Chapter 1

Introduction

As stated in the preface, we assume that the reader knows the basic concepts of the finite element method (see, e.g., [11, 39, 46, 47, 119, 178, 180, 191]). Nevertheless, let us review at least the most commonly used concepts – scalar and vector-valued Hilbert spaces, trivia of finite elements in these spaces, basic principles of the discretization of time-independent and evolutionary problems and a few additional topics that will be important for higher-order finite element technology.

1.1 Finite elements

DEFINITION 1.1 (Finite element) Finite element *in the sense of Ciarlet [47] is a triad* $\mathcal{K} = (K, P, \Sigma)$*, where*

- K *is a domain in* $\mathbf{R}^d$ *– we will confine ourselves to intervals* ($d = 1$)*, triangles and quadrilaterals* ($d = 2$)*, and tetrahedra, bricks and prisms* ($d = 3$)*.*
- P *is a space of polynomials on* K *of dimension* $dim(P) = N_P$*.*
- $\Sigma = \{L_1, L_2, \ldots, L_{N_P}\}$ *is a set of linear forms*

$$L_i : P \to \mathbf{R}, \quad i = 1, 2, \ldots, N_P. \tag{1.1}$$

The elements of Σ *are called* degrees of freedom *(and often abbreviated as DOF).*

1.1.1 Function spaces H^1, $\boldsymbol{H}$(curl) and $\boldsymbol{H}$(div)

Let $\Omega \subset \mathbf{R}^d$ be a bounded domain with Lipschitz-continuous boundary, d being the spatial dimension. The scalar Hilbert space of functions

$$H^1 = \{u \in L^2(\Omega); \partial u/\partial x_i \in L^2(\Omega),\ 1 \le i \le d\} \tag{1.2}$$

is the basic and most commonly used Sobolev space. Recall that the partial derivatives in (1.2) are understood in the sense of distributions. The Hilbert spaces

$$\boldsymbol{H}(\mathrm{curl}) = \{\boldsymbol{u} \in [L^2(\Omega)]^d; \mathbf{curl}\,\boldsymbol{u} \in [L^2(\Omega)]^d\} \tag{1.3}$$

and

$$\boldsymbol{H}(\mathrm{div}) = \{\boldsymbol{u} \in [L^2(\Omega)]^d; \mathrm{div}\,\boldsymbol{u} \in L^2(\Omega)\} \tag{1.4}$$

of vector-valued functions (defined for $d = 2, 3$) appear in variational formulations of problems rooted, e.g., in Maxwell's equations, mixed formulations in elasticity and acoustics.

Notice that the spaces $\boldsymbol{H}(\mathrm{curl})$ and $\boldsymbol{H}(\mathrm{div})$ fall between the spaces L^2 and H^1 in the sense that only some combinations of the partial derivatives need to be square-integrable.

1.1.2 Unisolvency of finite elements

Definition 1.2 introduces *unisolvency* as another expression for *compatibility* of the set of degrees of freedom Σ with the polynomial space P.

DEFINITION 1.2 (Unisolvency of finite elements) *The finite element $\mathcal{K} = (K, P, \Sigma)$ is said to be* unisolvent *if for every function $g \in P$ it holds*

$$L_1(g) = L_2(g) = \ldots = L_{N_P}(g) = 0 \ \Rightarrow\ g = 0. \tag{1.5}$$

In other words, every vector of numbers

$$\boldsymbol{L}(g) = (L_1(g), L_2(g), \ldots, L_{N_P}(g))^T \in \mathbb{R}^{N_P}$$

uniquely identifies a polynomial g in the space P.

Definition 1.3 together with Theorem 1.1 offer a useful characterization of unisolvency of finite elements.

DEFINITION 1.3 (δ-property) *Let $\mathcal{K} = (K, P, \Sigma)$, $dim(P) = N_P$, be a finite element. We say that a set of functions $\mathcal{B} = \{\theta_1, \theta_2, \ldots, \theta_{N_P}\} \subset P$ has the* δ-property *if*

$$L_i(\theta_j) = \delta_{ij} \quad \textit{for all } 1 \le i, j \le N_P. \tag{1.6}$$

Here δ_{ij} is the standard Kronecker delta, $\delta_{ij} = 1$ if $i = j$ and $\delta_{ij} = 0$ otherwise.

THEOREM 1.1 (Characterization of unisolvency)
Consider a finite element $\mathcal{K} = (K, P, \Sigma)$, $dim(P) = N_P$. The finite element $\mathcal{K}$ is unisolvent if and only if there exists a unique basis $\mathcal{B} = \{\theta_1, \theta_2, \ldots, \theta_{N_P}\} \subset P$ satisfying the δ-property.

PROOF First assume that the finite element $\mathcal{K}$ is unisolvent. We take an arbitrary basis $\{g_1, g_2, \ldots, g_{N_P}\} \subset P$ and express each sought function θ_j, $j = 1, \ldots, N_P$, as

$$\theta_j = \sum_{k=1}^{N_P} a_{kj} g_k.$$

To satisfy the δ-property, we require that

$$L_i(\theta_j) = L_i\left(\sum_{k=1}^{N_P} a_{kj} g_k\right) = \sum_{k=1}^{N_P} a_{kj} L_i(g_k) = \delta_{ij}, \quad 1 \le i, j \le N_P, \tag{1.7}$$

which yields a system of N_P linear equations for each j. Summarized, these linear systems give a matrix equation

$$\boldsymbol{L}\boldsymbol{A} = \boldsymbol{I},$$

where $\boldsymbol{L} = \{L_i(g_k)\}_{i,k=1}^{N_P}$, $\boldsymbol{I} = \{\delta_{ij}\}_{i,j=1}^{N_P}$ is the canonical matrix and the unknown matrix $\boldsymbol{A} = \{a_{kj}\}_{k,j=1}^{N_P}$ contains in its rows coefficients corresponding to the functions $\theta_1, \theta_2, \ldots, \theta_{N_P}$, respectively. Let us assume that the columns of $\boldsymbol{L}$ are linearly dependent, i.e., that there exists a nontrivial set of coefficients $\alpha_1, \alpha_2, \ldots, \alpha_{N_P}$ such that

$$\sum_{k=1}^{N_P} \alpha_k L_i(g_k) = L_i\left(\sum_{k=1}^{N_P} \alpha_k g_k\right) = 0 \quad \text{for all } i = 1, 2, \ldots, N_P. \tag{1.8}$$

Since $\sum_{k=1}^{N_P} \alpha_k g_k$ is a nontrivial function, (1.8) is in contradiction with unisolvency of the element $\mathcal{K}$. Hence, the matrix $\boldsymbol{L}$ is invertible and the functions $\theta_1, \theta_2, \ldots, \theta_{N_P}$ are uniquely identified by the coefficients

$$\boldsymbol{A} = \boldsymbol{L}^{-1}\boldsymbol{I}.$$

It remains to be shown that the functions $\theta_1, \theta_2, \ldots, \theta_{N_P}$ are linearly independent, i.e., that

$$\sum_{j=1}^{N_P} \beta_j \theta_j = 0 \;\Rightarrow\; \beta_i = 0 \text{ for all } i = 1, 2, \ldots, N_P.$$

Obviously this is true since

$$0 = L_i \left(\sum_{j=1}^{N_P} \beta_j \theta_j \right) = \sum_{j=1}^{N_P} \beta_j L_i(\theta_j) = \beta_i$$

for all $i = 1, 2, \ldots, N$. Therefore the functions $\theta_1, \theta_2, \ldots, \theta_{N_P}$ constitute a basis in the space P, which by (1.7) has the δ-property.

The other implication is also easy to verify: Let $\mathcal{B} = \{\theta_1, \theta_2, \ldots, \theta_{N_P}\}$ be a basis of the space P satisfying the δ-property. Every function $g \in P$ can be expressed as

$$g = \sum_{j=1}^{N_P} \gamma_j \theta_j.$$

Assuming that

$$L_1(g) = L_2(g) = \ldots = L_{N_P}(g) = 0,$$

we immediately conclude that

$$0 = L_i(g) = L_i \left(\sum_{j=1}^{N_P} \gamma_j \theta_j \right) = \gamma_i \quad \text{for all } i = 1, 2, \ldots, N_P.$$

Hence necessarily $g = 0$ and the finite element is unisolvent. ∎

REMARK 1.1 (Checking unisolvency) The proof to Theorem 1.1 offers a simple procedure to check the unisolvency of a finite element: one considers an arbitrary basis of the polynomial space P and constructs the matrix $\boldsymbol{L}$ by applying the linear forms $L_1, L_2, \ldots, L_{N_P} \in \Sigma$ to the basis functions. If the matrix $\boldsymbol{L}$ is invertible, one knows that the element is unisolvent, and moreover $\boldsymbol{L}^{-1}$ yields the basis functions satisfying the δ-property. If the matrix $\boldsymbol{L}$ is not invertible, the element is not unisolvent. ∎

REMARK 1.2 (Nodal, hierarchic and dual basis) The expression *node* has in the finite element analysis several different meanings. To begin with, by *nodal* some authors denote the unique basis $\mathcal{B} \subset P$ that satisfies the δ-property (1.6). We will work with *nodal* and *hierarchic* bases of the space P which both satisfy the δ-property. Definitions will be given when appropriate. Existence and uniqueness of a basis $\mathcal{B} \subset P$ satisfying the δ-property is equivalent to the condition that the linear forms $L_1, \ldots, L_{N_P}$ form a *dual basis* to $\mathcal{B}$ in the space P' of linear forms over P. ∎

Example 1.1 (A nonunisolvent element)
Consider a square domain $K = (-1, 1)^2$, polynomial space

$$P = \text{span}\{1, x_1, x_2, x_1 x_2\}$$

and a set of degrees of freedom Σ consisting of linear forms $L_i : P \to \mathbb{R}$ associated with function values at points $[-1,0]$, $[1,0]$, $[0,-1]$ and $[0,1]$, as shown in Figure 1.1.

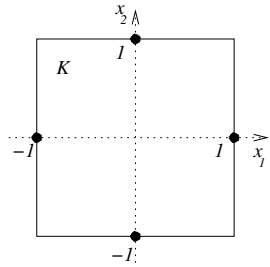

FIGURE 1.1: An example of a nonunisolvent finite element.

Hence, the degrees of freedom L_i are defined by

$$\begin{aligned} L_1(g) &= g(-1,0), \\ L_2(g) &= g(1,0), \\ L_3(g) &= g(0,-1), \\ L_4(g) &= g(0,1), \end{aligned}$$

and the matrix $\boldsymbol{L}$ corresponding to the functions $1, x_1, x_2, x_1x_2$ has the form

$$\boldsymbol{L} = \begin{pmatrix} 1 & -1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & -1 & 0 \\ 1 & 0 & 1 & 0 \end{pmatrix}.$$

Obviously the matrix $\boldsymbol{L}$ is singular and therefore the finite element (K, P, Σ) is *not unisolvent*. It can be easily checked that the finite element becomes unisolvent after using the vertices instead of edge midpoints. □

Example 1.2 (Unisolvent nodal elements)
Consider, for example, the interval $K_a = (-1,1)$ and a space $P_a = P^p(K_a)$ of polynomials of the order at most p over K_a. Let us cover K_a with $N_P = p+1$ points (geometrical nodes) $-1 = X_1 < X_2 < \ldots < X_{N_P} = 1$. These points have to be chosen carefully since their distribution determines the basis functions and consequently the conditioning of the discrete problem. Define the set of degrees of freedom $\Sigma_a = \{L_1, L_2, \ldots, L_{N_P}\}$, $L_i : P_a \to \mathbb{R}$ by

$$L_i(g) = g(X_i) \quad \text{for all } g \in P_a \quad \text{and} \quad i = 1, 2, \ldots, N_P. \tag{1.9}$$

Obviously the forms L_i are linear. Since every polynomial $g \in P_a$ satisfies

$$g(X_1) = g(X_2) = \ldots = g(X_{N_P}) = 0 \;\Rightarrow\; g = 0,$$

the finite element $\mathcal{K}_a = (K_a, P_a, \Sigma_a)$ is unisolvent. It is easy to construct a basis of the space P_a that satisfies the δ-property – in 1D one does not even have to solve systems of linear equations since the Lagrange interpolation polynomial can be exploited (see Section 1.3, formula 1.76).

Nodal elements in higher spatial dimensions are constructed in the same way. Consider, for instance, an equilateral triangle T, $\mathrm{diam}(T) = 2$ and a space $P_T = P^p(T)$ of polynomials of the order at most p over T. One may cover T (for example) with $N_P = (p+1)(p+2)/2$ Gauss-Lobatto points $X_1, X_2, \ldots, X_{N_P}$ as shown in Figure 1.2. Basis functions satisfying the δ-property are constructed by inverting the $N_P \times N_P$ matrix $\boldsymbol{L}$ as described in Remark 1.1.

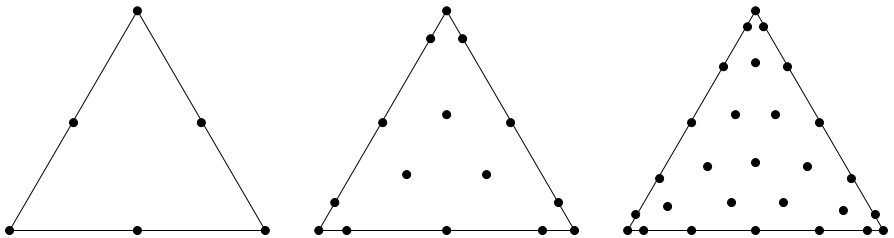

FIGURE 1.2: Nodal points in the equilateral triangle based on Gauss-Lobatto points for $p = 2$ (6 points), $p = 4$ (15 points) and $p = 6$ (28 points) with optimized interpolation properties. For the construction and properties of these geometrical nodes see, e.g., [109].

Finite elements based on nodal values are popular in connection with h-adaptive methods (see, e.g., [109, 111, 110, 112] and others). □

Example 1.3 (Unisolvent hierarchic elements)
Application of hierarchic shape functions represents another major approach to the design of finite elements. Consider a domain K and a space P of polynomials of the order at most p of dimension N_P. Consider a *hierarchic basis* $\mathcal{B}^p = \{\theta_1, \theta_2, \ldots, \theta_{N_p}\}$ in the space P. By hierarchic we mean that

$$\mathcal{B}^p \subset \mathcal{B}^{p+1}$$

for every p. Every polynomial $g \in P$ can be uniquely expressed as a linear

combination

$$g = \sum_{i=1}^{N_p} \beta_i \theta_i = \sum_{i=1}^{N_p} L_i(g)\theta_i, \tag{1.10}$$

where β_i are real coefficients and $L_i(g) = \beta_i$ linear forms

$$L_i : P \to \mathbb{R}, \quad i = 1, 2, \ldots, N_p. \tag{1.11}$$

Obviously the choice $\Sigma = \{L_1, L_2, \ldots, L_{N_P}\}$ yields a unisolvent finite element $(\mathcal{K}, P, \Sigma)$, and by definition the hierarchic basis $\mathcal{B}$ has the δ-property (1.6).

Hierarchic elements allow for locally nonuniform distribution of the order of polynomial approximation more easily than nodal elements, which makes them suitable for *p*- and *hp*-adaptivity. We will pursue the hierarchic approach in this book. ∎

REMARK 1.3 (Selection of finite elements) The choice of a finite element (or their combination) depends upon the problem solved and upon the expectations that we put into the finite element scheme. Obviously it has a crucial effect on the behavior of the finite element scheme (see, for example, comparison of *conditioning properties* of various types of elements in Section 1.3). Several examples of standard as well as exotic finite elements were collected in [38]. ∎

1.1.3 Finite element mesh

We assume that the bounded domain Ω with a Lipschitz-continuous boundary, where the underlying problem is investigated, is approximated by a computational domain Ω_h whose boundary is piecewise-polynomial.

DEFINITION 1.4 (Finite element mesh) Finite element mesh $\mathcal{T}_{h,p} = \{K_1, K_2, \ldots, K_M\}$ *over a domain* $\Omega_h \subset \mathbb{R}^d$ *with a piecewise-polynomial boundary is a geometrical division of* Ω_h *into a finite number of nonoverlapping (curved) open polygonal cells* K_i *such that*

$$\Omega_h = \bigcup_{i=1}^{M} \overline{K}_i.$$

Each cell K_i, $1 \le i \le M$ *is equipped with a polynomial order* $1 \le p(K_i) = p_i$.

DEFINITION 1.5 (Hybrid mesh) *If various types of cells are combined, the mesh is called* hybrid.

DEFINITION 1.6 (Regular mesh) *The mesh is called* regular *if for any two elements K_i and K_j, $i \neq j$ only one of the following alternatives holds:*
- $\overline{K}_i \cup \overline{K}_j$ *is empty,*
- $\overline{K}_i \cup \overline{K}_j$ *is a single common vertex,*
- $\overline{K}_i \cup \overline{K}_j$ *is a single (whole) common edge,*
- $\overline{K}_i \cup \overline{K}_j$ *is a single (whole) common face.*

By assuming that the mesh is regular we avoid *hanging nodes*, the existence of which substantially complicates the discretization procedure. In this book we will use the word *node* as an abstraction for vertices, edges, faces and element interiors (the reasons for this soon become clear). Basically, hanging nodes can be grid vertices which lie in the interior of an edge or face of another cell, grid edges which lie in the interior of an edge or of a face of another cell, and grid faces which lie in the interior of a face of another cell. The constrained approximation technique in 2D and 3D will be discussed in more detail in Section 3.6. Various constellations involving hanging nodes are illustrated in Figure 1.3.

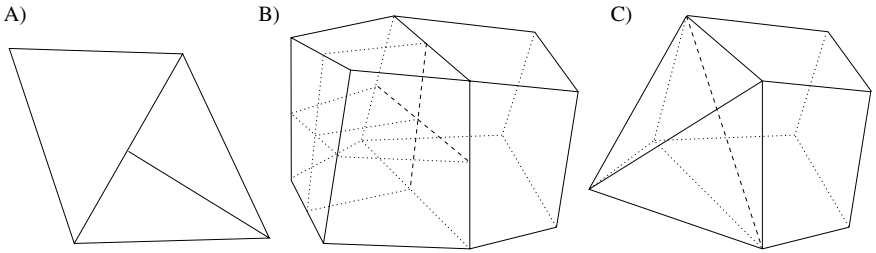

FIGURE 1.3: Examples of hanging nodes. A) Single hanging vertex and two edges in a 2D mesh. B) Five hanging vertices, twelve edges and four faces in a 3D mesh. C) Single hanging edge and two faces in a 3D hybrid mesh.

We will consider all of the most commonly used types of cells: one-dimensional elements, triangles and quadrilaterals in two spatial dimensions and tetrahedra, prisms and hexahedra in 3D.

1.1.4 Finite element interpolants and conformity

In Paragraph 1.1.2 we introduced *unisolvency* of the finite element (K, P, Σ) as another expression for the compatibility of the set of degrees of freedom Σ with the polynomial space P. Now we will address the compatibility of finite elements with spaces of functions where they will be used for approximation purposes – their *conformity* to these spaces.

The notion of conformity of finite elements to spaces of functions is tightly

connected with their interpolation properties in these spaces.

Finite element interpolant

In the beginning let us assume that the degrees of freedom $L_1, L_2, \ldots, L_{N_P}$ of the finite element (K, P, Σ) are defined in a larger Hilbert space $V(K)$, $P \subset V(K)$.

DEFINITION 1.7 (Finite element interpolant) *Given a unisolvent finite element (K, P, Σ), let $\mathcal{B} = \{\theta_1, \theta_2, \ldots, \theta_{N_P}\}$ be the unique basis of the space P satisfying the δ-property (1.6). Let $v \in V$, where $P \subset V$, be a function for which all the linear forms $L_1, L_2, \ldots, L_{N_P}$ are defined. We define the (local)* interpolant *as*

$$\mathcal{I}_K(v) = \sum_{i=1}^{N_P} L_i(v)\theta_i. \tag{1.12}$$

It follows immediately from the linearity of the forms L_i that the interpolation operator $\mathcal{I}_K : V \to P$ is linear.

PROPOSITION 1.1
Let (K, P, Σ) be a unisolvent finite element and let $v \in V$, $P \subset V$ be a function for which all the linear forms $L_1, L_2, \ldots, L_{N_P}$ are defined. Then

$$L_i(\mathcal{I}_K(v)) = L_i(v), \quad 1 \le i \le N_P. \tag{1.13}$$

PROOF It follows immediately from Definition 1.7 and the δ-property (1.6) that

$$L_i\left(\sum_{j=1}^{N_P} L_j(v)\theta_j\right) = \sum_{j=1}^{N_P} L_j(v)L_i(\theta_j) = L_i(v).$$

∎

PROPOSITION 1.2
Let (K, P, Σ) be a unisolvent finite element. The finite element interpolation operator $\mathcal{I}_K$ is idempotent*,*

$$\mathcal{I}_K^2 = \mathcal{I}_K. \tag{1.14}$$

PROOF It follows immediately from Proposition 1.1 that

$$\mathcal{I}_K(v) = v, \quad v \in P. \tag{1.15}$$

Hence, for all $v \in V$, where $P \subset V$, it is

$$\mathcal{I}_K(\underbrace{\mathcal{I}_K(v)}_{\in P}) = \mathcal{I}_K(v)$$

which was to be shown. ∎

Conformity of finite elements to spaces of functions

Let $V(\Omega_h)$ be a general Hilbert space of functions and let the computational domain Ω_h be covered with a finite element mesh $\mathcal{T}_{h,p}$. Global finite element interpolant $\mathcal{I}$ is defined elementwise by means of local element interpolants $\mathcal{I}_{K_i}$,

$$\mathcal{I}(v)|_{K_i} \equiv \mathcal{I}_{K_i} \quad \text{for all } i = 1, 2, \ldots, M. \tag{1.16}$$

It is natural to request that for all functions $v \in V$ also $\mathcal{I}(v) \in V$ whenever $\mathcal{I}(v)$ is defined.

Most commonly used definitions of conformity cover neither finite element meshes that comprise various types of finite elements nor hierarchic elements which are the main issue in this book. Therefore we will speak about the conformity of finite elements in the sense of conformity of *whole finite element meshes*. This notion naturally reduces to the standard *conformity of finite elements* if all the finite elements in the mesh are of the same type.

DEFINITION 1.8 (Conformity of finite elements) *Let $\mathcal{T}_{h,p}$ be a finite element mesh consisting of M unisolvent finite elements (K_i, P_i, Σ_i), $i = 1, 2, \ldots, M$. Let $V(\Omega_h)$ be a Hilbert space of functions and $\mathcal{I}_{K_i} : V(K_i) \to P_i$ the (local) finite element interpolation operators. We say that the finite element mesh $\mathcal{T}_{h,p}$ is conforming to the space V if and only if there exists a subspace $V^*(\Omega_h) \subset V(\Omega_h)$, which is dense in V (i.e., $\overline{V^*} = V$), such that for each function $v \in V^*(\Omega_h)$ the corresponding global interpolant $\mathcal{I}(v)$ is defined and lies in the space $V(\Omega_h)$.*

Global conformity requirements for the most commonly used Hilbert spaces H^1, $\boldsymbol{H}(\mathrm{curl})$ and $\boldsymbol{H}(\mathrm{div})$ are formulated in the following Lemmas 1.1 – 1.3 (see also, e.g., [143, 166, 159]).

LEMMA 1.1 (Conformity requirements of the space H^1) *Consider a domain $\Omega_h \subset \mathbb{R}^d$ covered with a finite element mesh $\mathcal{T}_{h,p}$. A function $v : \Omega_h \to \mathbb{R}$ belongs to $H^1(\Omega_h)$ if and only if*

1. *$v|_K \in H^1(K)$ for each element $K \in \mathcal{T}_{h,p}$,*
2. *for each common face $f = \overline{K}_1 \cap \overline{K}_2$, $K_1, K_2 \in \mathcal{T}_{h,p}$ the trace of $v|_{K_1}$ and $v|_{K_2}$ on f is the same.*

PROOF Using *1.*, define the functions $w_j \in L^2(\Omega_h)$, $j = 1, 2, \ldots, d$ as

$$w_j|_K = D_j(v|_K)$$

for all $K \in \mathcal{T}_{h,p}$. We will show that $v \in H^1(\Omega_h)$ by simply verifying that $w_j = D_j v$.

Using the Green theorem we have for every $\varphi \in \mathcal{D}(\Omega_h)$

$$\int_{\Omega_h} w_j \varphi = \sum_{K \in \mathcal{T}_{h,p}} \int_K w_j \varphi = -\sum_K \int_K (v|_K) D_j \varphi + \sum_K \int_{\partial K} v|_K \varphi \nu_{K,j},$$

where ν_K is the outward normal vector to K on ∂K. The symbol $\mathcal{D}(\Omega_h)$ stands for the space of *distributions* over Ω_h, where distributions are infinitely smooth functions $\varphi : \mathbb{R}^d \to \mathbb{R}$ whose support lies within the domain Ω_h. Since φ is vanishing on $\partial\Omega_h$ and $\nu_{K_1} = -\nu_{K_2} = \nu$ on the common face f, we have by *2.*

$$\begin{aligned}\int_{\Omega_h} w_j \varphi &= -\int_{\Omega_h} v D_j \varphi + \sum_{f, f = \overline{K}_1 \cap \overline{K}_2, K_1, K_2 \in \mathcal{T}_{h,p}} \int_f (v|_{K_1} - v|_{K_2}) \varphi \nu_j \\ &= -\int_{\Omega_h} v D_j \varphi,\end{aligned}$$

and thus $w_j = D_j v$.

Conversely, if we assume that $v \in H^1(\Omega_h)$, it follows at once that *1.* holds. Using further $w_j = D_j v$, in the same way as before we obtain that

$$\sum_{f, f = \overline{K}_1 \cap \overline{K}_2, K_1, K_2 \in \mathcal{T}_{h,p}} \int_f (v|_{K_1} - v|_{K_2}) \varphi \nu_j = 0$$

for all $\varphi \in \mathcal{D}(\Omega_h)$, $j = 1, 2, \ldots, d$. Hence, *2.* is satisfied. ∎

LEMMA 1.2 (Conformity requirements of the space $\boldsymbol{H}(\mathrm{div})$) *Consider a domain $\Omega_h \subset \mathbb{R}^d$ covered with a finite element mesh $\mathcal{T}_{h,p}$. Consider a function $\boldsymbol{v} : \Omega_h \to \mathbb{R}^d$ such that*

1. *$\boldsymbol{v}|_K \in [H^1(K)]^d$ for each element $K \in \mathcal{T}_{h,p}$,*
2. *for each common face $f = \overline{K}_1 \cap \overline{K}_2$, $K_1, K_2 \in \mathcal{T}_{h,p}$ the trace of the normal component $\boldsymbol{n} \cdot \boldsymbol{v}|_{K_1}$ and $\boldsymbol{n} \cdot \boldsymbol{v}|_{K_2}$ on f is the same (here $\boldsymbol{n}$ is a unique normal vector to the face f).*

Then $\boldsymbol{v} \in \boldsymbol{H}(\mathrm{div})$. On the other hand, if $\boldsymbol{v} \in \boldsymbol{H}(\mathrm{div})$ and 1. holds, then 2. is satisfied.

PROOF Define $w \in L^2(\Omega_h)$ by

$$w|_K = \mathrm{div}(\boldsymbol{v}|_K)$$

for all $K \in \mathcal{T}_{h,p}$. The Green formula implies for every $\varphi \in \mathcal{D}$

$$\begin{aligned}(\mathrm{div}\boldsymbol{v}, \varphi) &= -\int_{\Omega_h} \boldsymbol{v} \cdot \nabla\varphi = -\sum_{K\in\mathcal{T}_{h,p}} \int_K (\boldsymbol{v}|_K) \cdot \nabla\varphi \\ &= \sum_{K\in\mathcal{T}_{h,p}} \int_K \mathrm{div}(\boldsymbol{v}|_K)\varphi \\ &\quad - \sum_{f, f=\overline{K}_1\cap\overline{K}_2,\ K_1,K_2\in\mathcal{T}_{h,p}} \int_f (\boldsymbol{n}\cdot\boldsymbol{v}|_{K_1} - \boldsymbol{n}\cdot\boldsymbol{v}|_{K_2})\varphi = \int_{\Omega_h} w\varphi\end{aligned}$$

and therefore $\mathrm{div}\boldsymbol{v} = w$.

Conversely, if $\boldsymbol{v} \in \boldsymbol{H}(\mathrm{div})$, we have $w = \mathrm{div}\boldsymbol{v}$. Since $\boldsymbol{v}|_K \in [H^1(\Omega_h)]^d$, the trace on f is well defined and we obtain

$$\sum_{f, f=\overline{K}_1\cap\overline{K}_2, K_1,K_2\in\mathcal{T}_{h,p}} \int_f (\boldsymbol{n}\cdot\boldsymbol{v}|_{K_1} - \boldsymbol{n}\cdot\boldsymbol{v}|_{K_2})\varphi = 0$$

for all $\varphi \in \mathcal{D}$. Hence *1.* holds. ∎

LEMMA 1.3 (Conformity requirements of the space $\boldsymbol{H}$(curl)) *Consider a domain $\Omega_h \subset \mathbb{R}^d$ covered with a finite element mesh $\mathcal{T}_{h,p}$. Consider a function $\boldsymbol{v} : \Omega_h \to \mathbb{R}^d$ such that*

1. *$\boldsymbol{v}|_K \in [H^1(K)]^d$ for each element $K \in \mathcal{T}_{h,p}$,*
2. *for each common face $f = \overline{K}_1 \cap \overline{K}_2$, $K_1, K_2 \in \mathcal{T}_{h,p}$ the trace of the tangential component $\boldsymbol{n} \times \boldsymbol{v}|_{K_1}$ and $\boldsymbol{n} \times \boldsymbol{v}|_{K_2}$ on f is the same (again, $\boldsymbol{n}$ is a unique normal vector to the face f).*

Then $\boldsymbol{v} \in \boldsymbol{H}$(curl). On the other hand, if $\boldsymbol{v} \in \boldsymbol{H}$(curl) and 1. holds, then 2. is satisfied.

PROOF Similar to the previous case. ∎

REMARK 1.4 Although the conditions declared in Lemmas 1.2 and 1.3 are weaker than those associated with the space H^1, their algorithmic realization is more demanding. More about $\boldsymbol{H}$(curl)- and $\boldsymbol{H}$(div)-conforming approximations will follow. ∎

REMARK 1.5 (Conformity requirements of the space L^2) There are *no continuity requirements* on interelement boundaries in L^2-conforming

approximations. See, e.g., [143, 166, 159] for additional conformity properties of finite elements. ∎

REMARK 1.6 (Interpolation on hierarchic elements) Notice that Definition 1.7 says nothing about the interpolation on hierarchic elements since the degrees of freedom L_i (coefficients β_i in (1.10)) are undefined when $v \notin P$. Definition of local interpolation operators on hierarchic elements requires deeper mathematical analysis. Since the standard Lagrange interpolation has to be combined with *projection* onto hierarchically constructed subspaces of the space P, sometimes the technique is called *projection-based interpolation*. We find it appropriate to postpone the discussion of this issue to Chapter 3 where relevant machinery will be in place. The reader does not need to worry in the meantime since, of course, the projection-based interpolation operators will be compatible with the global conformity requirements presented in Lemmas 1.1 – 1.3. ∎

Example 1.4 (A conforming and a nonconforming element)
We find it useful to present a very simple example where all ideas presented in this paragraph can be fixed. Let us consider linear triangular finite elements of the (i) Lagrange and (ii) Crouzeix-Raviart type (for the latter see [56]). Both of them are defined on a triangular domain K, using the space of linear polynomials $P(K)$. The degrees of freedom $L_1, L_2, \ldots, L_{N_P}$ are associated with (i) element vertices $\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}$ and (ii) midpoints $\boldsymbol{d}, \boldsymbol{e}, \boldsymbol{f}$ of the edges (as shown in Figure 1.4).

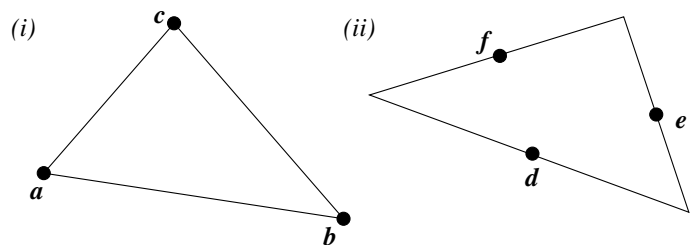

FIGURE 1.4: Nodal points for (i) linear Lagrange and (ii) Crouzeix-Raviart elements.

In both cases $\dim(P) = 3$. For the Lagrange element the degrees of freedom $L_i : P \to \mathbb{R}$ are defined as

$$L_1(g) = g(\boldsymbol{a}),\ L_2(g) = g(\boldsymbol{b}),\ L_3(g) = g(\boldsymbol{c}),$$

and unisolvency check (see Remark 1.1) yields a matrix $\boldsymbol{L}$ in the form

$$\boldsymbol{L}^{Lagr} = \begin{pmatrix} 1 & a_1 & a_2 \\ 1 & b_1 & b_2 \\ 1 & c_1 & c_2 \end{pmatrix}. \tag{1.17}$$

Analogously, in the Crouzeix-Raviart case we have the linear forms

$$L_1(g) = g(\boldsymbol{d}),\ L_2(g) = g(\boldsymbol{e}),\ L_3(g) = g(\boldsymbol{f}) \tag{1.18}$$

and

$$\boldsymbol{L}^{C-R} = \begin{pmatrix} 1 & d_1 & d_2 \\ 1 & e_1 & e_2 \\ 1 & f_1 & f_2 \end{pmatrix}. \tag{1.19}$$

Obviously both the matrices $\boldsymbol{L}^{Lagr}, \boldsymbol{L}^{C-R}$ are regular unless the triangle K is degenerated (and this is forbidden since K is a domain in $\mathbb{R}^2$).

Consider now a finite element mesh $\mathcal{T}_{h,p} = \{K_1, K_2\}$ consisting of two elements as illustrated in Figure 1.5.

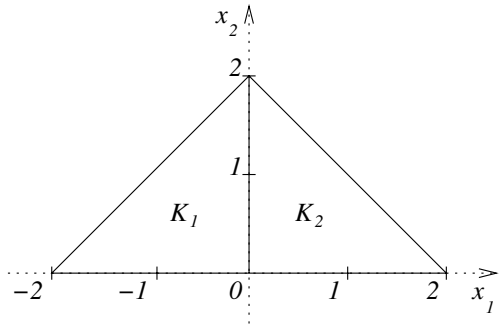

FIGURE 1.5: Sample mesh consisting of two triangular elements.

In the Lagrange case, the corresponding two sets of basis functions that satisfy the δ-property (1.6) are obtained after inverting the matrices $\boldsymbol{L}_{K_1}^{Lagr}$ and $\boldsymbol{L}_{K_2}^{Lagr}$ (see Remark 1.1). We obtain

$$\theta_1^{(1)} = -x_1/2,\ \ \theta_2^{(1)} = (x_1 - x_2 + 2)/2,\ \ \theta_3^{(1)} = x_2/2,$$
$$\theta_1^{(2)} = (2 - x_1 - x_2)/2,\ \ \theta_2^{(2)} = x_1/2,\ \ \theta_3^{(2)} = x_2/2.$$

Hence the interpolation operators $\mathcal{I}_{K_1}^{Lagr} : V(K_1) \to P(K_1)$ and $\mathcal{I}_{K_2}^{Lagr} : V(K_2) \to P(K_2)$ (Definition 1.7 with $V = H^1$ and $V^* = C$) have the form

$$\mathcal{I}_{K_1}^{Lagr}(v) = v(-2,0)\theta_1^{(1)} + v(0,0)\theta_2^{(1)} + v(0,2)\theta_3^{(1)},$$
$$\mathcal{I}_{K_2}^{Lagr}(v) = v(0,0)\theta_1^{(2)} + v(2,0)\theta_2^{(2)} + v(0,2)\theta_3^{(2)}.$$

Analogously in the Crouzeix-Raviart case the inversion of the matrices $\boldsymbol{L}_{K_1}^{C-R}$ and $\boldsymbol{L}_{K_2}^{C-R}$ yields the nodal bases

$$\theta_1^{(1)} = 1 - x_2, \quad \theta_2^{(1)} = x_1 + 1, \quad \theta_3^{(1)} = -(x_1 - x_2 + 1),$$
$$\theta_1^{(2)} = 1 - x_2, \quad \theta_2^{(2)} = x_1 + x_2 - 1, \quad \theta_3^{(2)} = 1 - x_1$$

and the local interpolation operators

$$\mathcal{I}_{K_1}^{C-R}(v) = v(-1,0)\theta_1^{(1)} + v(0,1)\theta_2^{(1)} + v(-1,1)\theta_3^{(1)},$$
$$\mathcal{I}_{K_2}^{C-R}(v) = v(1,0)\theta_1^{(2)} + v(1,1)\theta_2^{(2)} + v(0,1)\theta_3^{(2)}.$$

Consider now a function

$$v(x_1, x_2) = (x_1 - x_2)^2 \in C(\overline{K}_1 \cup \overline{K}_2).$$

The pair of the corresponding piecewise-linear global interpolants are depicted in Figures 1.6 and 1.7.

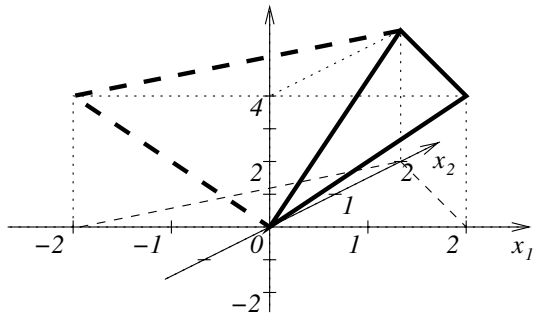

FIGURE 1.6: Continuous piecewise-linear Lagr. interpolant on $\overline{K}_1 \cup \overline{K}_2$.

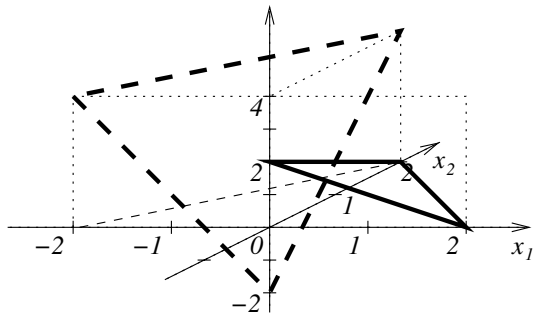

FIGURE 1.7: Discontinuous piecewise-linear Crouzeix-Raviart interpolant on $\overline{K}_1 \cup \overline{K}_2$. This figure illustrates that the finite element of Crouzeix-Raviart does not conform to the space H^1.

Obviously one example is not enough to prove *conformity* – one has to consider each pair of adjacent elements $K_i, K_j \in \mathcal{T}_{h,p}$ and show that the interpolant of all functions $v \in V(\overline{K}_i \cup \overline{K}_j)$ still lies in the space $V(\overline{K}_i \cup \overline{K}_j)$, using global conformity requirements of the space V (see Lemmas 1.1 – 1.3). In the linear Lagrange case, the coincidence of values of the function v at the pair of shared vertices together with the linearity of the local interpolant on element edges imply the continuity along the whole shared edge of K_i, K_j. Hence,

$$v \in V(\overline{K}_1 \cup \overline{K}_2) \;\Rightarrow\; \mathcal{I}^{Lagr}(v) \in V(\overline{K}_1 \cup \overline{K}_2)$$

holds for all $v \in H^1(\overline{K}_1 \cup \overline{K}_2)$, therefore the linear Lagrange finite element is conforming to the space $H^1(\overline{K}_1 \cup \overline{K}_2)$. ∎

REMARK 1.7 (Nonconforming elements) For selected types of problems, nonconforming finite elements are used with excellent results. We refer to [45, 56, 90, 106, 121, 124, 125, 165] to mention at least a couple of examples. In this book we confine ourselves to conforming finite element approximations only. ∎

REMARK 1.8 (Interpolation error estimates) At this point the next logical step would be to introduce local *element interpolation error estimates*. However, for all standard types of nodal elements this can be found in standard finite element textbooks. Relevant for our purposes are *projection-based interpolation* error estimates for *hierarchic elements* in spaces H^1, $\boldsymbol{H}(\mathrm{curl})$ and $\boldsymbol{H}(\mathrm{div})$. We will address them in Chapter 3. ∎

1.1.5 Reference domains and reference maps

For piecewise-linear approximation, degrees of freedom are usually associated with the solution values at the grid vertices (here the nodal and hierarchic approaches coincide), and the variational formulation can be evaluated directly in the grid.

The situation changes dramatically with higher-order finite elements since they use a large amount of overlapping information, whose efficient management requires more structure to be imposed. For example, while first-order numerical quadrature can be implemented using coordinates of grid vertices only, higher-order quadrature schemes require many integration points per element (the actual amount depends on the order of accuracy, and in 2D and 3D it easily achieves several hundred). Both storing these values (d spatial coordinates and a weight per point) in all elements as well as reconstructing them periodically from a reference configuration would be extremely inefficient. The situation is analogous for higher-order basis functions.

Therefore, for higher-order finite element discretizations the mesh cells $K_i \in \mathcal{T}_{h,p}$ are mapped onto a *reference domain* $\hat{K}$ by means of smooth bijective *reference maps*

$$\boldsymbol{x}_{K_i} : \hat{K} \to K_i. \tag{1.20}$$

The maps $\boldsymbol{x}_K$, together with polynomial spaces on the reference domain $\hat{K}$, will be used for the definition of the space of functions $V_{h,p}(\Omega_h)$ where the finite element solution will be sought. An example of a reference map in the quadrilateral case in 2D is illustrated in Figure 1.8.

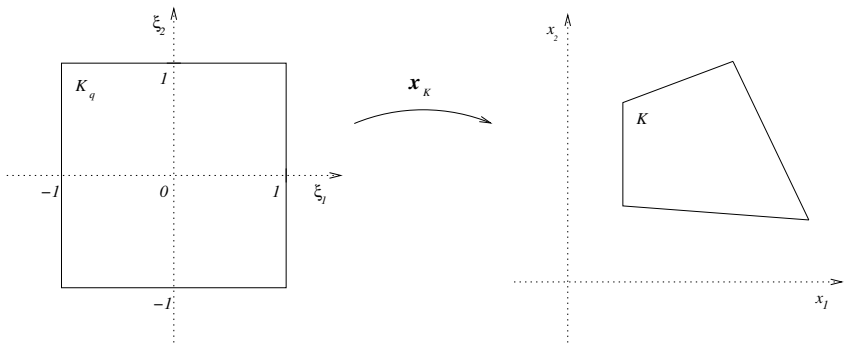

FIGURE 1.8: Reference map for a quadrilateral element.

The design of reference maps for all standard types of reference domains will be discussed in detail in Chapter 3. There we also show how one uses them to transfer the integrals over elements K_i from the variational formulation to the reference domains. The higher-order finite element discretization is performed almost exclusively on the reference domains.

1.1.6 Finite element discretization

Consider a bounded domain $\Omega \subset \mathbb{R}^d$ with Lipschitz-continuous boundary, a partial differential equation (PDE) to be solved, and a set of conventional boundary conditions. Multiplying the PDE with a test function v from a suitable function space V, integrating over the domain Ω, applying the Green's theorem and incorporating the boundary conditions, one obtains a variational formulation

$$L(u^* + \bar{u}, v) = f(v) \quad \text{for all } v \in V. \tag{1.21}$$

Both the forms L and f are assumed linear in v. If nonhomogeneous Dirichlet conditions are present, the solution $u = u^* + \bar{u}$ is sought in an affine function space which is different from V (functions satisfying nonhomogeneous

Dirichlet boundary conditions obviously cannot constitute any linear function space). The *lift function* u^* is chosen to satisfy nonhomogeneous Dirichlet conditions. Only the unknown component $\bar{u}$ satisfying *homogeneous* Dirichlet boundary conditions is sought. All functions $v \in V$ vanish on any Dirichlet part of the boundary $\partial\Omega$.

Neumann and Newton (Robin) boundary conditions are enforced by substituting them directly into boundary integrals in (1.21) over the corresponding part of the boundary $\partial\Omega$. See any basic finite element textbook for more details. An example of higher-order finite element discretization in 1D involving nonhomogeneous Dirichlet boundary conditions will be given in Section 1.3.

Approximation of weak forms and discretization

The finite element discretization of (1.21) is done in the following standard steps:

Step 1: Approximate the domain Ω with another domain Ω_h which is more convenient for meshing and computation.

Step 2: Cover the domain Ω_h with a finite element mesh $\mathcal{T}_{h,p}$. Choose appropriate reference domains for all geometrical types of elements and for each $K \in \mathcal{T}_{h,p}$ construct a smooth bijective reference map $\boldsymbol{x}_K$.

Step 3: Approximate the space V, using appropriate polynomial spaces on the reference domains and the reference maps, by a suitable subspace $V_{h,p} = \mathrm{span}(v_1, v_2, \ldots, v_N)$.

REMARK 1.9 (Variational crimes) Let us remark that in reality usually $V_{h,p} \not\subset V$ since the domains Ω and Ω_h differ, and moreover often even $\Omega_h \not\subset \Omega$. Hence this step is sometimes classified as a *variational crime* in the FE community. ∎

Step 4: Approximate the form L by another form $L_{h,p}$, replacing the exact integration over Ω and $\partial\Omega$ by numerical integration over Ω_h and $\partial\Omega_h$. Notice that boundary conditions are shifted from $\partial\Omega$ to $\partial\Omega_h$ in this step. The question of an optimal choice of quadrature schemes will be addressed in Chapter 4.

Step 5: Approximate the linear form f by another linear form $f_{h,p}$ in the same way as in Step 4.

Step 6: We arrive at a new, approximate variational formulation: The solution $u_{h,p}$ is sought in the form $u_{h,p} = u^*_{h,p} + \bar{u}_{h,p}$, $\bar{u}_{h,p} \in V_{h,p}$, satisfying

$$L_{h,p}(u^*_{h,p} + \bar{u}_{h,p}, v_{h,p}) = f_{h,p}(v_{h,p}), \tag{1.22}$$

for all $v_{h,p} \in V_{h,p}$. The function $u^*_{h,p}$ is a suitable piecewise polynomial (usually a simple piecewise-linear) approximation of the lift function u^* from (1.21).

Step 7: Express the function $\bar{u}_{h,p}$ as a linear combination of the basis functions v_i of the space $V_{h,p}$ with unknown coefficients y_i,

$$u_{h,p}(\boldsymbol{x}) = u^*_{h,p}(\boldsymbol{x}) + \bar{u}_{h,p}(\boldsymbol{x}) = u^*_{h,p}(\boldsymbol{x}) + \sum_{j=1}^{N} y_j v_j(\boldsymbol{x}). \tag{1.23}$$

Step 8: Insert the construction (1.23) into the approximate weak form (1.22) and select $v_{h,p} := v_i$, $i = 1, 2, \ldots, N$. This turns (1.22) into a system of algebraic equations

$$L_{h,p}\left(u^*_{h,p} + \sum_{j=1}^{N} y_j v_j, v_i\right) = f_{h,p}(v_i), \quad i = 1, 2, ..., N. \tag{1.24}$$

If the form $L_{h,p}$ is bilinear, (1.24) represents a system of linear algebraic equations which can be written in a matrix form $\boldsymbol{SY} = \boldsymbol{F}$,

$$\sum_{j=1}^{N} \underbrace{L_{h,p}\left(v_j, v_i\right)}_{\boldsymbol{S}_{ij}} \underbrace{y_j}_{\boldsymbol{Y}_j} = \underbrace{f_{h,p}(v_i) - L_{h,p}\left(u^*_{h,p}\right)}_{\boldsymbol{F}_i}, \quad i = 1, 2, ..., N.$$

Otherwise the algebraic system (1.24) is nonlinear.

Step 9: Solve the system (1.24) for the unknown coefficients $\boldsymbol{Y}$ of $\bar{u}_{h,p}$ with a suitable numerical scheme. Retrieve the approximate solution $u_{h,p}$ using (1.23). Numerical methods for the treatment of discrete problems will be discussed in Chapter 5.

1.1.7 Method of lines for evolutionary problems

The method of lines (MOL) is one of the most popular tools for the solution of evolutionary PDEs. The basic idea is to perform the discretization in space only while keeping the time-variable continuous. This is achieved by expressing the approximate solution $u_{h,p}(\boldsymbol{x}, t)$ in a form analogous to (1.23), *with time-dependent coefficients* $y_i = y_i(t)$:

$$u_{h,p}(\boldsymbol{x}, t) = u^*_{h,p}(\boldsymbol{x}) + \bar{u}_{h,p}(x, t) = u^*_{h,p}(\boldsymbol{x}) + \sum_{j=1}^{N} y_j(t) v_j(\boldsymbol{x}). \tag{1.25}$$

Thus, instead of a system of algebraic equations (1.24) we end up with a system of *ordinary differential equations*.

For details on the MOL itself as well as for error estimates for evolutionary problems solved by this method see, e.g., [13, 16, 78, 79, 80, 144, 155, 174, 175, 182, 194, 201, 198] and others.

1.2 Orthogonal polynomials

Orthogonal polynomials find applications in diverse fields of mathematics, both for theoretical and numerical issues. In our case they will play an essential role in the design of optimal higher-order shape functions. For additional information on orthogonal polynomials see, e.g., [192], which is usually referred to as a basic textbook on this subject.

1.2.1 The family of Jacobi polynomials

The class of Jacobi polynomials,

$$P_{n,\alpha,\beta}(x) = \frac{(-1)^n}{2^n n!}(1-x)^{-\alpha}(1+x)^{-\beta}\frac{\mathrm{d}^n}{\mathrm{d}x^n}\left[(1-x)^{\alpha+n}(1+x)^{\beta+n}\right], \quad (1.26)$$

holds the prominent position among orthogonal polynomials. It satisfies the Jacobi differential equation

$$(1-x^2)\frac{\mathrm{d}^2}{\mathrm{d}x^2}P_{n,\alpha,\beta}+(\beta-\alpha-(\beta+\alpha+2)x)\frac{\mathrm{d}}{\mathrm{d}x}P_{n,\alpha,\beta}+n(n+\alpha+\beta+1)P_{n,\alpha,\beta}=0 \quad (1.27)$$

($\alpha, \beta > -1$ are real parameters). Let $L^2_{\alpha,\beta}(I)$, where $I = (-1, 1)$, denote the space of all functions which are square integrable in I with the weight

$$w_{\alpha,\beta}(x) = (1-x)^{\alpha}(1+x)^{\beta} \quad (1.28)$$

and with the corresponding norm

$$\|u\|^2_{L^2_{\alpha,\beta}} = \int_{-1}^{1} |u|^2 w_{\alpha,\beta}\mathrm{d}x. \quad (1.29)$$

Then every $u \in L^2_{\alpha,\beta}(I)$ can be expanded into the series

$$u(x) = \sum_{n=0}^{\infty} c_n P_{n,\alpha,\beta}(x) \quad \text{satisfying} \quad \lim_{k\to\infty} \|u - \sum_{n=0}^{k} c_n P_{n,\alpha,\beta}\|_{L^2_{\alpha,\beta}(I)} = 0. \quad (1.30)$$

Orthogonality of the Jacobi polynomials is exactly specified by

$$\int_{-1}^{1} P_{n,\alpha,\beta} P_{m,\alpha,\beta} (1-x)^{\alpha} (1+x)^{\beta} \mathrm{d}x = \begin{cases} e_{n,\alpha,\beta} \text{ for } n = m, \\ \\ 0 \quad \text{otherwise} \end{cases} \tag{1.31}$$

where

$$e_{n,\alpha,\beta} = \frac{2^{\alpha+\beta+1}}{2n+\alpha+\beta+1} \frac{\Gamma(\alpha+n+1)\Gamma(\beta+n+1)}{\Gamma(n+1)\Gamma(\alpha+\beta+n+1)}. \tag{1.32}$$

Here Γ is the standard Γ-function. The coefficients c_n are computed using the relation

$$c_n = \frac{1}{e_{n,\alpha,\beta}} \int_{-1}^{1} (1-x)^{\alpha} (1+x)^{\beta} P_{n,\alpha,\beta}(x) u(x) \, \mathrm{d}x. \tag{1.33}$$

We have the relation

$$\frac{\mathrm{d}^k}{\mathrm{d}x^k} P_{n,\alpha,\beta}(x) = 2^{-k} \frac{\Gamma(n+k+\alpha+\beta+1)}{\Gamma(n+\alpha+\beta+1)} P_{n-k,\alpha+k,\beta+k}. \tag{1.34}$$

Ultraspherical polynomials

The ultraspherical polynomials are a special case of the Jacobi polynomials, defined by

$$U_{n,\alpha} = P_{n,\alpha,\alpha}, \quad n = 0, 1, 2 \ldots. \tag{1.35}$$

They inherit all basic properties from the Jacobi polynomials (1.26).

Gegenbauer polynomials

Putting $\alpha = \beta = \nu - 1/2$ in (1.26), the Jacobi polynomials come over to the Gegenbauer polynomials

$$G_n^{\nu}(x) = \frac{\Gamma(n+2\nu)\Gamma(\nu+1/2)}{\Gamma(2\nu)\Gamma(n+\nu+1/2)} P_{n,\nu-1/2,\nu-1/2}(x), \tag{1.36}$$

which again inherit all basic properties of the Jacobi polynomials.

Chebyshev polynomials

The Chebyshev polynomials are another special case of the Jacobi polynomials (1.26) and inherit all of their basic properties. Putting $\alpha = \beta = -1/2$ we obtain

$$C_n(x) = \frac{2^{2n}(n!)^2}{(2n!)} P_{n,-1/2,-1/2}(x). \tag{1.37}$$

1.2.2 Legendre polynomials

Of special importance among the descendants of the Jacobi polynomials $P_{n,\alpha,\beta}$ are the *Legendre polynomials*, defined as

$$L_n(x) = P_{n,0,0}(x). \tag{1.38}$$

They form an orthonormal basis of the space $L^2(I)$. Originally, they were constructed by means of the Gram-Schmidt orthogonalization process, and later many useful properties of these polynomials were found. For all of them let us mention, e.g., that their roots are identical with integration points for higher-order Gauss quadrature rules in one spatial dimension. They satisfy the Legendre differential equation

$$(1-x^2)\frac{\mathrm{d}^2 y}{\mathrm{d}x^2} - 2x\frac{\mathrm{d}y}{\mathrm{d}x} + k(k+1)y = 0. \tag{1.39}$$

There are several ways to define them, among which probably the most useful for the implementation of higher-order shape functions is the recurrent definition

$$\begin{aligned} L_0(x) &= 1, \\ L_1(x) &= x, \\ L_k(x) &= \frac{2k-1}{k}xL_{k-1}(x) - \frac{k-1}{k}L_{k-2}(x), \quad k = 2, 3, \ldots, \end{aligned} \tag{1.40}$$

but they can be defined also by the differential relation

$$L_k(x) = \frac{1}{2^k k!}\frac{\mathrm{d}^k}{\mathrm{d}x^k}\left(x^2-1\right)^k, \quad \text{for } k = 0, 1, 2, \ldots. \tag{1.41}$$

Their orthogonality is exactly specified by

$$\int_{-1}^{1} L_k(x)L_m(x)\mathrm{d}x = \begin{cases} \dfrac{2}{2k+1} & \text{for } k = m, \\ 0 & \text{otherwise.} \end{cases} \tag{1.42}$$

Each consequent triad of Legendre polynomials obeys the relation

$$L_n(x) = \left(\frac{\mathrm{d}}{\mathrm{d}x}L_{n+1}(x) - \frac{\mathrm{d}}{\mathrm{d}x}L_{n-1}(x)\right), \quad n \geq 1, \tag{1.43}$$

and all of them satisfy

$$L_n(1) = 1, \; L_n(-1) = (-1)^n, \quad n \geq 0. \tag{1.44}$$

The Legendre expansion of a function $u \in L^2(I)$ has the form

$$u(x) = \sum_{n=0}^{\infty} c_n L_n(x), \tag{1.45}$$

which is understood as

$$\lim_{k\to\infty} \|u - \sum_{n=0}^{k} c_n L_n(x)\|_{L^2(I)} = 0. \tag{1.46}$$

The coefficients c_n are computed using the relation

$$c_n = \frac{2n+1}{2} \int_{-1}^{1} u(x) L_n(x). \tag{1.47}$$

It is not difficult to obtain explicit formulae for Legendre and also other sets of orthogonal polynomials up to very high orders using standard mathematical software. Let us list a few Legendre polynomials as a reference for computer implementation.

$$\begin{aligned}
L_0(x) &= 1, \\
L_1(x) &= x, \\
L_2(x) &= \frac{3}{2}x^2 - \frac{1}{2}, \\
L_3(x) &= \frac{1}{2}x(5x^2 - 3), \\
L_4(x) &= \frac{1}{8}(35x^4 - 30x^2 + 3), \\
L_5(x) &= \frac{1}{8}x(63x^4 - 70x^2 + 15), \\
L_6(x) &= \frac{1}{16}(231x^6 - 315x^4 + 105x^2 - 5), \\
L_7(x) &= \frac{1}{16}x(429x^6 - 693x^4 + 315x^2 - 35), \\
L_8(x) &= \frac{1}{128}(6435x^8 - 12012x^6 + 6930x^4 - 1260x^2 + 35), \\
L_9(x) &= \frac{1}{128}x(12155x^8 - 25740x^6 + 18018x^4 - 4620x^2 + 315), \\
L_{10}(x) &= \frac{1}{256}(46189x^{10} - 109395x^8 + 90090x^6 - 30030x^4 + 3465x^2 - 63).
\end{aligned} \tag{1.48}$$

The functions $L_0, L_1, \ldots, L_9$ are illustrated in Figures 1.9 – 1.13.

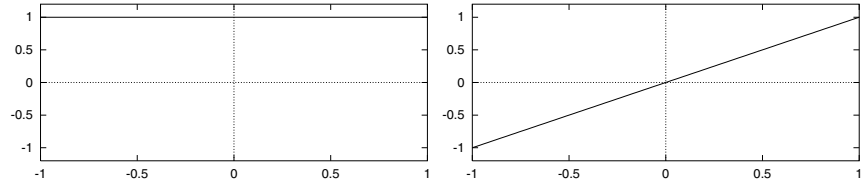

FIGURE 1.9: Legendre polynomials L_0, L_1.

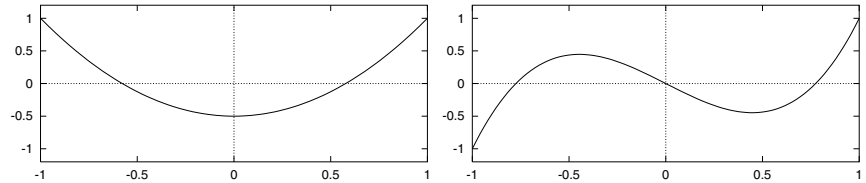

FIGURE 1.10: Legendre polynomials L_2, L_3.

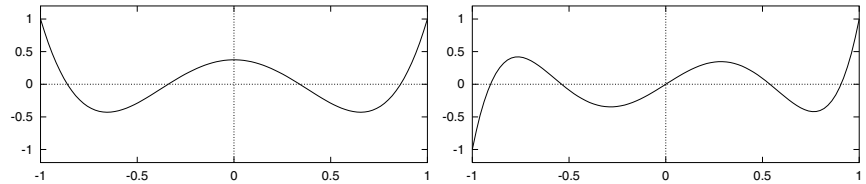

FIGURE 1.11: Legendre polynomials L_4, L_5.

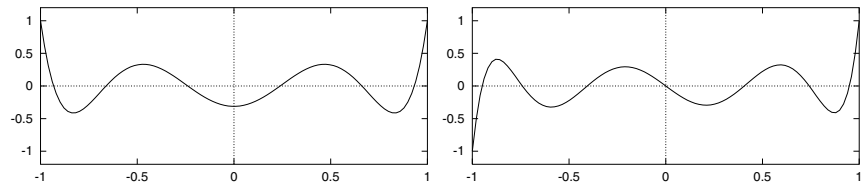

FIGURE 1.12: Legendre polynomials L_6, L_7.

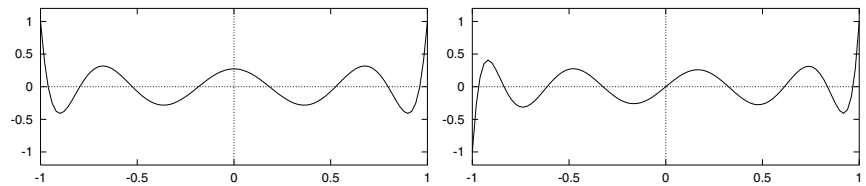

FIGURE 1.13: Legendre polynomials L_8, L_9.

1.2.3 Lobatto shape functions

Let us define functions

$$l_0(x) = \frac{1-x}{2}, \quad l_1(x) = \frac{x+1}{2}, \tag{1.49}$$
$$l_k(x) = \frac{1}{\|L_{k-1}\|_2} \int_{-1}^{x} L_{k-1}(\xi)\, \mathrm{d}\xi, \quad 2 \le k,$$

where $\|L_{k-1}\|_2 = \sqrt{2/(2k-1)}$ from (1.42). Obviously $l_k(-1) = 0$, $k = 2, 3, \ldots$. It follows from the orthogonality of higher-order Legendre polynomials L_k to $L_0 \equiv 1$,

$$\int_{-1}^{1} L_k(x)\, \mathrm{d}x = 0, \quad k \ge 1, \tag{1.50}$$

that also $l_k(1) = 0$, $k = 2, 3, \ldots$. The *Lobatto shape functions* $l_0, l_1, l_2, \ldots, l_p$ form a complete basis of the space $P_p(-1,1)$ of polynomials of the order of at most p in the interval $(-1,1)$. Let us list some of them for reference:

$$l_2(x) = \frac{1}{2}\sqrt{\frac{3}{2}}(x^2-1), \tag{1.51}$$
$$l_3(x) = \frac{1}{2}\sqrt{\frac{5}{2}}(x^2-1)x,$$
$$l_4(x) = \frac{1}{8}\sqrt{\frac{7}{2}}(x^2-1)(5x^2-1),$$
$$l_5(x) = \frac{1}{8}\sqrt{\frac{9}{2}}(x^2-1)(7x^2-3)x,$$
$$l_6(x) = \frac{1}{16}\sqrt{\frac{11}{2}}(x^2-1)(21x^4-14x^2+1),$$
$$l_7(x) = \frac{1}{16}\sqrt{\frac{13}{2}}(x^2-1)(33x^4-30x^2+5)x,$$
$$l_8(x) = \frac{1}{128}\sqrt{\frac{15}{2}}(x^2-1)(429x^6-495x^4+135x^2-5),$$
$$l_9(x) = \frac{1}{128}\sqrt{\frac{17}{2}}(x^2-1)(715x^6-1001x^4+385x^2-35)x,$$
$$l_{10}(x) = \frac{1}{256}\sqrt{\frac{19}{2}}(x^2-1)(2431x^8-4004x^6+2002x^4-308x^2+7).$$

The Lobatto shape functions will play an essential role in the design of hierarchic shape functions in Chapter 2. Some of them are illustrated in Figures 1.14 – 1.18 (notice the different scales).

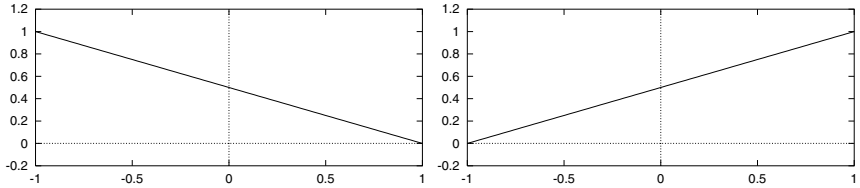

FIGURE 1.14: Lobatto shape functions l_0, l_1.

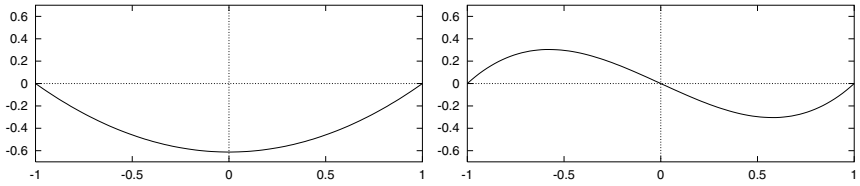

FIGURE 1.15: Lobatto shape functions l_2, l_3.

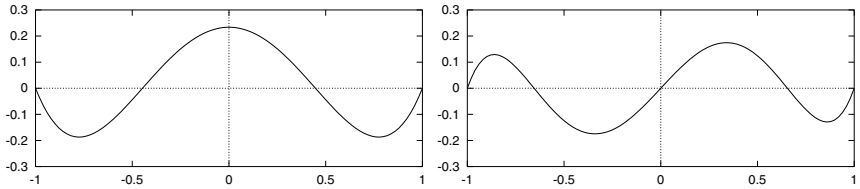

FIGURE 1.16: Lobatto shape functions l_4, l_5.

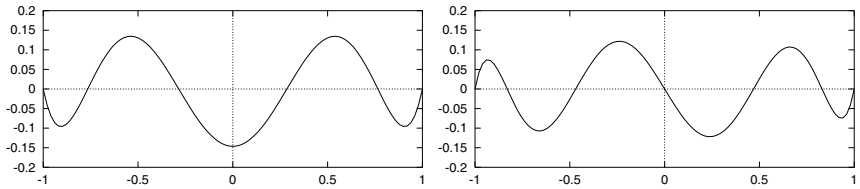

FIGURE 1.17: Lobatto shape functions l_6, l_7.

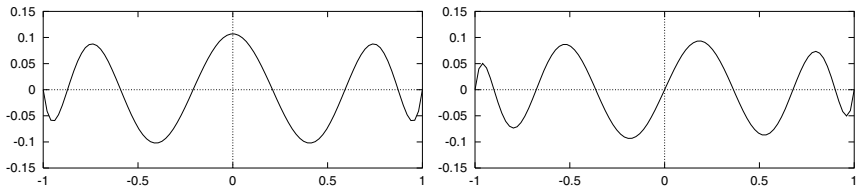

FIGURE 1.18: Lobatto shape functions l_8, l_9.

1.2.4 Kernel functions

For future use it is convenient to decompose the higher-order Lobatto shape functions $l_2, l_3, \ldots$ into products of the form

$$l_k(x) = l_0(x) l_1(x) \phi_{k-2}(x), \quad 2 \le k. \tag{1.52}$$

Since all functions l_k, $2 \le k$ vanish at ± 1, the *kernel functions* ϕ_{k-2}, $k = 2, 3, \ldots$ are *polynomials* of the order $k-2$. Let us list the following,

$$\begin{aligned}
\phi_0(x) &= -2\sqrt{\frac{3}{2}}, \qquad (1.53)\\
\phi_1(x) &= -2\sqrt{\frac{5}{2}}x,\\
\phi_2(x) &= -\frac{1}{2}\sqrt{\frac{7}{2}}(5x^2-1),\\
\phi_3(x) &= -\frac{1}{2}\sqrt{\frac{9}{2}}(7x^2-3)x,\\
\phi_4(x) &= -\frac{1}{4}\sqrt{\frac{11}{2}}(21x^4-14x^2+1),\\
\phi_5(x) &= -\frac{1}{4}\sqrt{\frac{13}{2}}(33x^4-30x^2+5)x,\\
\phi_6(x) &= -\frac{1}{32}\sqrt{\frac{15}{2}}(429x^6-495x^4+135x^2-5),\\
\phi_7(x) &= -\frac{1}{32}\sqrt{\frac{17}{2}}(715x^6-1001x^4+385x^2-35)x,\\
\phi_8(x) &= -\frac{1}{64}\sqrt{\frac{19}{2}}(2431x^8-4004x^6+2002x^4-308x^2+7),\\
&\vdots
\end{aligned}$$

The kernel functions $\phi_0, \phi_1, \ldots$ will be used for the definition of higher-order hierarchic shape functions on triangular, tetrahedral and prismatic elements in Chapter 2.

1.2.5 Horner's algorithm for higher-order polynomials

An attempt to implement the formulae (1.48), (1.51) and (1.53) in the same form as they are written on paper could cause significant roundoff errors for higher polynomial orders. Here is a simple explanation of what would happen.

Recall that a floating point operand is stored in the form of a *mantisa* and *exponent*. As long as the number of decimal digits does not exceed the length of mantisa, no information is lost. Therefore we can have both extremely large

and extremely small numbers stored exactly. However, problems arise when summation is involved, as operands are set to have the same exponent, and digits stored in the mantisa are shifted (to the right if exponent is increased, or to the left if it is decreased). In either case digits get lost when the difference between the exponents is large.

For the evaluation of higher-order polynomials this means that we should avoid direct summation of contribution of the form ax^n, bx^m where the powers m and n significantly differ. One of the standard ways to avoid these problems (sometimes called *Horner's algorithm*) is to transform the polynomials into the form

$$\sum_{n=0}^{M} a_n x^n = a_0 + x(a_1 + x(a_2 + x(a_3 + \ldots))). \tag{1.54}$$

A simple concrete example would be

$$a_0 + a_1 x^2 + a_2 x^4 + a_3 x^6 = a_0 + x^2(a_1 + x^2(a_2 + a_3 x^2)). \tag{1.55}$$

Moreover, computer summation takes less time than computer multiplication and is therefore a more efficient means to evaluate higher-order polynomials.

1.3 A one-dimensional example

The technical simplicity of the one-dimensional situation allows us to present the higher-order discretization procedure in detail. Let us begin with the formulation of a simple model problem.

1.3.1 Continuous and discrete problem

Consider an interval $I = (a, b) \subset \mathbb{R}$ and a load function $f \in L^2(I)$. We will solve the Poisson equation

$$-u''(x) = f(x) \tag{1.56}$$

in I, equipped with nonhomogeneous Dirichlet boundary conditions

$$\begin{aligned} u(a) &= g_a, \\ u(b) &= g_b. \end{aligned} \tag{1.57}$$

We proceed according to Paragraph 1.1.6. First notice that functions satisfying conditions (1.57) cannot constitute a vector space. This would be the case with zero Dirichlet boundary conditions; however, now the sum of two

functions satisfying (1.57) does not satisfy it anymore. Therefore we have to decompose the sought function u into

$$u(x) = u^*(x) + \bar{u}(x), \tag{1.58}$$

where the *Dirichlet lift* $u^* \in H^1(a,b)$ satisfies the boundary conditions (1.57),

$$\begin{aligned} u^*(a) &= g_a, \\ u^*(b) &= g_b, \end{aligned} \tag{1.59}$$

and the function $\bar{u}$, satisfying homogeneous Dirichlet boundary conditions

$$\bar{u}(a) = \bar{u}(b) = 0, \tag{1.60}$$

is the unknown part of the solution u. The function $\bar{u}$ already can be sought in a linear function space, namely

$$V = H_0^1(a,b). \tag{1.61}$$

Hence, the task is to find a function $\bar{u} \in V$ satisfying the variational formulation

$$\int_a^b [u^*(x) + \bar{u}(x)]' v'(x) \mathrm{d}\boldsymbol{x} = \int_a^b f(x) v(x) \mathrm{d}\boldsymbol{x} \quad \text{for all } v \in V. \tag{1.62}$$

This is the same as

$$\int_a^b (u^*)'(x) v'(x) \mathrm{d}\boldsymbol{x} + \int_a^b (\bar{u})'(x) v'(x) \mathrm{d}\boldsymbol{x} = \int_a^b f(x) v(x) \mathrm{d}\boldsymbol{x} \quad \text{for all } v \in V \tag{1.63}$$

and as

$$\int_a^b (\bar{u})'(x) v'(x) \mathrm{d}\boldsymbol{x} = \int_a^b f(x) v(x) - (u^*)'(x) v'(x) \mathrm{d}\boldsymbol{x} \quad \text{for all } v \in V. \tag{1.64}$$

Discretization of (1.64)

In the next step we specify a finite element mesh $\mathcal{T}_{h,p} = \{K_1, K_2, \ldots, K_M\}$ of elements with arbitrary polynomial orders $1 \leq p_1, p_2, \ldots, p_M$. We choose a reference domain $K_a = (-1,1)$ and for each element $K_i = (x_i, x_{i+1})$, $i = 1, 2, \ldots, M$ we define an affine reference map $x_{K_i} : K_a \to K_i$,

$$\begin{aligned} x_{K_i}(\xi) &= c_1^{(i)} + c_2^{(i)} \xi, \\ x_{K_i}(-1) &= x_i, \\ x_{K_i}(1) &= x_{i+1}. \end{aligned} \tag{1.65}$$

Hence it is

$$c_1^{(i)} = \frac{x_i + x_{i+1}}{2}, \quad c_2^{(i)} = J_{K_i} = \frac{x_{i+1} - x_i}{2}. \tag{1.66}$$

The space V is approximated by a subspace

$$V_{h,p} = \{v \in V;\ v|_{K_i} \circ x_{K_i} \in P^{p_i}(K_a) \text{ for all } i = 1, 2, \ldots, M\}, \tag{1.67}$$

where $(f \circ g)(x) \equiv f(g(x))$, of the dimension

$$N = \dim(V_{h,p}) = \underbrace{M - 1}_{\text{first-order part}} + \underbrace{\sum_{i=1}^{M}(p_i - 1)}_{\text{higher-order part}} = -1 + \sum_{i=1}^{M} p_i. \tag{1.68}$$

Hence, the approximate variational formulation (discrete problem) is

$$\int_a^b (\bar{u}_{h,p})'(x) v'_{h,p}(x) \mathrm{d}\boldsymbol{x} = \int_a^b f(x) v_{h,p}(x) - (u^*_{h,p})'(x) v'_{h,p}(x) \mathrm{d}\boldsymbol{x} \tag{1.69}$$

for all $v_{h,p} \in V_{h,p}$.

Choice of the lift function u^*

It is natural to require that the solution $u_{h,p} = u^*_{h,p} + \bar{u}_{h,p}$ is a polynomial of the order p_i on each element K_i, $i = 1, 2, \ldots, M$. Since $\bar{u}_{h,p} \in V_{h,p}$, this only can be achieved if the lift function $u^*_{h,p}$ is a polynomial of the order p_i on each element K_i as well. In practice one usually selects $u^*_{h,p}$ to be as simple as possible, i.e., as a continuous piecewise-linear function that vanishes in all interior elements (see Figure 1.19).

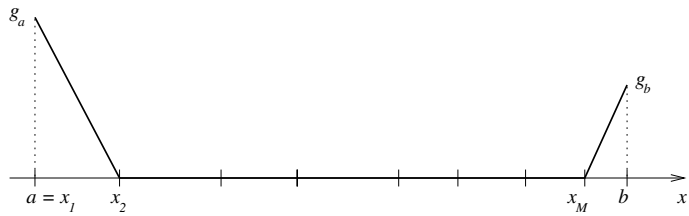

FIGURE 1.19: Example of a Dirichlet lift function for 1D problems.

In the following we will describe an algorithm that turns (1.69) into a system of linear algebraic equations.

1.3.2 Transformation to reference domain

The first step toward an efficient element-by-element assembly of the discrete problem is to transform the approximate variational formulation (1.69) elementwise from the mesh $\mathcal{T}_{h,p}$ to the reference domain K_a.

Transformation of function values

By $\bar{u}_{h,p}^{(i)}(\xi)$ we denote the approximate solution $\bar{u}_{h,p}$, transformed from the element K_i to the reference domain K_a, i.e.,

$$\bar{u}_{h,p}^{(i)}(\xi) \equiv (\bar{u}_{h,p} \circ x_{K_i})(\xi) = \bar{u}_{h,p}(x)|_{x=x_{K_i}(\xi)}. \tag{1.70}$$

In other words, the function value of $\bar{u}_{h,p}^{(i)}$ at a reference point $\xi \in K_a$ has to be the same as the function value of $\bar{u}_{h,p}$ at its image $x_{K_i}(\xi) \in K_i$.

Transformation of derivatives

One has to be a little more careful when transforming derivatives. The chain rule yields

$$[\bar{u}_{h,p}^{(i)}(\xi)]' = (u_{h,p} \circ x_{K_i})'(\xi) = u'_{h,p}(x)|_{x=x_{K_i}(\xi)} J_{K_i}. \tag{1.71}$$

This means that

$$[\bar{u}_{h,p}]'(x) = \frac{1}{J_{K_i}}[\bar{u}_{h,p}^{(i)}]'(\xi), \tag{1.72}$$

i.e., the derivative of $\bar{u}_{h,p}$ at the physical point $x = x_{K_i}(\xi) \in K_i$ is expressed as the derivative of the new function $\bar{u}_{h,p}^{(i)}$ at the corresponding reference point $\xi \in K_a$ divided by the Jacobian J_{K_i} (which obviously for the affine map x_{K_i} is constant).

Transformation of integrals from the variational formulation

The test functions $v_{h,p}$ and their derivatives are transformed analogously. Using the Substitution Theorem (that produces a factor J_{K_i} behind the integral sign), it is easy to conclude that

$$\int_{K_i} [\bar{u}_{h,p}]'(x) v'_{h,p}(x) \mathrm{d}\boldsymbol{x} = \int_{K_a} \frac{1}{J_{K_i}} [\bar{u}_{h,p}^{(i)}]'(\xi) [\bar{v}_{h,p}^{(i)}]'(\xi) \, \mathrm{d}\xi \quad \text{for all } i = 1, 2, \ldots, M. \tag{1.73}$$

The right-hand side is transformed analogously,

$$\begin{aligned} &\int_{K_i} f(x) v_{h,p}(x) - (u^*_{h,p})'(x) v'_{h,p}(x) \mathrm{d}\boldsymbol{x} \\ &= \int_{K_a} J_{K_i} f^{(i)}(\xi) v_{h,p}^{(i)}(\xi) - \frac{1}{J_{K_i}} [u_{h,p}^{*(i)}]'(\xi) [v_{h,p}^{(i)}]'(\xi) \, \mathrm{d}\xi, \end{aligned} \tag{1.74}$$

where $f^{(i)}(\xi) = (f \circ x_{K_i})(\xi)$ and so on.

1.3.3 Higher-order shape functions

Basic ideas of the design of *nodal* and *hierarchic* elements were revisited in Paragraph 1.1.2. The one-dimensional model problem offers a good chance to look in more detail at both the nodal and hierarchic shape functions and their conditioning properties.

Nodal higher-order shape functions

Consider an element K_i of the polynomial order p_i. For simplicity we distribute the nodal points equidistantly, i.e., we define

$$X_{j+1} = -1 + \frac{2j}{p_i} \in K_a, \quad j = 0, 1, \ldots, p_i. \tag{1.75}$$

Exploiting the Lagrange interpolation polynomial and the δ-property (1.6), we obtain the $p_i + 1$ nodal functions of the order p_i of the form

$$\theta_i(\xi) = \frac{\displaystyle\prod_{1 \le j \le p_i+1, j \ne i} (\xi - X_j)}{\displaystyle\prod_{1 \le j \le p_i+1, j \ne i} (X_i - X_j)}. \tag{1.76}$$

For piecewise linear approximations ($p_i = 1$) there are two points $X_1 = -1$, $X_2 = 1$, and the two linear (affine) shape functions have the form

$$\theta_1(\xi) = \frac{1 - \xi}{2}, \ \theta_2(\xi) = \frac{\xi + 1}{2}. \tag{1.77}$$

Complete sets of nodal shape functions for $p_i = 2$ and $p_i = 3$ are illustrated in Figures 1.20 and 1.21.

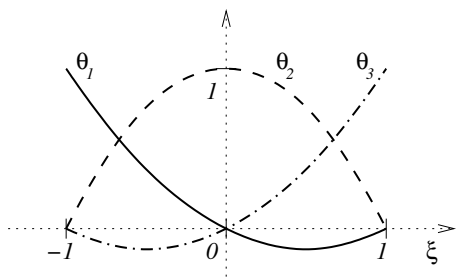

FIGURE 1.20: Quadratic nodal shape functions.

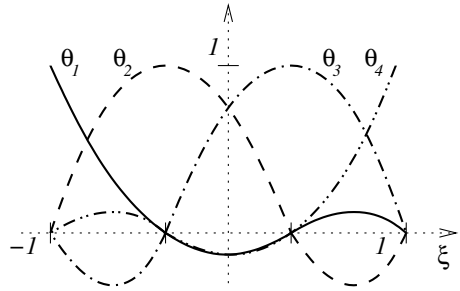

FIGURE 1.21: Cubic nodal shape functions.

These shape functions are advantageous from the point of view that the corresponding unknown coefficients, obtained from the solution of the discrete problem, directly represent the value of the approximate solution $u_{h,p}$ at the geometrical nodes X_i. On the other hand they are not hierarchic and thus one has to replace the whole set of shape functions when increasing the polynomial order of elements. Further, in higher spatial dimensions it is difficult to combine nodal elements with various polynomial orders in the mesh and therefore they are not suitable for *p*- and *hp*-adaptivity. With simple choices of nodal points these shape functions yield ill-conditioned stiffness matrices.

Hierarchic higher-order shape functions

As mentioned in Paragraph 1.1.2, by hierarchic we mean that the basis $\mathcal{B}^{p+1}$ of the polynomial space $P_{p+1}(K_a)$ is obtained from the basis $\mathcal{B}^p$ of the polynomial space $P_p(K_a)$ by adding new shape functions only. Particularly in 1D we add always a single $(p+1)$th-order shape function to the previous basis only. This is essential for *p*- and *hp*-adaptive finite element codes since one does not have to change his shape functions completely when increasing the order of polynomial approximation. Among hierarchic shape functions, one of the most popular choices is *Lobatto shape functions* (integrated Legendre polynomials) $l_0, l_1, \ldots$ that we introduced in Section 1.2. Their excellent conditioning properties are rooted in the fact that their derivatives are (normalized) Legendre polynomials, and thus that their H^1_0-product satisfies

$$\int_{-1}^{1} l'_{i-1}(\xi) l'_{j-1}(\xi)\,\mathrm{d}\xi = 0 \quad \text{whenever } i > 2 \text{ or } j > 2, \quad i \neq j. \tag{1.78}$$

DEFINITION 1.9 (Master element stiffness matrix) *Let θ_1, θ_2, ..., θ_{p+1} be a basis in the polynomial space $P^p(K_a)$. Then the master element stiffness matrix of order p corresponding to the problem (1.56), i.e., to the Laplace operator in 1D, is a $(p+1) \times (p+1)$ matrix of the form*

$$\hat{\boldsymbol{S}} = \{\hat{s}_{ij}\}_{i,j=1}^{p+1}, \ \hat{s}_{ij} = \int_{K_a} \theta_i'(\xi)\theta_j'(\xi)\, d\xi. \tag{1.79}$$

REMARK 1.10 (Role of master element stiffness matrix) It was shown in Paragraph 1.3.2 that stiffness integrals from the variational formulation, transformed from an element $K_i \in \mathcal{T}_{h,p}$ to the reference interval K_a, keep their original form up to the multiplication by an element-dependent constant (which was the inverse Jacobian of the affine map x_{K_i}). Thus, all integrals, which are needed for the assembly of the global stiffness matrix, are available in the master element stiffness matrix. This essentially reduces the cost of the computation.

Unfortunately the master element stiffness matrix cannot be exploited in this way when the operator in the variational formulation is explicitly space-dependent (consider, e.g., the operator $\tilde{L}(u) = ((1+x)u')'$ instead of the Laplace operator $L(u) = u''$).

Master element stiffness matrices can also be utilized in the same way in 2D and 3D when the reference maps are affine. Stiffness contributions of elements which are equipped with other than affine reference maps must be physically integrated on every mesh element. ∎

The relation (1.78) implies that the master element stiffness matrix $\hat{\boldsymbol{S}}$ of the order p for the Lobatto shape functions $l_0, l_1, \ldots, l_p$ looks like

$$\hat{\boldsymbol{S}} = \begin{pmatrix} \hat{s}_{11} & \hat{s}_{12} & 0 & 0 & \ldots & 0 \\ \hat{s}_{21} & \hat{s}_{22} & 0 & 0 & & \vdots \\ 0 & 0 & \hat{s}_{33} & 0 & & \\ 0 & 0 & 0 & \ddots & & \\ \vdots & & & & & 0 \\ 0 & \ldots & & & 0 & \hat{s}_{p+1,p+1} \end{pmatrix}. \tag{1.80}$$

The only nonzero nondiagonal entries correspond to products of the first-order shape functions l_0, l_1. This sparse structure is what makes the Lobatto shape functions so popular for the discretization of problems involving the Laplace operator.

DEFINITION 1.10 (Condition number) *Let $\boldsymbol{M}$ be a regular $n \times n$ matrix. The product*

$$\varkappa(\boldsymbol{M}) = \|\boldsymbol{M}\|\|\boldsymbol{M}^{-1}\|, \tag{1.81}$$

where $\|\,.\,\|$ is a matrix norm, is called the condition number *of the matrix $\boldsymbol{M}$ (relative to the norm $\|\,.\,\|$).*

REMARK 1.11 (Spectral condition number) The matrix norm $\|\,.\,\|$ in Definition 1.10 can be chosen in many different ways. Most commonly used are the *Euclidean (Frobenius) norm*

$$\|\boldsymbol{M}\| = \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{n} m_{ij}^2} \tag{1.82}$$

and the *spectral norm*

$$\|\boldsymbol{M}\| = \sqrt{\rho(\boldsymbol{M}\boldsymbol{M}^T)}, \tag{1.83}$$

where $\rho(\boldsymbol{M}\boldsymbol{M}^T)$ is the *spectral radius* (i.e., the largest eigenvalue) of the symmetric positive definite matrix $\boldsymbol{M}\boldsymbol{M}^T$.

Using the spectral norm (1.83), one arrives at the frequently used *spectral (Todd) condition number*

$$1 \leq \varkappa^*(\boldsymbol{M}) = \frac{\max_{\lambda\in\sigma(\boldsymbol{M})}|\lambda|}{\min_{\lambda\in\sigma(\boldsymbol{M})}|\lambda|}, \tag{1.84}$$

(see, e.g., [195]) where $\sigma(\boldsymbol{M})$ is the *spectrum* (set of all eigenvalues) of the matrix $\boldsymbol{M}$. It holds

$$\varkappa^*(\boldsymbol{M}) \leq \varkappa(\boldsymbol{M})$$

for any other matrix norm. It is a widely known fact that the performance of solvers for systems of linear algebraic equations improves in accordance with lower condition numbers of the solved matrices. □

Example 1.5 (Conditioning properties of various types of shape functions) Conditioning of the master element stiffness matrix is used as an orientation factor for the selection of optimal shape functions. Let us return to the Laplace operator in one spatial dimension. To illustrate the importance of a good choice of shape functions, in addition to the nodal shape functions (1.76) and Lobatto shape functions (1.49) consider a set of hierarchic shape functions of the simple form

$$\begin{aligned}
\theta_1(\xi) &= \frac{1-\xi}{2}, \\
\theta_2(\xi) &= \frac{1+\xi}{2}, \\
\theta_k(\xi) &= \frac{(1-\xi)^{k-1}(1+\xi)}{4}.
\end{aligned} \tag{1.85}$$

Figure 1.22 compares the conditioning of the corresponding master element stiffness matrices. More precisely, for $p = 2, 3, \ldots, 10$ we depict the spectral

condition number of the submatrix of the master element stiffness matrix, corresponding to the bubble functions only (the master element stiffness matrix itself corresponds to the solution of the original problem using one single element and no boundary conditions, and obviously it is singular).

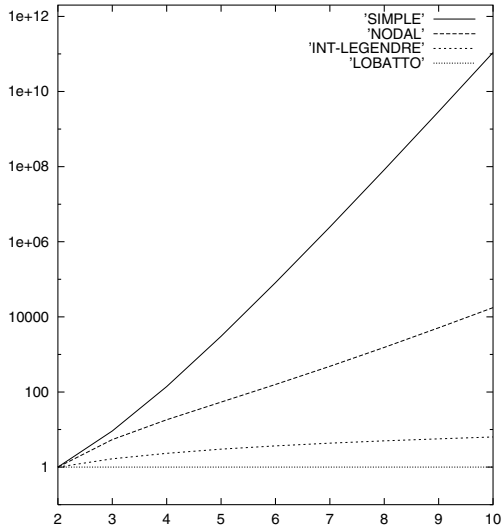

FIGURE 1.22: Conditioning of master element stiffness matrices for the simple shape functions (1.85), nodal shape functions (1.76), integrated Legendre polynomial (without normalization) and finally Lobatto shape functions (1.49), in this order. Notice that the scale is decimal logarithmic.

For the sake of completeness let us add that, unfortunately, conditioning properties of bubble functions get worse in higher spatial dimensions. The case $\varkappa \equiv 1$ is idealistic and only true for the Lobatto shape functions in 1D. □

REMARK 1.12 (Invariance of the condition number) The condition numbers of (bubble-submatrices of) the master element stiffness matrices, shown in Figure 1.22, are invariant with respect to permutation of indices of bubble functions. To see this, it is sufficient to consider a permutation that exchanges the indices k, l in a pair of bubble functions θ_k and θ_l only. In this case it easily follows from the definition that all eigenvalues of the new matrix will be the same, and the new eigenvectors will be obtained from the original ones by switching their kth and lth components. □

1.3.4 Design of basis functions

Both sets of the nodal and hierarchic higher-order shape functions that we have introduced in Paragraph 1.3.3 consist of *vertex* and *bubble* shape functions. The vertex shape functions are nonzero at one of the endpoints of the reference interval $K_a = (-1, 1)$ and vanish at the other (in the hierarchic case these are l_0 and l_1, in the nodal case θ_1 and θ_{p+1}). The rest are bubble shape functions that vanish at both endpoints. Accordingly, the basis functions of the space $V_{h,p}$ will be split into vertex and bubble functions.

Vertex basis functions

The vertex functions are related to grid points and their support consists of two neighboring elements. Consider a grid point x_{i+1} and the adjacent elements K_i, K_{i+1}. In the hierarchic case the corresponding vertex basis function v_i is defined as

$$v_i(x) = \begin{cases} (l_1 \circ x_{K_i}^{-1})(x), & x \in K_i, \\ (l_0 \circ x_{K_{i+1}}^{-1})(x), & x \in K_{i+1}. \end{cases} \tag{1.86}$$

These functions are sometimes called "hat functions" (see Figure 1.23).

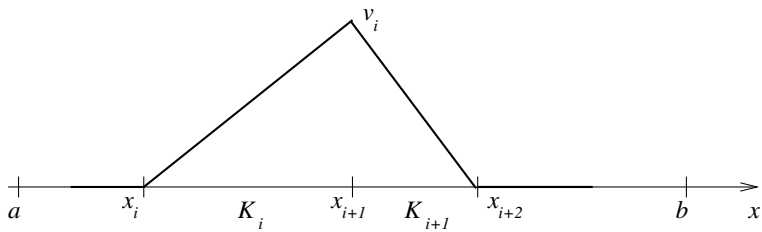

FIGURE 1.23: Vertex basis functions v_i of the space $V_{h,p}$ in the hierarchic case.

In the context of nodal elements the vertex basis functions are defined as

$$v_i(x) = \begin{cases} (\theta_{p+1} \circ x_{K_i}^{-1})(x), & x \in K_i, \\ (\theta_1 \circ x_{K_{i+1}}^{-1})(x), & x \in K_{i+1}. \end{cases} \tag{1.87}$$

Here p is the polynomial order associated with both the elements K_i and K_{i+1}. An example of a quadratic nodal vertex basis function is depicted in Figure 1.24.

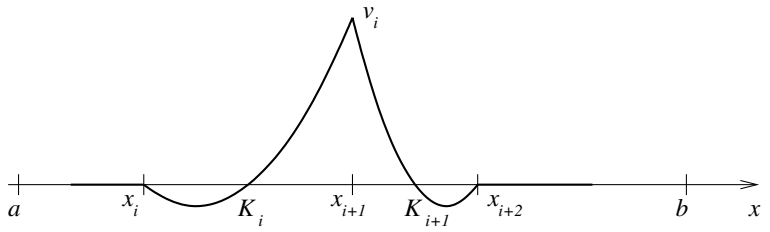

FIGURE 1.24: Vertex nodal basis functions v_i of the space $V_{h,p}$ for piecewise-quadratic approximations.

Bubble basis functions

Bubble basis functions for an element K_i of pth order are defined analogously as a composition of bubble shape functions and the inverse map $x_{K_i}^{-1}$. In the hierarchic case one uses the bubble shape functions $l_2, l_3, \ldots, l_p$, and in the nodal case the bubble shape functions $\theta_2, \theta_3, \ldots, \theta_p$. Examples of hierarchic quadratic and cubic bubble functions are shown in Figures 1.25 and 1.26.

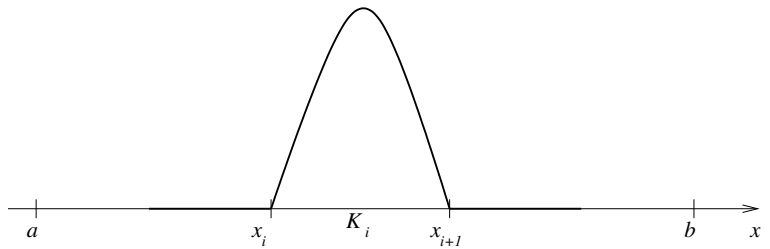

FIGURE 1.25: An example of a hierarchic quadratic bubble basis function.

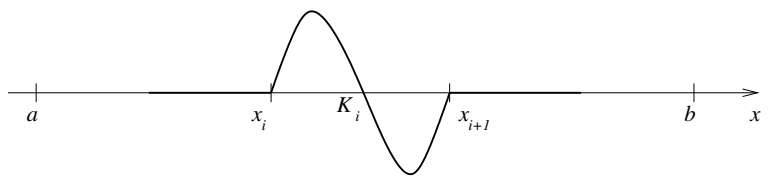

FIGURE 1.26: An example of a hierarchic cubic bubble basis function.

1.3.5 Sparsity structure and connectivity

It is widely known that when indexing the grid points in the interval $I = (a, b)$ consecutively from the left to the right, the global stiffness matrix for piecewise-linear approximations is tridiagonal, analogously to finite difference schemes. This analogy between first-order finite elements and finite differences extends to 2D and 3D when using Cartesian grids. However, higher-order finite elements in 1D and already first-order FE discretizations on unstructured meshes in 2D and 3D yield general sparse structures.

Enumeration of basis functions of $V_{h,p}$

The sparsity structure of the stiffness matrix is uniquely determined by the ordering of the basis functions of the space $V_{h,p}$. Generally one is free to index the basis functions in any way she/he wishes; however, from the point of view of compatibility between first- and higher-order approximations it is reasonable to put the vertex functions according to consecutively denumerated grid points first. After that, all higher-order basis functions can be denumerated consecutively within elements according to their polynomial order, one element at a time.

Connectivity information

The connectivity information is a data construction which for each element K_i links the master element shape functions $l_0, l_1, l_2, \ldots$ to the basis functions $v_1, v_2, \ldots, v_N \subset V_{h,p}$ in the finite element mesh (let us work with Lobatto shape functions for instance). These data are stored elementwise and their actual amount depends on the needs (and sophistication) of the assembling algorithm. The simplest option, which surely is not the most efficient one possible, is to store for each mesh element K_i an integer index array $\boldsymbol{c}_i$ of the length $p_i + 1$, where p_i is the order of approximation on K_i. This array is filled with the indices of basis functions of $V_{h,p}$, which are related to shape functions $l_0, l_1, \ldots, l_{p_i}$ on K_i:

$$\begin{aligned} l_0 &\ \ldots\ c_{i,1} \\ l_1 &\ \ldots\ c_{i,2} \\ l_2 &\ \ldots\ c_{i,3} \\ &\ \vdots \\ l_{p_i} &\ \ldots\ c_{i,p_i+1} \end{aligned} \tag{1.88}$$

We may use here, e.g., -1 instead of an index in order to indicate that a shape function is not related to any basis function because of a Dirichlet boundary condition.

Example 1.6 (Connectivity information)
Consider a mesh $\mathcal{T}_{h,p}$ consisting of three elements K_1, K_2 and K_3 of polynomial orders $p_1 = 3$, $p_2 = 4$ and $p_3 = 2$. Let us look at connectivity data for a discretization with zero Dirichlet boundary conditions on both interval endpoints.

In this case the dimension of the space $V_{h,p}$ is $N = 8$ and the element connectivity arrays look like the following:

$$K_1 \ldots \boldsymbol{c}_1 = \{-1, 1, 3, 4\},$$
$$K_2 \ldots \boldsymbol{c}_2 = \{1, 2, 5, 6, 7\},$$
$$K_3 \ldots \boldsymbol{c}_3 = \{2, -1, 8\}.$$

The components of these index arrays are related to the shape functions $l_0, l_1, l_2, \ldots$ in this order. In this case $c_{1,1} = c_{3,2} = -1$ means that the shape function l_0 on element K_1 and shape function l_1 on element K_3 are not used due to the Dirichlet boundary conditions. ∎

Let us show a general algorithm that defines connectivity information in 1D for various types of boundary conditions at the interval endpoints $x_1 = a$ and $x_{M+1} = b$, and for an arbitrary distribution of the polynomial orders p_i, $i = 1, 2, \ldots, M$.

ALGORITHM 1.1 (Preparing connectivity data in 1D)

1. counter $:= 1$
2. First-order basis functions of the element K_1*:*
if(Dirichlet boundary condition at a) then $c_{1,1} := -1$
else $c_{1,1} :=$ *counter*, *counter* $:=$ *counter* $+ 1$
Shape function l_1 *for* K_1 *corresponds to* $v_{counter}$*:*
$c_{1,2} :=$ *counter*
3. Loop over elements $K_2, K_3, \ldots, K_{M-1}$ *indexing hat functions:*
for($k = 2$ *to* $M - 1$*) do* {
  (global index for l_0*:)* $c_{k,1} :=$ *counter*, *counter* $:=$ *counter* $+ 1$
  (global index for l_1*:)* $c_{k,2} :=$ *counter*
}
4. First-order basis functions of the element K_M*:*
Shape function l_0 *for* K_M *corresponds to* $v_{counter}$*:*
$c_{M,1} :=$ *counter*, *counter* $:=$ *counter* $+ 1$
if(Dirichlet boundary condition at b) then $c_{M,2} := -1$
else $c_{M,2} :=$ *counter*, *counter* $:=$ *counter* $+ 1$
5. Loop over all elements indexing higher-order basis functions:
for($k = 1$ *to* M*) do* {

for$(p = 2$ *to* $p_k)$ *do* {
(global index for l_p*:)* $c_{k,p+1} := counter$, $counter := counter + 1$
}
}

Treatment of connectivity information in higher spatial dimensions will be discussed in detail in Chapter 3.

1.3.6 Assembling algorithm

In higher spatial dimensions we need to store a flag for each element specifying whether it lies on the boundary or not, and a link to the appropriate boundary data. This is not necessary in 1D since we know that if -1 is the first entry in the connectivity array, we are on element K_1, and we are on element K_M if -1 is its second entry. The algorithm consists basically of *one single loop* over all mesh elements $K_1, K_2, \ldots, K_M$. This is an essential difference with respect to first-order discretizations where it is sufficient to loop over grid vertices – the first-order approach indeed does not generalize to higher-order schemes.

ALGORITHM 1.2 (Assembling algorithm)

Evaluate the master element stiffness matrix $\hat{\boldsymbol{S}}$ *corresponding to the highest polynomial order in the mesh. If this is not possible, for example because the operator is explicitly space-dependent, one will have to integrate stiffness terms analogous to (1.79) on each mesh element.*
Store the Jacobian of the reference map x_{K_i} *for each element in the mesh.*
Set the matrix $\boldsymbol{S} = \{s_{ij}\}_{i,j=1}^{N}$ *zero.*
Set the right-hand side vector $\boldsymbol{F} = (F_1, F_2, \ldots, F_N)^T$ *zero.*

*(*element loop*:)* **for** $k = 1, 2, \ldots, M$ **do** {
*(*first loop over shape (test) functions*:)* **for** $i = 1, 2, \ldots, p_k + 1$ **do** {
*(*second loop over shape (basis) functions*:)* **for** $j = 1, 2, \ldots, p_k + 1$ **do** {
put $m_1 = c_{k,i}$ *(if not* -1*, this is the global index of a* test *function* $v_{m_1} \in V_{h,p}$*, i.e.,* row *in the global stiffness matrix)*
put $m_2 = c_{k,j}$ *(if not* -1*, this is the global index of a* basis *function* $v_{m_2} \in V_{h,p}$*, i.e.,* column *in the global stiffness matrix)*
if*(*$m_1 \neq -1$ **and** $m_2 \neq -1$*)* **then put** $s_{m_1,m_2} = s_{m_1,m_2} + \frac{1}{J_{K_k}} \hat{s}_{i,j}$
else { *(beginning of treatment of Dirichlet bdy. conditions)*
if*(*$m_1 \neq -1$ **and** $m_2 == -1$*)* **then** { *(the condition* $m_2 == -1$ *means that the shape functions* l_0, l_1 *can represent the Dirichlet lift* $u^*_{h,p}$*. By* $m_1 \neq -1$ *we do not allow them to represent test functions from the space*

$V_{h,p}$ – *recall that all functions from the space* $V_{h,p} \subset H_0^1(a,b)$ *must respect homogeneous Dirichlet bdy. conditions)*

if$(j == 1)$ **then** { *(we are on* K_1 *and use* $g_a l_0(\xi)$ *for the transformed Dirichlet lift)*

put $F_{m_1} = F_{m_1} - \int_{K_a} \frac{1}{J_{K_1}} g_a l_0'(\xi) l_{i-1}'(\xi)\, d\xi = F_{m_1} - \frac{1}{J_{K_1}} g_a \hat{s}_{1,i}$ *(extra contribution to the right-hand side)*

}

else { *(now* $j == 2$*, we are on* K_M *and use* $g_b l_1(\xi)$ *for the transformed Dirichlet lift)*

put $F_{m_1} = F_{m_1} - \int_{K_a} \frac{1}{J_{K_M}} g_b l_1'(\xi) l_{i-1}'(\xi)\, d\xi = F_{m_1} - \frac{1}{J_{K_1}} g_b \hat{s}_{2,i}$ *(extra contribution to the right-hand side)*

}

}

} *(end of treatment of Dirichlet bdy. conditions)*

} *(end of second loop over shape (basis) functions)*

if$(m_1 \neq -1)$ **then put** $F_{m_1} = F_{m_1} + \int_{K_a} J_{K_k} \tilde{f}^{(k)}(\xi) l_{i-1}(\xi)\, d\xi$ *(regular contribution to the right-hand side)*

} *(end of first loop over shape (test) functions)*

} *(end of element loop)*

(where again $\tilde{f}^{(k)}(\xi) = f(x_{K_k}(\xi))$*).*

The assembling algorithm for 2D and 3D approximations will be introduced in Chapter 3.

1.3.7 Compressed representation of sparse matrices

There is a well-established compressed format for the representation of sparse matrices (*Compressed Sparse Row (CSR)*: see, e.g., [72, 156, 169, 193]). Once one adapts to this format, she/he will be able to find many software packages that will precondition and solve the discrete problem. Let N be the rank of the matrix $\boldsymbol{S}$ (i.e., the number of unknown coefficients of the discrete problem) and by NNZ denote the number of nonzero entries in $\boldsymbol{S}$. Virtually all sparse matrix solvers require that the matrix $\boldsymbol{S}$ is written in the form of three arrays:

1. Array A of length NNZ: this is a real-valued array containing all nonzero entries of the matrix $\boldsymbol{S}$ listed from the left to the right, starting with the first and ending with the last row.

2. Array IA of length $N+1$: this is an integer array, $IA[1] = 1$. $IA[k+1] = IA[k] + nnz_k$ where nnz_k is the number of nonzero entries in the kth row.

3. Array JA of length NNZ: this is an integer array containing the row-positions of all entries from array A.