

In many technical applications a sensor or/and actuator is immersed in an acoustic fluid, e.g. ultrasound transducers for nondestructive testing as well as medical diagnostic and therapy, ultrasound cleaning, electrodynamic loudspeakers, capacitive microphones, etc. For many applications, the numerical simulation of the actuator

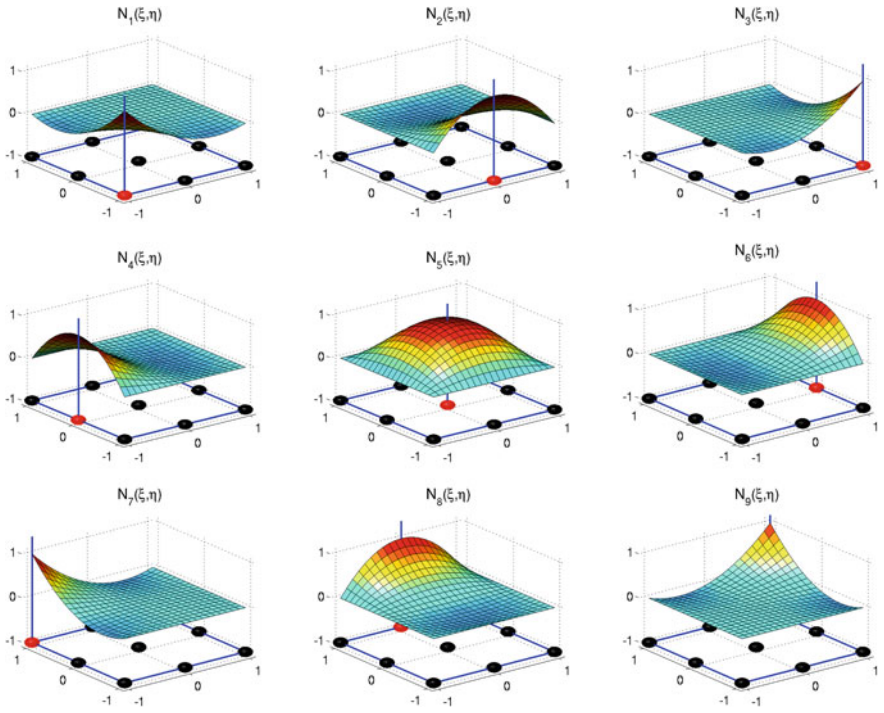


Fig. 2.36 All quadratic shape functions defined on the quadrilateral element

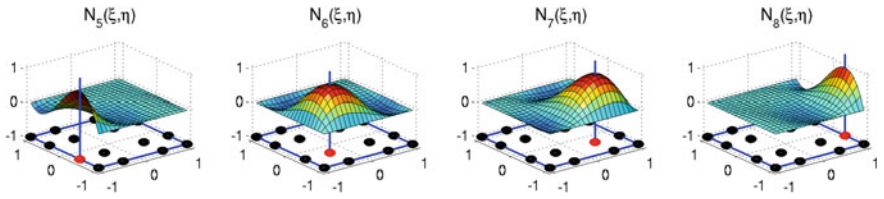


Fig. 2.37 Four of twelve cubic shape functions on the quadrilateral element

mechanism within the structure is quite complex, since in most cases we have to deal with a nonlinear coupled problem (e.g. the electrostatic-mechanical principle used in many micro-electromechanical systems), where in addition to the nonlinear coupling terms each single field is nonlinear (e.g., geometric nonlinearity in mechanics, moving body problem in the electrostatic field, see Chap. 10). Furtheron, in most cases the discretization within the structure has to be much finer than the one we need for the acoustic wave propagation in the fluid. A very similar problem arises in computational aeroacoustics, when solving the inhomogeneous wave equation according to Lighthill's analogy (see Sect. 14.8.2). The inhomogeneous term of the wave equation is calculated by the fluid flow data within the fluid region and, to

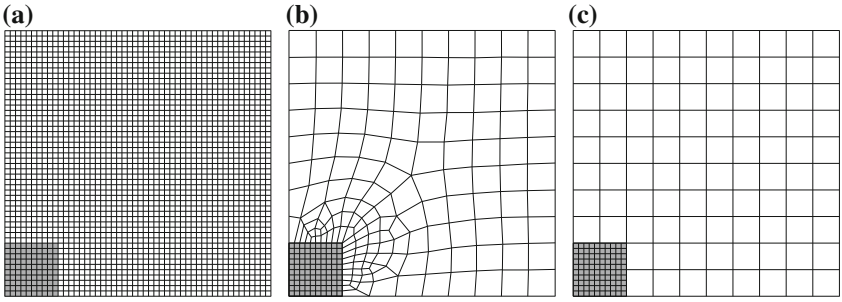


Fig. 2.38 Mesh types. **a** Uniform grid. **b** Unstructured mesh. **c** Non-conforming mesh

obtain reliable results, the discretization of the wave equation within this domain has to be very fine (up to 1,000 linear finite elements per wavelength). However, outside the flow region, the homogeneous wave equation is solved, and one could have a relatively coarse mesh (about 20 linear finite elements per wavelength).

For standard FEM computational meshes are required to be geometrically conforming. This means that either uniform grids which result in many unknowns (c.f. Fig. 2.38a) or many transition elements between fine subdomain grids and coarse ones (c.f. Fig. 2.38b) need to be applied. This can lead to grave problems concerning the mesh quality. Meshes with a poor quality can even cause phenomena like numerical dispersion and reflection in acoustics or mechanics and can therefore lead to physically wrong results.

A computational framework which does not suffer from the mentioned problems and provides the necessary flexibility is the Mortar Finite Element Method (Mortar FEM). In contrast to the traditional FEM, it provides the freedom to apply non-conforming grids across subdomains as Fig. 2.38c shows. The term Mortar finite elements refers to all methods where a Mortar method is used in a finite element context. Here, the coupling is between subdomains with different triangulations and associated finite element spaces. Due to the non-conforming grids, special attention has to be paid to the coupling of the physical quantities across the interfaces. The corresponding projection operation for the involved quantities is in general termed as *mesh intersection problem*, *inter-grid communication problem* or *grid transfer problem*.

2.10.1 Mortar FEM

We start with the geometrical properties of the domain decomposition and the corresponding geometric operations to handle non-conforming meshes, followed by a detailed description of the numerical procedure [31–33].

2.10.1.1 Geometrical Definitions and Terminology

The coupled problem is defined on a decomposition of the global domain Ω into N_Ω non-overlapping subdomains Ω_k satisfying

$$\overline{\Omega} = \bigcup_{k=1}^{N_\Omega} \overline{\Omega}_k, \quad \Omega_k \cap \Omega_{k'} = \emptyset \text{ for } k \neq k'. \tag{2.139}$$

Such decompositions can either be geometrically conforming or geometrically non-conforming. Similar to the definition of conforming finite element meshes, the restriction for a geometrically conforming decomposition is that two subdomains are either allowed to touch along an entire common edge (2D and 3D) or face (3D) Γ_i, in one point, or not at all (cf. Fig. 2.39a). In contrast to that, hanging nodes (2D) respectively edges (3D) are allowed for the geometrically non-conforming case as Fig. 2.39b shows.

In the example in Fig. 2.40 there are three subdomains $N_\Omega = 3$. The partition into subdomains is geometrically non-conforming in this example. There may exist $i \in \{1 \dots N_\Gamma\}$ inner interfaces in total. The *skeleton* $\mathcal{S}$ of the decomposition,

$$\mathcal{S} = \bigcup_{k=1}^{N_\Omega} \partial\Omega_k \setminus \partial\Omega \tag{2.140}$$

consists of the union of the inner interfaces which satisfy the condition

$$\Gamma_i \cap \Gamma_{i'} = \emptyset \text{ for } i \neq i' \text{ and } i, i' \in \{1 \dots N_\Gamma\}. \tag{2.141}$$

In Fig. 2.40 three inner interfaces $N_\Gamma = 3$ are depicted as solid lines. The geometric realization of the skeleton of a d-dimensional domain is given by a number of injective parametric mappings

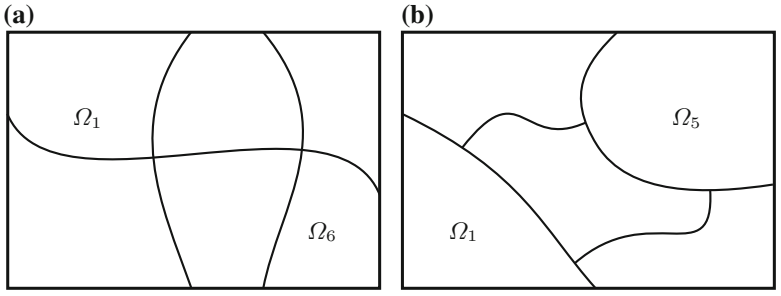


Fig. 2.39 Examples for geometrically conforming and non-conforming domain decompositions. **a** Conforming decomposition. **b** Non-conforming decomposition

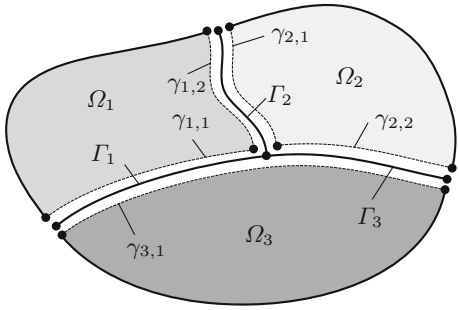


Fig. 2.40 Example of a global domain decomposed into three subdomains. Exploded view of the subdomains with inner interfaces Γ_i and mappings $\gamma_{k,j}$

$$\gamma_{k,j} : \mathcal{R}^{d-1} \to \mathcal{R}^d \quad k \in \{1 \dots N_\Omega\},$$

and j being the number of whole curves (d = 2) or faces (d = 3) of the subdomain Ω_k. Therewith, the inner interfaces for the example in Fig. 2.40 can be written more precisely as

$$\Gamma_1 = \gamma_{1,1} \cap \gamma_{3,1}, \tag{2.142}$$
$$\Gamma_2 = \gamma_{1,2} \cap \gamma_{2,1}, \tag{2.143}$$
$$\Gamma_3 = \gamma_{2,2} \cap \gamma_{3,1}. \tag{2.144}$$

The set of all mappings $\gamma_{k,j}$ admits a partition of the skeleton into a subset $\mathcal{G}_m$ of N_m *mortar* or *master* edges or faces

$$\mathcal{G}_m = \overline{\mathcal{S}} = \bigcup_{m=1}^{N_m} \overline{\gamma}_m, \quad \text{and } \gamma_m \cap \gamma_{m'} = \emptyset,\ m \neq m',\ \gamma_m, \gamma_{m'} \in \{\gamma_{k,j}\}.$$

The subdomain Ω_k associated with a γ_m and therefore also with the corresponding Γ_i is then denoted as mortar or master side $\Omega_{m(i)}$ in respect to γ_m (c.f. Fig. 2.41). The choice of mortar faces γ_m is not unique in general. One possible choice for the given domain could be $\mathcal{G}_m = \{\gamma_{3,1}, \gamma_{1,2}\}$ (cf. Fig. 2.41d). Once this choice has been made the remaining subset of N_s faces which also make up a partition of the skeleton is fixed and the $\gamma_{k,j}$ contained therein are called *non-mortar* or *slave* faces

$$\mathcal{G}_s = \overline{\mathcal{S}} = \bigcup_{s=1}^{N_s} \overline{\gamma}_s, \quad \text{and } \gamma_s \cap \gamma_{s'} = \emptyset,\ s \neq s',\ \gamma_s, \gamma_{s'} \in \{\gamma_{k,j}\} \setminus \mathcal{G}_m.$$

The function values on the slave side of the interface are determined by the values on the corresponding master side. For our example (see Fig. 2.41d) the remaining set of slave faces is $\mathcal{G}_s = \{\gamma_{1,1}, \gamma_{2,1}, \gamma_{2,2}\}$. Depending on the actual choices, the tuple of (N_m, N_s) for this example case, maybe either (2,3) or (3,2), since there are five curves $\gamma_{k,j}$. An algorithm for partitioning the faces $\gamma_{i,j}$ of the skeleton into mortar

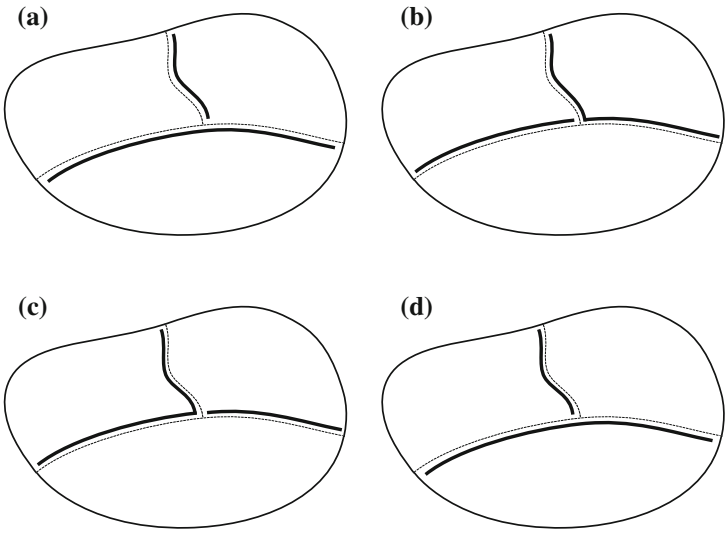


Fig. 2.41 Four possible choices for partitioning the set of main edges making up the skeleton into master and slave sides. Master sides are marked by a *thick line*. **a** $\mathcal{G}_m = \{\gamma_{2,1}, \gamma_{3,1}\}$, $\mathcal{G}_s = \{\gamma_{1,1}, \gamma_{1,2}, \gamma_{2,2}\}$. **b** $\mathcal{G}_m = \{\gamma_{1,1}, \gamma_{2,1}, \gamma_{2,2}\}$, $\mathcal{G}_s = \{\gamma_{1,2}, \gamma_{3,1}\}$. **c** $\mathcal{G}_m = \{\gamma_{1,1}, \gamma_{2,1}, \gamma_{2,2}\}$, $\mathcal{G}_s = \{\gamma_{1,2}, \gamma_{3,1}\}$. **d** $\mathcal{G}_m = \{\gamma_{1,2}, \gamma_{3,1}\}$, $\mathcal{G}_s = \{\gamma_{1,1}, \gamma_{2,1}, \gamma_{2,2}\}$

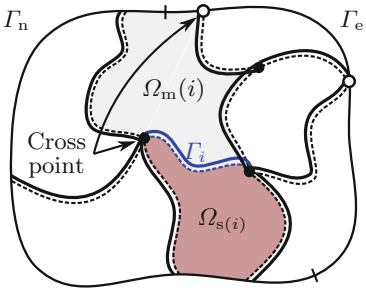


Fig. 2.42 Subdomains with cross points in the interior (*filled dots*) and on the Dirichlet boundary Γ_{e} (*hollow dots*). Master and slave sides in respect to Γ_i

and slave faces on arbitrary geometrically non-conforming polygonal domains is proposed in [33].

The points of $\mathcal{S}$, where the interfaces Γ_i meet and where multiple master sides are involved are called *cross points* or *vertex points* (cf. Fig. 2.42). At these points the function values for the slave sides are determined by multiple master sides, which would result in an overdetermined system of equations. Therefore a special treatment has to be applied for these points. Cross point treatment must also be applied to points where an interface touches a Dirichlet boundary, since the function value is already given there.

2.10.1.2 Formulation of the Coupled Problem

In the following the well-known time independent diffusion or Laplace problem for the unknown scalar function u on a domain Ω will be considered as a prototype for the introduction of the Mortar FEM

$$\begin{aligned} -\nabla\kappa(\boldsymbol{x})\nabla u &= f \quad \text{in} \quad \Omega, \\ u &= u_{\mathrm{e}} \quad \text{on} \quad \Gamma. \end{aligned} \tag{2.145}$$

In order to reformulate the Laplace problem (2.145) on the decomposed domain with possibly discontinuous functions on the skeleton, physical coupling conditions have to be introduced. First of all, the physical quantity has to be continuous across the inner interfaces. Therefore, the jump $[u]$ has to be zero

$$[u] = 0 \text{ on } \mathcal{S}.$$

This condition has to be reformulated as a weak condition using test functions μ from a suitable Lagrange multiplier space M (cf. [32, 34])

$$\int_{\mathcal{S}} [u]\ \mu \mathrm{d}\Gamma = 0 \quad \forall\, \mu \in M \quad \text{and} \quad [u]\ \in H^{1/2}(\mathcal{S}).$$

Furthermore, the flux of the unknown is continuous across the interfaces $\mathcal{S}$ in normal direction. The condition on the jump of the normal derivatives is therefore $[\kappa \partial u/\partial \boldsymbol{n}] = 0$, where $\boldsymbol{n}$ is defined in respect to the slave side. This can be achieved by introducing a Lagrange multiplier (LM)

$$\lambda = -\kappa_s \frac{\partial u_s}{\partial \boldsymbol{n}} = -\kappa_m \frac{\partial u_m}{\partial \boldsymbol{n}} \circ \Phi \text{ on } \mathcal{S}\,. \tag{2.146}$$

Here Φ denotes a spatial mapping, which relates the points on the slave sides to the points on the master sides of the interfaces. The weak formulation of the Laplace problem can now be rewritten on many subdomains by substituting the definition of the Lagrange multiplier according to (2.146). The arising boundary integral over the outer boundary is set to zero for simplicity. Summing up, one arrives at the symmetric saddle point problem of finding (u_i, λ) and $i = 1 \ldots N_\Omega$ such that

$$\begin{aligned} \sum_{i=1}^{N_\Omega} \left(\int_{\Omega_i} \kappa(\boldsymbol{x})(\nabla u)\cdot(\nabla v)\,\mathrm{d}\Omega \right) + \int_{\mathcal{S}} [v]\,\lambda\ \mathrm{d}\Gamma = 0 \\ \int_{\mathcal{S}} [u]\,\mu\ \mathrm{d}\Gamma = 0, \end{aligned} \tag{2.147}$$

for all (μ, v_i). It shall be noted, that μ is a test function with the same basis as the LM λ. This property is responsible for the symmetry of the arising stiffness matrix.

2.10.1.3 Projection of Coordinates

The jump of the unknown function across a single internal interface Γ_i from the master side Γ_m to the slave side Γ_s is given by

$$[u]_i = u_\mathrm{s} - u_\mathrm{m} \circ \Phi_i \quad \text{on} \quad \Gamma_i, \tag{2.148}$$

where Φ_i denotes a bijective global coordinate mapping $\Phi_i : \Gamma_\mathrm{s} \to \Gamma_\mathrm{m}$ on Γ_i. This mapping relates every point on a subset Γ_s of the curve γ_s to a point on its corresponding subset of Γ_m of the master curve γ_m along their common intersection Γ_i. In the continuous setting it is simply the identity mapping

$$\Phi_i(\boldsymbol{x}) = \boldsymbol{x}. \tag{2.149}$$

After the spatial discretization of the subdomains by finite elements the mapping gets more complicated since the discretizations Γ_s and Γ_m of the coincident curves γ_s and γ_m need not coincide any more as Fig. 2.43 shows for two linear elements.

Here an initial point on the curve γ_s gets mapped onto a linear element on the slave side by the mapping ϕ_s and the same point on the curve γ_m gets mapped to an element on the master side by the mapping ϕ_m. As long as both mappings are known, i.e. the finite element code has information about the analytical description of the geometry of the inner interfaces, the mortar mapping Φ_i can be written as

$$\Phi_i(\boldsymbol{x}) = \phi_\mathrm{m} \circ \phi_\mathrm{s}^{-1}(\boldsymbol{x}). \tag{2.150}$$

The information about the analytical geometry may simply be lost, when only the discretized subdomain meshes generated by a preprocessor are available as input. In

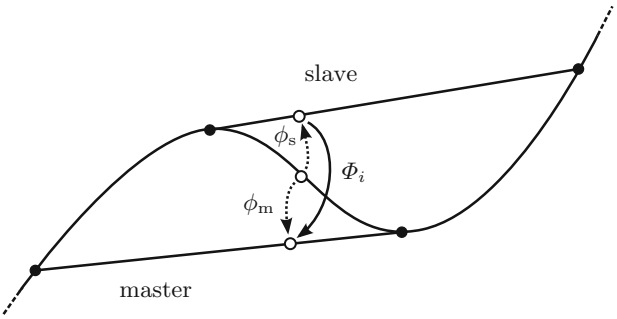


Fig. 2.43 Mortar mapping along *curved* interface

such a situation it is necessary to find good local approximations of Φ. In practice this can be achieved by finding pairs of closely spaced elements and by defining a mapping in respect to these pairs. Algorithms for calculating such approximative mappings are described in [35].

2.10.1.4 Mesh Intersection Operations

In 2D the intersection of line elements has to be considered. If the interface is planar this amounts to do simple interval checks. If the interface is curved, the elements have to be projected onto a common line segment first for doing the interval checks there. These considerations also apply in a modified way for domains in 3D where interfaces are surfaces. It has to be noted however that the seemingly simple operation of finding the intersection domain of arbitrary surface elements is a highly non-trivial task even for first order elements with straight edges.

- *Intersection of Straight Line Elements*
  If an intersection of two co-linear line elements exists, it is again a line element sharing two of the four endpoints of both parent elements in the co-linear case. To check for an intersection one has to project the endpoints $[\mathbf{m}_1, \mathbf{m}_2]$ of the element on the master side of the interface in two dimensional coordinates to the one dimensional local coordinate system defined by the endpoints of the slave element $[\mathbf{s}_1, \mathbf{s}_2]$.
  The local coordinates of the slave nodes $[\mathbf{s}_1, \mathbf{s}_2]$ are trivially given by 0 and 1. The four local coordinates of the pair of lines are then brought into ascending order and therefore four possible cases for the intersection of two line elements may be identified as depicted in Fig. 2.44.

  1. $\xi_1 < 0 \ \wedge \ 0 < \xi_2 \leq 1$: the intersection is the line $[\mathbf{s}_1, \mathbf{m}_2]$
  2. $0 \leq \xi_1 < 1 \ \wedge \ \xi_2 > 1$: the intersection is the line $[\mathbf{m}_1, \mathbf{s}_2]$
  3. $\xi_1 \leq 0 \ \wedge \ \xi_2 \geq 1$: the intersection is the line $[\mathbf{s}_1, \mathbf{s}_2]$
  4. $\xi_1 > 0 \ \wedge \ \xi_2 < 1$: the intersection is the line $[\mathbf{m}_1, \mathbf{m}_2]$

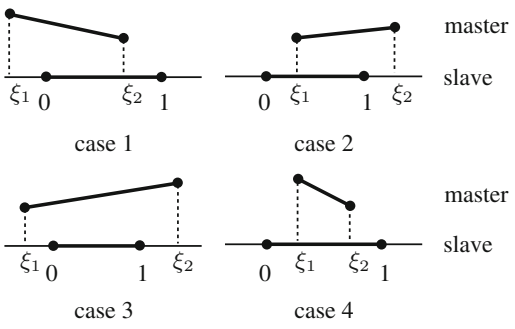


Fig. 2.44 Four possible cases of *two lines* intersecting each other

- *Intersection on Coplanar Interfaces*
  The algorithm for finding intersections of lines can be extended in a straight forward manner to a 3D setting if only axis-parallel quadrilateral elements are to be considered on the interface. The term axis-parallel does not refer to the global coordinate axes in this context, but to the fact that the quadrilateral edges on both sides of the interface have to be parallel. This includes the case of parallelograms. The method is more thoroughly described in [32, 36].
  Again, the local coordinates (ξ_1, η_1) and (ξ_2, η_2) of the first and third corner of the master element are computed in respect to the slave element (c.f. Fig. 2.45). After bringing the local coordinates for both directions into ascending order there are sixteen possible cases for the intersection of two quadrilaterals. The ordering is necessary due to the fact, that the order of nodes for elements is just guaranteed to be counter-clockwise, but the master element might have been rotated in respect to the slave element as a whole.
  The algorithm used for intersecting a pair of arbitrary planar convex polygons is a specialization of the Sutherland-Hodgman algorithm [37] and is designed to fit the needs of finite element grids. The polygons are considered to be cyclic directed graphs with the same orientation (i.e. clockwise or counter-clockwise). The basic idea of this algorithm is to find a cut of two edges of the polygons as a starting point and then to "walk" along the boundary of the intersection. Once the intersection polygon has been obtained it gets either triangulated or a quadrilateral element is generated if the number of corners is equal to four. The algorithm is described in detail in [38].
- *Intersection on Arbitrarily Curved Interfaces*
  The intersection calculation on curved interfaces is quite complicated since the elements have to be mapped to a common plane before the actual intersection calculation can be performed. The approach proposed in [32] relies on pair-wise mappings of the elements to the plane or line of the element on the slave side of the interface along the face normal vector. Figure 2.46 depicts the situation for two skewed triangle elements which only intersect in a common line element. In order to perform the intersection calculation triangle 2 gets projected into the plane of 1 resulting in triangle 3. The actual intersection takes place between the triangles 1 and triangle 3.
  It should not go unnoticed that this method still has the drawback of overlapping elements and gaps are not being treated properly. It may also lead to holes in the intersection mesh. This inevitably leads to poor results while evaluating the integrals and can just be counteracted, by increasing the resolution of the interface mesh.

2.10.1.5 Discrete Problem

For some convergence estimates to hold, each triangulation $\mathcal{T}_{k,h}$ is usually assumed to be quasi-uniform. Quasi-uniformity for a triangular mesh means that there exist two

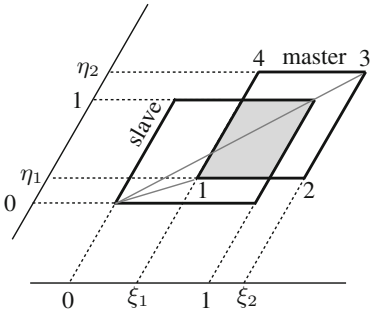


Fig. 2.45 Intersection of two axis-parallel parallelogram-shaped elements

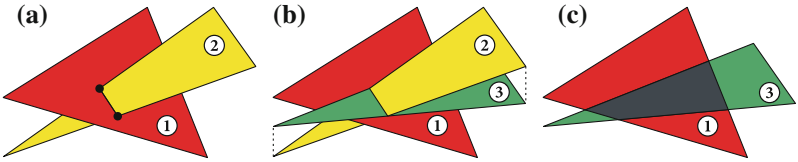


Fig. 2.46 Projection of two triangles onto common plane (taken from [38]). **a** Intersection of two triangles. **b** Projection onto common plane. **c** Actual configuration for polygon intersection

positive constants τ and σ such that for all triangles T of $\mathcal{T}_{k,h}$, $\tau h_{\max} \leq h_T \leq \sigma \rho_T$. In this relation $h_{\max}$ and h_T denote the maximum diameter of the triangles in $\mathcal{T}_{k,h}$ and the diameter of the triangle T respectively. The diameter of the inscribed circle is denoted by ρ_T. The associated finite element spaces with piecewise linear elements are denoted by $S_h(\mathcal{T}_{k,h})$. The unconfined product space with no continuity condition between the subdomains is then given by

$$X_h = \prod_{k=1}^{N_\Omega} S_h(\mathcal{T}_{k,h}) \tag{2.151}$$
$$= \{u \in L_2(\Omega) \mid u_{|\Omega_k} \in S_h(\mathcal{T}_{k,h}) \text{ for } k = 1, \ldots, N_\Omega,\ u_{|\Gamma_n} = u_{\Gamma_e}\}.$$

Unfortunately this space is not suitable for the discretization. This fact can quickly be understood if the jumps of the functions across the skeleton are considered (cf. Fig. 2.47). The derivatives across those interfaces evaluate to indefinite values. Since the Lebesgue measure of $\mathcal{S}$ is not zero the functions u are elements of $L_2(\Omega)$ but not of $H^1(\Omega)$.

Since the locations where the unknowns are defined are in general not the same across the inner interfaces, no pointwise continuity of the unknown function can be required. Therefore, a weak continuity condition is required to be fulfilled. After some necessary definitions this so-called mortar condition can be stated.

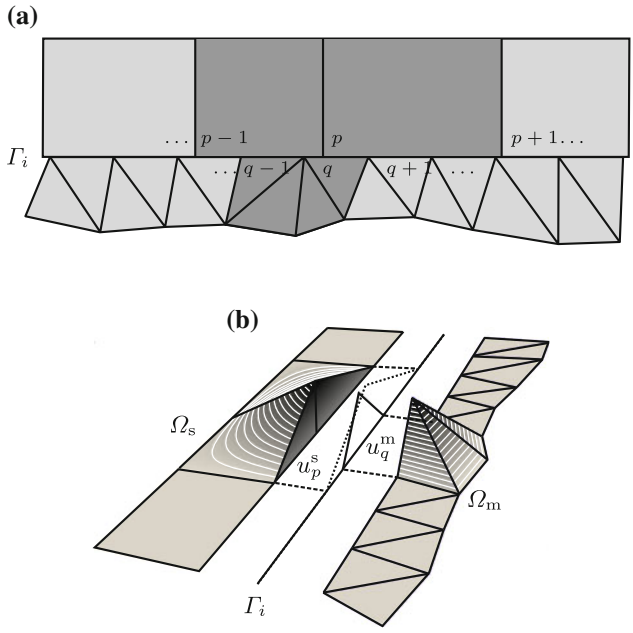


Fig. 2.47 Illustration of the traces of finite element basis functions along a non-conforming interface. The basis functions for two degrees of freedom on the interface are depicted. A bilinear nodal basis is used on the quadrilateral elements, while a linear basis is used on the triangles. **a** Non-conforming meshes along coplanar interface. **b** Basis functions for two DOFs on the interface and their traces

Let $\mathcal{T}_\mathrm{p} = \mathcal{T}_{\mathrm{k},h}|\gamma_\mathrm{p}$ denote the discretization inherited from the volume discretization in subdomain k to which the master or slave edge γ_p $p = m, s$ belongs. On every discretization $\mathcal{T}_\mathrm{s}$ a space of test functions is introduced which, for piece-wise (bi-)linear basis functions in the slave subdomains $\Omega(\mathrm{s})$, fulfill the following conditions which characterize the Mortar method among other domain decomposition methods:

$$\begin{aligned} &(i) \quad M_{s,h} \subset S_{s,h}(\mathcal{T}_\mathrm{s}) \subset H^{-1/2} \\ &(ii) \quad \dim(M_{s,h}) = \dim(S_{s,h}(\mathcal{T}_\mathrm{s}) \cap H_0^1(\mathcal{T}_\mathrm{s})) \\ &(iii) \quad M_{s,h} \text{ contains constants on } \partial\mathcal{T}_\mathrm{s} \end{aligned} \tag{2.152}$$

Condition (i) states that the space on interface $\mathcal{T}_\mathrm{s}$ has to be a subspace of the trace space inherited from the triangulation on the slave side $S_h(\mathcal{T}_{\mathrm{k},h})$. The second condition (ii) states that the dimension of the space should be equal to the dimension of the trace space on the slave side with homogeneous Dirichlet conditions on the boundaries of the discretized interface $\mathcal{T}_\mathrm{s}$. The third condition (iii) makes sure, that the function value from the boundaries of the master triangulations get transferred to the boundary nodes of the slave side. For higher order basis functions on the slave

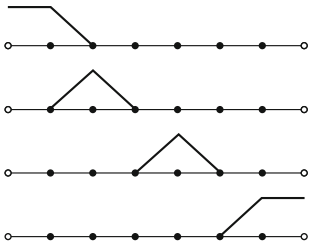


Fig. 2.48 Basis functions for the standard Lagrange multiplier for one interface

side, this condition gets modified by stipulating that the space should be able to represent the first derivative of the functions. In Fig. 2.48 the standard nodal basis functions derived from the piece-wise linear basis of the unknown function on the slave side is depicted. It has to be noted that no LM degree of freedom is assigned to the first and last node of the discretization.

Let q be the number of inner interfaces Γ_i geometrically corresponding to a single non-mortar edge γ_s. With these definitions it is possible to define the discrete Mortar bilinear form for all N_s slave side discretizations

$$b(u_h, \mu_h) := \sum_{s=1}^{N_s} \int_{\mathcal{I}_s} \left(u_{s,h} - u_{m,i,h} \circ \Phi_i\right) \mu_h \, \mathrm{d}\Gamma \quad \text{and} \quad i = 1...q. \tag{2.153}$$

Therewith, a global function u_h is a Mortar function if the discrete Mortar condition

$$b(u_h, \mu_h) = 0 \tag{2.154}$$

is fulfilled. It states that the L_2-projection of the jump has to be orthogonal to the Lagrange multiplier space. The domain bilinear form $a(\cdot, \cdot)$ and right hand side linear form $(\cdot, \cdot)$ are defined as

$$a(u_h, v_h) = \sum_{k=1}^{N_\Omega} \int_{\Omega_k} \kappa \nabla u_h \cdot \nabla v_h \, \mathrm{d}\Omega, \tag{2.155}$$

$$(f, v_h) = \sum_{k=1}^{N_\Omega} \int_{\Omega_k} f v_h \, \mathrm{d}\Omega, \quad f \in L_2(\Omega). \tag{2.156}$$

The material parameter $\kappa(\boldsymbol{x})$ is continuous in each Ω_k and satisfies the following relation $\kappa(\boldsymbol{x}) \geq \kappa_0 > 0$. For simplicity of the presentation $\kappa(\boldsymbol{x}) = \kappa_k =$ constant for each Ω_k. The previous relation ensures the positive definiteness of the bilinear form $a(\cdot, \cdot)$. With these definitions, the mixed formulation of the Mortar coupled problem can be written as: Find (u_h, λ_h) such that

$$a(u_h, v_h) + b(v_h, \lambda_h) = (f, v_h) \tag{2.157}$$

$$b(u_h, \mu_h) = 0\,, \tag{2.158}$$

where

$$\begin{aligned} u_h &\in X_h, \\ v_h &\in \{v \in X_h \,|\, v = 0 \,\text{on}\, \Gamma_\text{e}\} \\ \lambda_h\,, \mu_h &\in M^h \subset H^{-1/2}(\mathcal{S})\,. \end{aligned}$$

For the special case of $k = 2$ subdomains the algebraic system arising from (2.157 and 2.158) has the following form

$$\begin{pmatrix} \mathbf{K}_{u1} & 0 & \mathbf{D} \\ 0 & \mathbf{K}_{u2} & \mathbf{M} \\ \mathbf{D}^T & \mathbf{M}^T & 0 \end{pmatrix} \begin{pmatrix} \underline{u_1} \\ \underline{u_2} \\ \underline{\lambda} \end{pmatrix} = \begin{pmatrix} \underline{f_1} \\ \underline{f_2} \\ \underline{0} \end{pmatrix}. \tag{2.159}$$

In (2.159) $\mathbf{K}_{u1}, \mathbf{K}_{u2}$ are standard stiffness matrices of the subdomains. The new matrices $\mathbf{D}$ and $\mathbf{M}$ are due to the non-conforming interface and are formally mass matrices. They compute element-wise as

$$\begin{aligned} \mathbf{D} &= \bigwedge_{e=1}^{n_\text{es}} \mathbf{d}^e;\ \mathbf{d}^e = [d_{ab}]; \\ d_{ab} &= \int_{\Gamma_e} N_a^s N_b^s \,\mathrm{d}\Gamma\,, \qquad (2.160) \\ \mathbf{M} &= \bigwedge_{e=1}^{n_\text{isec}} \mathbf{m}^e;\ \mathbf{m}^e = [m_{ab}]; \\ m_{ab} &= \int_{\Gamma_e} (N_a^m \circ F_m^{-1} \circ \Phi)(N_b^s \circ F_s^{-1})\,\mathrm{d}\Gamma\,. \qquad (2.161) \end{aligned}$$

Here n_es is the number of surface elements on the slave side of the interface and n_isec is the number of intersection elements on the interface. The finite element basis functions N_a^s and N_a^m denote the traces of the FE basis on the slave and on the master side of the interface, N_b^s denotes the Lagrange multiplier basis given with respect to the slave side, Φ is the global coordinate mapping from slave to master side and F^{-1} is the coordinate mapping from global to element local coordinates (see Sect. 2.3.2). The integrals in (2.160) are defined in the usual manner with respect to $\mathcal{T}_s$. The necessary steps for the computation of the integrals in (2.161) are discussed in the following.

2.10.1.6 Evaluation of Coupling Integrals

The feature which makes the Mortar FEM so flexible, namely the usage of non-conforming meshes in different subdomains, comes at the cost of a more elaborate implementation. Since the grids are allowed to be non-conforming on the interfaces of two subdomains, the integrals defined on these interfaces involving basis functions from both sides have to be evaluated with respect to two different meshes. It is therefore necessary to first compute the domains where pairs of elements on the interface intersect. In the developed implementation the intersection elements on curved interfaces are found by projecting the element on the master side to element on the slave side along the slave side normal vector. The coupling integrals defined in (2.161) are then evaluated over these intersection elements and it is up to the assembly operator to assemble the corresponding results into the correct positions of the coupling matrices.

Once the intersection elements have been found, the coupling integral (2.161) can be evaluated on these elements by means of standard Gauss quadrature. For a single element the integral can be rewritten as follows

$$\int_{\Gamma_e} (N_a^m \circ F_m^{-1} \circ \Phi)(N_b^s \circ F_s^{-1}) \, \mathrm{d}\Gamma \approx \sum_{l=1}^{n_{\text{int}}} W_l N_a(\boldsymbol{\xi}_l^m) N_b(\boldsymbol{\xi}_l^s) \mathcal{J}_{F_e}(\boldsymbol{\xi}_l^e) \, . \tag{2.162}$$

Here n_{isec} is the number of intersection elements, n_{int} is the number of quadrature points, W_l are the quadrature weights and the determinant of the Jacobian $\mathcal{J}^e$ accounts for the change in volume due to the element mapping. The difficulty which arises when this quadrature formula is applied, is that only the quadrature point $\boldsymbol{\xi}_l^e$ in respect to the local coordinates of the intersection element is known in advance and that the points $\boldsymbol{\xi}_l^m$ in the master element and $\boldsymbol{\xi}_l^s$ in the slave element have to be projected into those elements, before the basis functions can be evaluated there (see Fig. 2.49).

It is very important to notice that nodes of the intersection element do not carry any degrees of freedom by themselves. The intersection element is just an auxiliary geometrical entity which only serves as integration domain. The projection operation for general elements involves the following steps:

1. Map local coordinates $\boldsymbol{\xi}_l^e$ of quadrature point in intersection element to global coordinates using F_e
2. Project global coordinates of integration point on slave side to global coordinate on master side using Φ
3. Map global coordinates of quadrature point to local coordinates $\boldsymbol{\xi}_l^m$ of master element using F_m^{-1}
4. Map global coordinates of quadrature point to local coordinates $\boldsymbol{\xi}_l^s$ of slave element using F_s^{-1}.

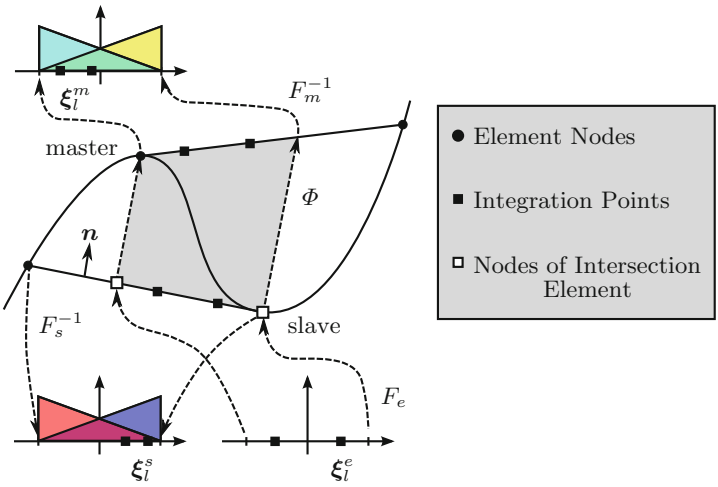


Fig. 2.49 Projection of quadrature points from the intersection element into the master element and into the slave element

Points 3 and 4 in general involve the application of a Newton algorithm. A linear mapping algorithm may only be used for 2-node isoparametric line elements, 3-node isoparametric triangle elements or higher order elements which just use a linear local-to-global mapping. Once the values of the basis functions N_a and N_b have been obtained and (2.162) has been evaluated, the assembly operator has to make sure, that the contribution gets added to the corresponding entry in the coupling matrix.

2.10.2 Nitsche Type Mortaring

In the previous section, we have discussed in detail the standard Mortar FEM, which fulfills the continuity of the trace (the physical quantity) in a weak sense and the flux (normal derivative of the physical quantity) in a strong sense by introducing the Lagrange multiplier (LM). The main drawback of this approach can be seen in the arising saddle point structure of the coupled system of Eq. (2.147). Therefore, we present a second approach, which is named Nitsche type mortaring.

The method of Nitsche [39] was originally introduced to impose essential boundary conditions weakly. Unlike the penalty method, it is consistent with the original differential equation. The strong point of Nitsche's method is that it retains the convergence rate of the underlying FE method, whereas the standard penalty method either requires a very large penalty parameter or destroys the condition number of the resulting algebraic system of equations. Let us consider the Laplace problem as stated in (2.145). For standard FEM, we will require a functional space with the constraint according to the Dirichlet (essential) boundary condition

$$V = \{\varphi \in H^1 \,|\, \varphi|_\Gamma = u_\mathrm{e}\}. \tag{2.163}$$

The weak formulation will then read as follows: Find $u \in V$ such that

$$\int_\Omega \kappa(\boldsymbol{x})\, \nabla v \cdot \nabla u \, \mathrm{d}\Omega = \int_\Omega f v \, \mathrm{d}\Omega \tag{2.164}$$

for all $v \in V_0 = \{\varphi \in H^1 \,|\, \varphi|_\Gamma = 0\}$. Now, Nitsche's approach incooperates the Dirichlet boundary condition and the functional space for u is free of constraints. Thereby the weak formulation reads as follows: Find $u \in H^1$ such that

$$\begin{aligned}\int_\Omega \kappa\, \nabla v \cdot \nabla u \, \mathrm{d}\Omega - \int_\Gamma \kappa v \frac{\partial u}{\partial \boldsymbol{n}} \, \mathrm{d}\Gamma - \int_\Gamma \kappa u \frac{\partial v}{\partial \boldsymbol{n}} \, \mathrm{d}\Gamma + \beta\kappa \sum_{E(\Gamma)} \frac{1}{h_E} \int_{\Gamma_E} uv \, \mathrm{d}\Gamma \\ = \int_\Omega f v \, \mathrm{d}\Omega - \int_\Gamma u_\mathrm{e} \frac{\partial v}{\partial \boldsymbol{n}} \, \mathrm{d}\Gamma + \beta\kappa \sum_{E(\Gamma)} \frac{1}{h_E} \int_{\Gamma_E} gv \, \mathrm{d}\Gamma.\end{aligned} \tag{2.165}$$

Thereby, the boundary Γ is discretized into surface elements and the two terms with the sum penalize the deviation of u at the boundary from u_e. Please note that the material parameter κ is explicitly within the penalty term. For this formulation, one can proof convergence for $h \to 0$ and $\beta > 0$.

We will now apply Nitsche's ansatz for a computational domain consisting of two subdomains Ω_1 and Ω_2 with a common interface Γ_I as display in Fig. 2.50. Thereby, we introduce two test functions v_1 and v_2 and write in a first step the weak formulation of each subdomain individually

$$\int_{\Omega_1} \kappa_1\, \nabla w_1 \cdot \nabla u_1 \mathrm{d}\,\mathrm{d}\Omega - \int_{\Gamma_\mathrm{I}} \kappa_1\, w_1 \frac{\partial u_1}{\partial \mathbf{n}_1} \, \mathrm{d}\Gamma = \int_{\Omega_1} w_1 f_1 \, \mathrm{d}\Omega \tag{2.166}$$

$$\int_{\Omega_2} \kappa_2\, \nabla w_2 \cdot \nabla u_2 \, \mathrm{d}\Omega - \int_{\Gamma_\mathrm{I}} \kappa_2\, w_2 \frac{\partial u_2}{\partial \mathbf{n}_2} \, \mathrm{d}\Gamma = \int_{\Omega_2} w_2 f_2 \, \mathrm{d}\Omega. \tag{2.167}$$

For the sake of a simpler notation, we choose κ_1 and κ_2 constant in Ω_1 and Ω_2, which is by no means a restriction. In general, the material parameter can even depend on

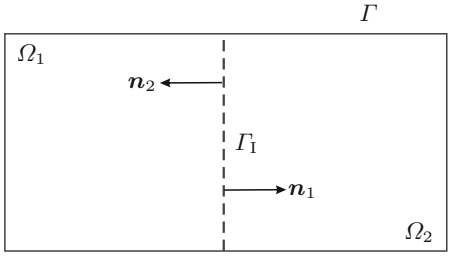


Fig. 2.50 Domain with two sub-regions Ω_1 and Ω_2

the physical quantity u (nonlinear case). In a next step, we add the two Eq. (2.166) and (2.167), and explore the relation

$$\mathbf{n} = \mathbf{n}_1 = -\mathbf{n}_2\,; \quad \kappa_1 \frac{\partial u_1}{\partial \mathbf{n}_1} = \kappa_1 \frac{\partial u_1}{\partial \mathbf{n}} = \kappa_2 \frac{\partial u_2}{\partial \mathbf{n}_2} = -\kappa_2 \frac{\partial u_2}{\partial \mathbf{n}}$$

to arrive at

$$\int_{\Omega_1} \kappa_1 \nabla w_1 \cdot \nabla u_1 \, \mathrm{d}\Omega + \int_{\Omega_2} \kappa_2 \nabla w_2 \cdot \nabla u_2 \, \mathrm{d}\Omega - \int_{\Gamma_1} \kappa_1 [w] \frac{\partial u_1}{\partial \mathbf{n}} \, \mathrm{d}\Gamma = \int_{\Omega_1} w_1 f_1 \, \mathrm{d}\Omega + \int_{\Omega_2} w_2 f_2 \, \mathrm{d}\Omega. \tag{2.168}$$

In (2.169) the operator [ ] defines the jump operator, e.g., $[w] = w_1 - w_2$. In order to retain symmetry, we add to (2.169) the following term

$$-\int_{\Gamma_1} \kappa_1 \frac{\partial w_1}{\partial \mathbf{n}} [u] \, \mathrm{d}\Gamma \quad \text{with} \quad [u] = u_1 - u_2.$$

This operation is allowed, since we postulate on the interface $u_1 = u_2$. In a final step, we add the penalization term

$$\beta \bar{\kappa} \sum_{E(\Gamma_1)} \frac{1}{h_E} \int_{\Gamma_E} [w][u] \, \mathrm{d}\Gamma$$

with β a penalty factor and $\bar{\kappa} = (\kappa_1 + \kappa_2)/2$. Therewith, we arrive at the final formulation for Nitsche's approach

$$\begin{aligned} &\int_{\Omega_1} \kappa_1 \nabla w_1 \cdot \nabla u_1 \, \mathrm{d}\Omega + \int_{\Omega_2} \kappa_2 \nabla w_2 \cdot \nabla u_2 \, \mathrm{d}\Omega \\ &\quad \underbrace{- \int_{\Gamma_1} \kappa_1 [w] \frac{\partial u_1}{\partial \mathbf{n}} \, \mathrm{d}\Omega}_{Consistency} \underbrace{- \int_{\Gamma_1} \kappa_1 \frac{\partial w_1}{\partial \mathbf{n}} [u] \, \mathrm{d}\Gamma}_{Symmetrization} \\ &\quad \underbrace{+ \beta \bar{\kappa} \sum_{E(\Gamma_1)} \frac{1}{h_E} \int_{\Gamma_E} [w][u] \, \mathrm{d}\Gamma}_{Penalty/Stabilization} \\ &\quad = \int_{\Omega_1} w_1 f_1 \, \mathrm{d}\Omega + \int_{\Omega_2} w_2 f_2 \, \mathrm{d}\Omega. \end{aligned} \tag{2.169}$$

If the penalty parameter β in (2.169) is chosen large enough, the bilinear form is coercive on the discrete space $V_h \subset H^1$ and one can derive optimal a priori error estimates in both the energy norm and the L_2 norm for polynomials of arbitrary degree (but same at both sides) [40].

$$\left(\sum_{i=1}^{2} ||\nabla(u - u^h)||^2_{0,\Omega_i} + h^{-1} ||[u]||^2_{\Gamma_1} \right)^{\frac{1}{2}} \leq C h^p ||u||_{\Omega, p+1} \tag{2.170}$$

$$||u - u^h||_{0,\Omega} \leq C h^{p+1} ||u||_{\Omega, p+1}. \tag{2.171}$$

In order to obtain the system of equations, we rewrite (2.169) as two equations with all individual terms according to the two test functions w_1 and w_2

$$\begin{aligned} &\int_{\Omega_1} \kappa_1 \nabla w_1 \cdot \nabla u_1 \, \mathrm{d}\Omega - \int_{\Gamma_1} \kappa_1 \, w_1 \frac{\partial u_1}{\partial \mathbf{n}} \, \mathrm{d}\Gamma - \int_{\Gamma_1} \kappa_1 \frac{\partial w_1}{\partial \mathbf{n}} u_1 \, \mathrm{d}\Gamma \\ &\quad + \int_{\Gamma_1} \kappa_1 \frac{\partial w_1}{\partial \mathbf{n}} u_2 \, \mathrm{d}\Gamma + \beta \bar{\kappa} \sum_{E(\Gamma_1)} \frac{1}{h_E} \int_{\Gamma_E} w_1 u_1 \, \mathrm{d}\Gamma \\ &\quad - \beta \bar{\kappa} \sum_{E(\Gamma_1)} \frac{1}{h_E} \int_{\Gamma_E} w_1 \, u_2 \, \mathrm{d}\Gamma = \int_{\Omega_1} w_1 f_1 \, \mathrm{d}\Omega \\ &\int_{\Omega_2} \kappa_2 \nabla w_2 \cdot \nabla u_2 \, \mathrm{d}\Omega + \int_{\Gamma_1} \kappa_1 \, w_2 \frac{\partial u_1}{\partial \mathbf{n}} \, \mathrm{d}\Gamma + \beta \bar{\kappa} \sum_{E(\Gamma_1)} \frac{1}{h_E} \int_{\Gamma_E} w_2 u_2 \, \mathrm{d}\Gamma \\ &\quad - \beta \bar{\kappa} \sum_{E(\Gamma_1)} \frac{1}{h_E} \int_{\Gamma_E} w_2 \, u_1 \, \mathrm{d}\Gamma = \int_{\Omega_1} w_2 f_2 \, \mathrm{d}\Omega. \end{aligned} \tag{2.172}$$

We now assume a discretization with non-conforming meshes at the interface Γ_1 as displayed in Fig. 2.51. Furthermore, we perform the FE ansatz according to

$$w_1 \approx w_1^h = \sum_i N_{1i} \, w_{1i} \, ; \quad u_1 \approx u_1^h = \sum_j N_{1j} \, u_{1j} \tag{2.173}$$

$$w_2 \approx w_2^h = \sum_i N_{2i} \, w_{2i} \, ; \quad u_2 \approx u_2^h = \sum_j N_{2j} \, u_{2j} \tag{2.174}$$

to arrive at the discrete system of equations. As in the case of Mortar FEM, we will also need all operations (projection of coordinates, intersection operations, etc.) between the two surface meshes in order to compute the entries of the matrices. Now, the matrix system of equations reads as follows

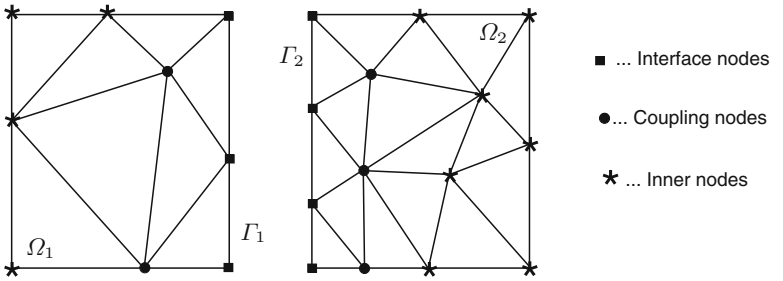


Fig. 2.51 Discretized subdomains Ω_1 and Ω_2

$$\begin{pmatrix} \boldsymbol{K}_{11} & 0 \\ 0 & \boldsymbol{K}_{22} \end{pmatrix} \begin{pmatrix} \underline{u}_1 \\ \underline{u}_2 \end{pmatrix} + \begin{pmatrix} \boldsymbol{K}_{\Gamma_1} & \boldsymbol{K}_{\Gamma_1\Gamma_2} \\ \boldsymbol{K}_{\Gamma_2\Gamma_1} & \boldsymbol{K}_{\Gamma_2} \end{pmatrix} \begin{pmatrix} \underline{u}_1 \\ \underline{u}_2 \end{pmatrix} = \begin{pmatrix} \underline{f}_1 \\ \underline{f}_2 \end{pmatrix}. \tag{2.175}$$

Thereby, the entries of the matrices as well as right hand side compute as

$$\boldsymbol{K}_{11}^{ij} = \int_{\Omega_1} \kappa_1 \, \nabla N_{1i} \cdot \nabla N_{1j} \, \mathrm{d}\Omega \tag{2.176}$$

$$\boldsymbol{K}_{22}^{ij} = \int_{\Omega_2} \kappa_2 \, \nabla N_{2i} \cdot \nabla N_{2j} \, \mathrm{d}\Omega \tag{2.177}$$

$$\begin{aligned} \boldsymbol{K}_{\Gamma_1}^{ij} = & -\int_{\Gamma_1} \kappa_1 \, N_{1i} \, \frac{\partial N_{1j}}{\partial \mathbf{n}} \, \mathrm{d}\Gamma - \int_{\Gamma_1} \kappa_1 \, \frac{\partial N_{1i}}{\partial \mathbf{n}} \, N_{1j} \, \mathrm{d}\Gamma \\ & + \beta \bar{\kappa} \sum_{E(\Gamma_1)} \frac{1}{h_E} \int_{\Gamma_E} N_{1i} N_{1j} \, \mathrm{d}\Gamma \end{aligned} \tag{2.178}$$

$$\begin{aligned} \boldsymbol{K}_{\Gamma_1\Gamma_2}^{ij} &= \int_{\Gamma_1} \kappa_1 \, \frac{\partial N_{1i}}{\partial \mathbf{n}} \, N_{2j} \, \mathrm{d}\Gamma - \beta \bar{\kappa} \sum_{E(\Gamma_1)} \frac{1}{h_E} \int_{\Gamma_E} N_{1i} N_{2j} \, \mathrm{d}\Gamma \\ &= \left(\boldsymbol{K}_{\Gamma_2\Gamma_1}^{ij} \right)^T \end{aligned} \tag{2.179}$$

$$\boldsymbol{K}_{\Gamma_2}^{ij} = \beta \bar{\kappa} \sum_{E(\Gamma_2)} \frac{1}{h_E} \int_{\Gamma_E} N_{2i} N_{2j} \, \mathrm{d}\Gamma \tag{2.180}$$

$$\underline{f}_1^i = \int_{\Omega_1} N_{1i} \, f_1 \, \mathrm{d}\Omega \tag{2.181}$$

$$\underline{f}_2^i = \int_{\Omega_2} N_{2i} \, f_2 \, \mathrm{d}\Omega \tag{2.182}$$

Here, we have already substituted Γ_I by Γ_1 as well as Γ_2, which are the discretized interfaces (see Fig. 2.51). Please note that not only nodes on the interfaces but also neighboring nodes in Ω_1 and in Ω_2 are involved, since the computation of some entries requires normal derivatives. Studying the structure of our coupled system of equations, we see that our formulation is symmetric and does not introduce any additional unknowns (we have no Lagrange multiplier as in case of Mortar FEM). However, we have a penalty parameter β, which has to be chosen large enough to guaranty $u_1 = u_2$ and on the other hand should not be too large to deteriorate the condition number of the system matrix. However, practical computations show that the result is not very sensitive to the choice of β and a value of about 100 is enough.

2.10.3 Numerical Example

In order to compare the two non-conforming methods, we consider an electromagnetic computation in 2D, which can be described by (2.145) when using for κ the magnetic reluctivity ν and A_z (z-component of magnetic vector potential) for u (see Chap. 6). The setup under investigation is a solenoid as displayed in Fig. 2.52.
We have generated three different meshes:

1. Conforming mesh for reference computations;
2. Non-conforming mesh, where the interface is completely inside the air region (see Fig. 2.52);
3. Non-conforming mesh, where the interface includes a part of the surface of the yoke (see Fig. 2.53).

For the iron core (yoke and anchor), we consider a nonlinear BH-curve (see Sect. 6.7.5) and perform computations on the conforming grid as well as the two non-conforming grids. Figure 2.54 shows the resulting magnetic flux density obtained by the classical Mortar approach. Comparing the computed fields of the different formulations, no visible differences can be seen. To perform a more detailed analysis,

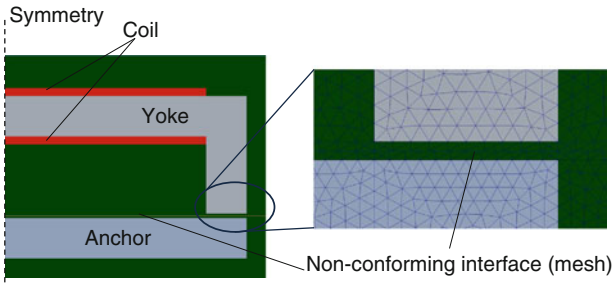


Fig. 2.52 Computational setup of the solenoid and detail of non-conforming mesh with the non-conforming interface just in the air region

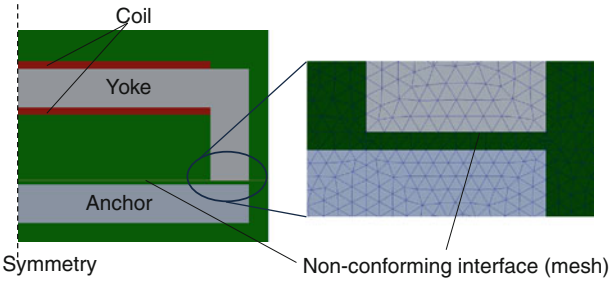


Fig. 2.53 Computational setup of the solenoid and detail of non-conforming mesh with the non-conforming interface including the iron core

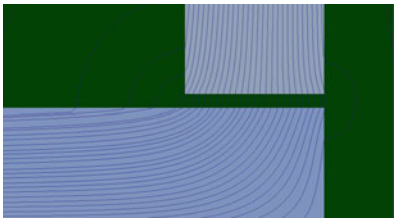


Fig. 2.54 Computed magnetic flux lines in case of non-conforming mesh using Mortar formulation

Table 2.3 Magnetic energy in Ws (computations with classical Mortar and Nitsche type mortaring on the non-conforming mesh according to Fig. 2.52)

	Conform (Ws)	Mortar (Ws)	Error	
Anchor	5288.62	5348.27	1.13 %	
Yoke	8268.42	8372.63	1.26 %	
	Nitsche $\beta = 20$		Nitsche $\beta = 100$	
	(Ws)	Error	(Ws)	Error
Anchor 1	5347.38	1.11 %	5329.47	0.77 %
Yoke	8371.11	1.24 %	8339.55	0.86 %
	Nitsche $\beta = 500$		Nitsche $\beta = 1000$	
	(Ws)	Error	(Ws)	Error
Anchor	5275.94	0.24 %	5233.28	1.05 %
Yoke	8243.98	0.30 %	8167.48	1.22 %

we compute the magnetic energy in the yoke and the anchor region. Table 2.3 lists the results obtained by the conforming mesh as well as non-conforming mesh according to Fig. 2.52. We have also investigated in different penalty factors β for the Nitsche type mortaring method. As it can be seen, in case of $\beta = 500$ we obtain the smallest error compared to computations with the conforming mesh. In conclusion, we can state that the Nitsche type mortaring approach is quite robust w.r.t. the penalty factor β and can be even superior to the classical Mortar method if the optimal value for β

Table 2.4 Magnetic energy in Ws (computations with classical Mortar and Nitsche type mortaring on the non-conforming mesh according to Fig. 2.53)

	Conform	Mortar	
	(Ws)	(Ws)	Error
Anchor	5288.62	5347.8	1.12 %
Yoke	8268.42	8371.03	1.24 %
	Nitsche $\beta = 20$		Nitsche $\beta = 100$	
	(Ws)	Error	(Ws)	Error
Anchor	5360.19	1.35 %	5340.34	0.97 %
Yoke	8393.31	1.50 %	8356.8	1.07 %
	Nitsche $\beta = 500$		Nitsche $\beta = 1000$	
	(Ws)	Error	(Ws)	Error
Anchor	5303.95	0.29 %	5274.77	0.26 %
Yoke	8288.53	0.24 %	8233.72	0.42 %

is chosen. These findings also hold in the case, where the non-conforming interface is not completely in air but also contains a part of the surface of the yoke, where the magnetic reluctivity ν is discontinuous (see Table 2.4).

For further applications we refer to Sect. 5.4.4 concerning acoustics, to Chap. 11 for rotating systems in electromagnetics, to Sect. 8.3.2 for mechanical-acoustic systems, to Sect. 14.8.2 for computational aeroacoustics, and for computational mechanics we refer to [41].

References

1. K.J. Bathe, *Finite Element Procedures* (Prentice Hall, New Jersey, 1996)
2. N. Ida, *Engineering Electromagnetics* (Springer, 2004)
3. O.C. Zienkiewicz, R.L. Taylor, *The Finite Element Method*, vol. **1** (Butterworth-Heinemann, UK, 2003)
4. T.J.R. Hughes, *The Finite Element Method*, 1st edn. (Prentice-Hall, New Jersey, 1987)
5. F.X. Zgainski, J.L. Coulomb, Y. Marechal, A new family of finite elements: the pyramidal element. IEEE Trans. Magn. **32**, 1393–1396 (1996)
6. M. Jung, U. Langer, *Methode der Finiten Elemente für Ingenieure*, 2nd edn. (Springer, 2013)
7. K. Meyberg, P. Vachenauer, *Höhere Mathematik 1* (Springer, 1993)
8. H. Whitney, *Geometric Integration Theory* (Princeton University Press, Princeton, 1957)
9. J.C. Nédélec, Mixed finite elements in R^3. Numer. Math. **35**, 315–341 (1980)
10. M.L. Barton, Z.J. Cendes, New vector finite elements for three-dimensional magnetic field computation. J. Appl. Phys. **61**(8), 3919–3921 (1987)
11. A. Bossavit, J.C. Verite, A mixed FEM-BIEM method to solve 3-D eddy current problems. IEEE Trans. Magn. **18**, 431–435 (1982)
12. A. Kameari, Three dimensional eddy current calculation using edge elements for magnetic vector potential. Appl. Electromagn. Mater. 225–236 (1986)

13. G. Mur, A.T. Hoop, A finite-element method for computing three-dimensional electromagnetic fields in inhomogeneous media. IEEE Trans. Magn. **21**, 2188–2191 (1985)
14. J.S. Welij, Calculation of eddy current in terms of H on hexahedra. IEEE Trans. Magn. **21**, 2239–2241 (1985)
15. P. Silvester, R. Ferrari, *Finite Elements for Electrical Engineers* (Cambridge, 1996)
16. M. Ainsworth, J.T. Oden, *A Posteriori Error Estimation in Finite Element Analysis*, (Wiley, 2000)
17. Ch. Großmann, H.G. Roos, *Numerik Partieller Differentialgleichungen* (Teubner, 1994)
18. G. Strang, G. Fix, *An Analysis of the Finite Element Method* (Cambridge Press, Wellesley, 2008)
19. I. Babuska, B. Szabo, I. Katz, The p-version of the finite element method. SIAM J. Numer. Anal. **18**(3), 515–545 (1981)
20. B. Szabó, I. Babuška, *Finite Element Analysis*, 1st edn. (Wiley, 1991)
21. O.C. Zienkiewicz, J.P. De, S.R. Gago, D.W. Kelly, The hierarchical concept in finite element analysis. Comput. Struct. **16**, 53–65 (1983)
22. G. Szegö, Orthogonal polynomials. American Mathematical Society Colloquium Publications, no. Bd. 23. American Mathematical Society (1959)
23. S. Zaglmayr, High order finite element methods for electromagnetic field computation. Ph.D. thesis, Johannes Kepler University, Linz (2006)
24. A. Düster, Lecture notes: high order FEM. 132 (2005)
25. G. Karniadakis, S.J. Sherwin, *Spectral/HP Element Methods for CFD* (Oxford University Press, 1999)
26. D.A. Kopriva, *Implementing Spectral Methods for Partial Differential Equations* (Springer, Dordrecht, 2009)
27. A.T. Patera, A spectral element method for fluid dynamics—laminar flow in a channel expansion. J. Comput. Phys. **54**, 468–488 (1984)
28. G.C. Cohen, *Higher-Order Numerical Methods for Transient Wave Equations* (Springer, New York, 2002)
29. William H. Press, Saul A. Teukolsky, William T. Vetterling, Brian P. Flannery, *Numerical Recipes 3rd edition: The Art of Scientific Computing*, 3rd edn. (Cambridge University Press, New York, 2007)
30. R. Courant, K. Friedrichs, H. Lewy, On the partial difference equations of mathematical physics. IBM J. Res. Dev. **11**, 215–234 (1967)
31. C. Bernardi, Y. Maday, F. Rapetti, Basics and some applications of the mortar element method. GAMM-Mitt. **28**(2), 97–123 (2005)
32. B. Flemisch, Non-matching triangulations of curvilinear interfaces applied to electromechanics and elasto-acoustics, Ph.D. thesis, University of Stuttgart, (2006)
33. J. Danek, H. Kutakova, The mortar finite element method in 2D: implementation in MATLAB, in *16th Annual Conference Proceedings of Technical Computing* (Prague, Czech Republic, 2008)
34. B.I. Wohlmuth, A mortar finite element method using dual spaces for the Lagrange multiplier. SIAM J. Numer. Anal. **38**(3), 989–1012 (2000). MRMR1781212 (2001h:65132)
35. S. Triebenbacher, Nonmatching grids for the numerical simulation of problems from aeroacoustics and vibroacoustics. Ph.D. thesis, Alpen-Adria-Universität Klagenfurt, Austria (2012)
36. S. Triebenbacher, M. Kaltenbacher, B. Flemisch, B. Wohlmuth, Applications of the mortar finite element method in vibroacoustics and flow induced noise computations. Acta Acust. United Acust. **96**, 536–553 (2010)
37. Ivan E. Sutherland, Gary W. Hodgman, Reentrant polygon clipping. Commun. ACM **17**, 32–42 (1974)
38. J. Grabinger, Mechanical-acoustic coupling on non-matching finite element grids. Master's thesis, University Erlangen-Nuremberg, June 2007
39. J. Nitsche, Über ein Variationsprinzip zur Lösung von Dirichlet-Problemen bei Verwendung von Teilräumen, die keinen Randbedingungen unterworfen sind. Tech. Report 36, Abh. Math. Univ. Hamburg, (1971)

40. A. Hansbo, P. Hansbo, M.G. Larson, A finite element method on composite grids based on nitsche's method. ESAIM Math. Model. Numer. Anal. **37**(3), 495–514 (2003)
41. A. Fritz, S. Hüeber, B. Wohlmuth, A comparison of mortar and Nitsche techniques for linear elasticity. CALCOLO (2004)

http://www.springer.com/978-3-642-40169-5

Numerical Simulation of Mechatronic Sensors and Actuators
Finite Elements for Computational Multiphysics
Kaltenbacher, M.
2015, XXVII, 587 p. 409 illus., 133 illus. in color.,
Hardcover
ISBN: 978-3-642-40169-5