FINITE ELEMENT METHODS: IMPLEMENTATIONS

Vít DOLEJŠÍ

Charles University Prague
Faculty of Mathematics and Physics
Department of Numerical Mathematics
Sokolovská 83, 186 75 Prague
Czech Republic

Praha March 9, 2025

Contents

Chapter 1

Implementation of FEM and DGM

In this chapter we deal with the implementation of the finite element and the discontinuous Galerkin methods which were introduced in the previous chapters. However, the implementation of these methods is a rather complicated problem which can differ based on the practical applications. Our aim is only to introduce some basic ideas. This chapter is relatively autonomous, all necessary definitions and notations are recalled.

In order to render these notes more readable, we start in Section 1.1 with the implementation of a simple problem, namely P^1 continuous finite element solution of the Poisson problem with homogeneous Dirichlet boundary condition. We describe the implementation of this problem in details including the code subroutines which are written in Fortran 90 syntax. We suppose that the non-Fortran readers will be able to understand this syntax too since the subroutines contain only standard 'if then' conditions and 'do' cycles. The subroutines are commented in the standard way, i.e., text behind the character '!' is ignored by a Fortran 90 translator.

In Section 1.2, we describe a general implementation of the conforming finite element method. We employ the concept of the reference element which seems (in the authors' opinion) more suitable for the implementation. Moreover, we include a description of the necessary data structures. In Section 1.3, we present a construction of the basis functions based on two possible sets of shape functions, the Lagrangian and the Lobatto shape functions. Section 1.4 contains an implementation of the discontinuous Galerkin method with emphasis on the differences with the conforming methods. Since DG methods allow a simple treatment of *hp*-methods, we consider an approximation of different polynomial degrees on different elements. Finally, Sections 1.5 and 1.6 describe possible numerical quadratures and visualization techniques, respectively.

1.1 P^1 solution of the model problem

In order to explain the basic aspects of the implementation of FEM and DGM, we start with a model problem represented by the Poisson equation. We recall the weak formulation and the finite element formulation of this elliptic problem.

Let Ω be a bounded polygonal domain in $\mathbb{R}^d$, $d = 2, 3$, with a boundary $\partial\Omega$. We seek a function $u : \Omega \to \mathbb{R}$ such that

$$\begin{aligned} -\Delta u(x) &= g(x), \quad x \in \Omega, && (1.1) \\ u(x) &= 0, \quad x \in \partial\Omega, && (1.2) \end{aligned}$$

where $g \in L^2(\Omega)$.

Let $V := H_0^1(\Omega)$ then the *weak formulation* of (1.1) reads

$$\text{find } u \in V : \qquad (\nabla u, \nabla v) = (g, v) \quad \forall v \in V. \tag{1.3}$$

Let V_h be a finite dimensional subspace of V then the *finite element* approximation (1.3) can be written as

$$\text{find } u_h \in V_h : \qquad (\nabla u_h, \nabla v_h) = (g, v_h) \quad \forall v_h \in V_h. \tag{1.4}$$

In the following, we describe in details the implementation of the method (1.4) with the aid of a continuous piecewise linear approximation constructed over a triangular grid.

1.1.1 Triangulation

Let $\mathscr{T}_h$ $(h > 0)$ be a partition of the closure $\overline{\Omega}$ of the domain Ω into a finite number of closed d-dimensional simplexes T with mutually disjoint interiors such that

$$\overline{\Omega} = \bigcup_{T \in \mathscr{T}_h} T. \tag{1.5}$$

In two-dimensional problems ($d = 2$), the elements $T \in \mathscr{T}_h$ are triangles and in three-dimensional problems ($d = 3$) the elements $T \in \mathscr{T}_h$ are tetrahedra. Figure 1.1 shows two examples of a triangular grid constructed over a square domain.

Moreover, we use the following notation for vertices and edges (faces for $d = 3$) of $\mathscr{T}_h$. We denote by $\{\mathbf{v}_i\}_{i=1}^N$ the set of all vertices of $T \in \mathscr{T}_h$ lying inside of Ω and by $\{\mathbf{v}_i\}_{i=N+1}^{N+N_b}$ the set of all vertices of $T \in \mathscr{T}_h$ lying on $\partial\Omega$. Finally, $\{e_i\}_{i=1}^E$ denotes the set of all edges (faces) of all $T \in \mathscr{T}_h$ lying inside of Ω and $\{e_i\}_{i=E+1}^{E+E_b}$ the set of all edges (faces) of all $T \in \mathscr{T}_h$ lying on $\partial\Omega$.

1.1.2 Finite element space

Let

$$V_h := \{v_h,\ v_h \in C(\overline{\Omega}) \cap H_0^1(\Omega), v_h|_T \in P^1(T)\ \forall T \in \mathscr{T}_h\} \tag{1.6}$$

denote the space of continuous piecewise linear functions on $\mathscr{T}_h$ vanishing on $\partial\Omega$. Obviously, each $v_h \in V_h$ is uniquely defined by its values in $\{\mathbf{v}_i\}_{i=1}^N$.

We denote by $B := \{\varphi_i\}_{i=1}^N$ the set of basis functions of V_h defined by

$$\varphi_i \in V_h,\ \varphi_i(\mathbf{v}_j) = \delta_{ij}, \quad i, j = 1, \dots, N, \tag{1.7}$$

where δ_{ij} is the Kronecker symbol. Obviously, the support of each $\varphi_i,\ i = 1, \dots, N$ is small and consists of all triangles $T \in \mathscr{T}_h$ having $\mathbf{v}_i$ as a vertex, see Figure 1.2.

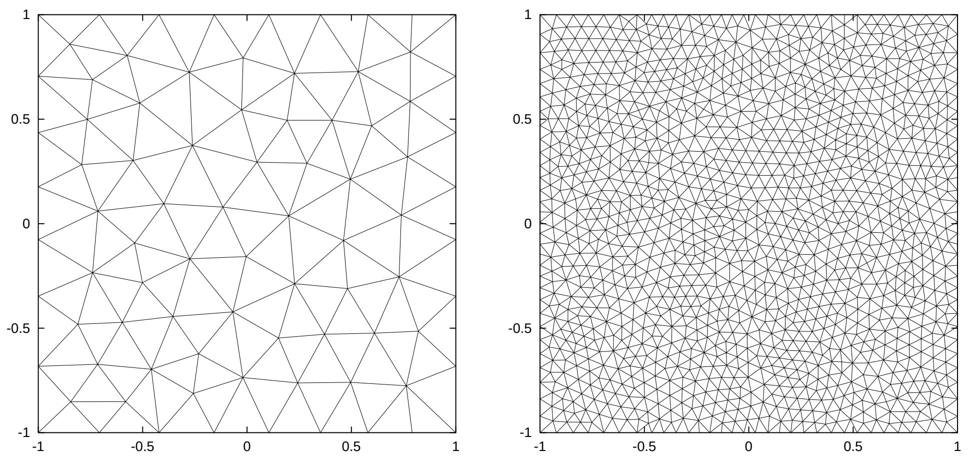


Figure 1.1: Examples of triangular meshes constructed over $\Omega = (-1, 1) \times (-1, 1)$: a coarser grid (left) and a finer one (right)

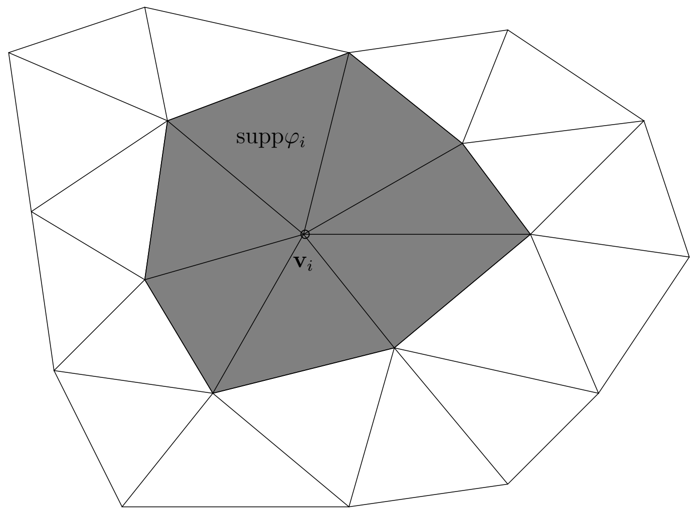


Figure 1.2: Support of the basis function φ_i corresponding to the node $\mathbf{v}_i$

1.1.3 Discretization

The approximate solution $u_h \in V_h$ can be expressed as

$$u_h(x) = \sum_{j=1}^{N} u_j \varphi_j(x), \qquad u_j \in \mathbb{R},\ j = 1, \dots, N. \tag{1.8}$$

Moreover, since $\varphi_i,\ i = 1, \dots, N$ form the basis of V_h, it is sufficient that the relation (1.4) is valid for each $\varphi_i,\ i = 1, \dots, N$. Then, using (1.8), the approximate problem (1.4) is equivalent to

$$\text{find } \{u_j\}_{j=1}^{N} \in \mathbb{R}^N \qquad \sum_{j=1}^{N} u_j (\nabla\varphi_j, \nabla\varphi_i) = (g, \varphi_i) \quad \forall i = 1, \dots, N. \tag{1.9}$$

Let us define the *stiffness matrix*

$$\mathbb{S} = \{S_{ij}\}_{i,j=1}^{N}, \quad S_{ij} = (\nabla\varphi_j, \nabla\varphi_i), \tag{1.10}$$

the *solution vector* $\boldsymbol{u} = \{u_i\}_{i=1}^{N}$ and the right-hand side

$$\boldsymbol{g} = \{g_i\}_{i=1}^{N}, \quad g_i = (g, \varphi_i),\ i = 1, \dots, N. \tag{1.11}$$

Then problem (1.9) is equivalent to the linear algebraic system

$$\mathbb{S}\boldsymbol{u} = \boldsymbol{g}. \tag{1.12}$$

Therefore, in order to implement the conforming piecewise linear FEM for (1.3), we have to

- evaluate $\mathbb{S}$ and $\boldsymbol{g}$ according to (1.10) – (1.11),
- solve the algebraic system (1.12),
- reconstruct the approximate solution $u_h \in V_h$ from $\boldsymbol{u}$ using (1.8).

1.1.4 Implementation

Data structures

In order to present the algorithms, we introduce global data structures defining the triangulation $\mathscr{T}_h$. For simplicity, we restrict to $d = 2$.

Let N denote the number of inner vertices of $\mathscr{T}_h$ and Nb denote the number of vertices of $\mathscr{T}_h$ lying on $\partial\Omega$. Let the real arrays `xp(1:N+Nb)` and `yp(1:N+Nb)` be the x_1- and x_2-coordinates of the nodes of mesh $\mathscr{T}_h$, indexed by $i = 1, \dots, N + Nb$, respectively. We assume that the first N components of `xp(:)` and `yp(:)` correspond to inner vertices and the last Nb components of `xp(:)` and `yp(:)` to boundary vertices.

Furthermore, let the vertices of triangles $T_i \in \mathscr{T}_h,\ i = 1, \dots, M$ be indexed by the (two-dimensional) array `node(1:M,1:3)` of integers. Therefore, indices of vertices of T_i are stored in `node(i,1)`, `node(i,2)` and `node(i,3)`. Obviously, if some vertex of T_i lies on $\partial\Omega$ then the corresponding value satisfies `node(i,j)>N`.

For simplicity, we assume that `xp(:), yp(:), node(:,:)` are global arrays so that we have access to them from any subroutine.

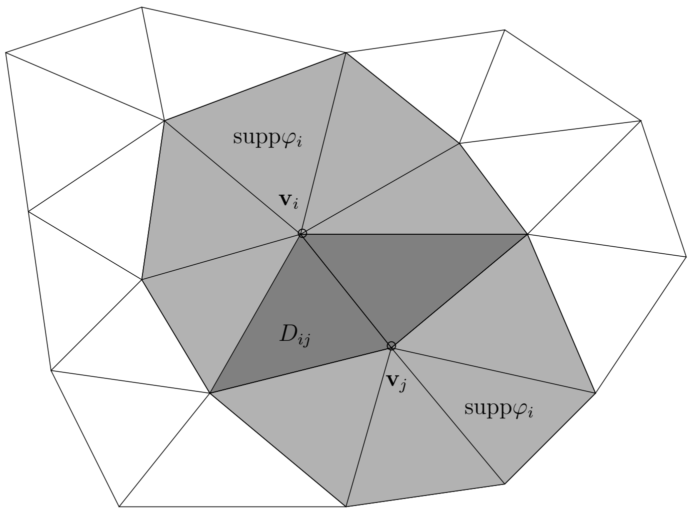


Figure 1.3: The volume $D_{ij} = \text{supp}(\varphi_i) \cap \text{supp}(\varphi_j)$ for nodes $\mathbf{v}_i$ and $\mathbf{v}_j$ connected by an edge

Setting of the stiffness matrix – non-optimized algorithm

Here we present a non-optimized algorithm for the evaluation of the entries of the stiffness matrix $\mathbb{S}$. According to (1.10), the entries of the stiffness matrix satisfy

$$S_{ij} = \int_\Omega \nabla\varphi_j \cdot \nabla\varphi_i \, \mathrm{d}\mathbf{x}. \tag{1.13}$$

Since the support of each $\varphi_i,\ i = 1,\dots,N$ is located around the corresponding node $\mathbf{v}_i$, we can distinguish three cases:

$$S_{ij} = \begin{cases} \int_{\text{supp}(\varphi_i)} |\nabla\varphi_i|^2 \, \mathrm{d}\mathbf{x} & \text{if } i = j, \\ \\ \int_{D_{ij}} \nabla\varphi_j \cdot \nabla\varphi_i \, \mathrm{d}\mathbf{x} & \text{if } \mathbf{v}_i \text{ and } \mathbf{v}_j \text{ are connected by an edge,} \\ \\ 0 & \text{otherwise.} \end{cases} \tag{1.14}$$

Here, $D_{ij} = \text{supp}(\varphi_i) \cap \text{supp}(\varphi_j)$, see Figure 1.3.

For the diagonal entries of the stiffness matrix $S_{ii},\ i = 1,\dots,N$, we have

$$S_{ii} = \int_{\text{supp}(\varphi_i)} |\nabla\varphi_i|^2 \, \mathrm{d}\mathbf{x} = \sum_{T\subset\text{supp}(\varphi_i)} \int_T |\nabla\varphi_i|^2 \, \mathrm{d}\mathbf{x} = \sum_{T\subset\text{supp}(\varphi_i)} |T| \big|\nabla\varphi_i|_T\big|^2. \tag{1.15}$$

The last equality follows from the fact that φ_i is linear on each $T \subset \text{supp}(\varphi_i)$ and thus $\nabla\varphi_i$ is constant on T. Therefore, it is necessary to evaluate $\nabla\varphi_i$ on triangles having $\mathbf{v}_i$ as an vertex.

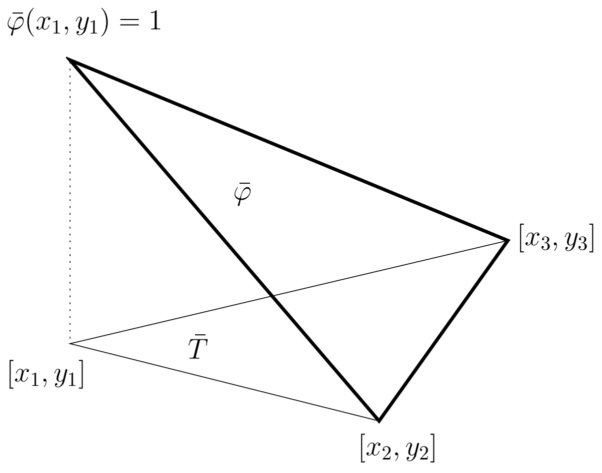


Figure 1.4: Triangle $\bar{T}$ and the function $\bar{\varphi}$

Let us consider a triangle $\bar{T}$ with vertices $[x_1,y_1]$, $[x_2,y_2]$ and $[x_3,y_3]$. Let $\bar{\varphi}(x,y)$ be the linear function on $\bar{T}$ such that

$$\bar{\varphi}(x_1,y_1)=1,\quad \bar{\varphi}(x_2,y_2)=0,\quad \bar{\varphi}(x_3,y_3)=0, \tag{1.16}$$

see Figure 1.4.

It is possible to derive explicit relations for the gradient of $\bar{\varphi}$. Let us put

$$D=\det\begin{pmatrix} x_1 & y_1 & 1\\ x_2 & y_2 & 1\\ x_3 & y_3 & 1\end{pmatrix},\ D_x=\det\begin{pmatrix} 1 & y_1 & 1\\ 0 & y_2 & 1\\ 0 & y_3 & 1\end{pmatrix},\ D_y=\det\begin{pmatrix} x_1 & 1 & 1\\ x_2 & 0 & 1\\ x_3 & 0 & 1\end{pmatrix}. \tag{1.17}$$

Then

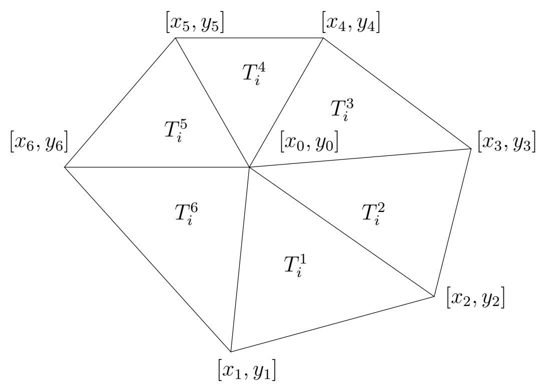
$$\nabla\bar{\varphi}=\left(\frac{D_x}{D},\frac{D_y}{D}\right)=\frac{(y_2-y_3,x_3-x_2)}{2|\bar{T}|}, \tag{1.18}$$

where $|\bar{T}|$ is the area of the triangle $\bar{T}$.

Hence, based on (1.17) – (1.18), we can define the following subroutine `GRADn`, which evaluates the gradient of the linear function on the triangle with vertices $[x_1,y_1]$, $[x_2,y_2]$ and $[x_3,y_3]$ such that this function has value 1 in $[x_1,y_1]$ and vanishes at $[x_2,y_2]$ and $[x_3,y_3]$. We use the convention that $x_1=$ `x1`, $y_1=$ `y1` etc.

```
subroutine GRADn (x1,y1, x2, y2, x3, y3, Dx, Dy)
  real, intent(in) :: x1,y1, x2, y2, x3, y3  ! input parameters
  real, intent(out) :: Dx, Dy                ! output parameters
  real :: D                                  ! local variable

  D = x1*(y2-y3) + x2*(y3-y1) + x3*(y1-y2)
  Dx = (y2 - y3) / D
  Dy = (x3 - x2) / D
end subroutine GRADn
```



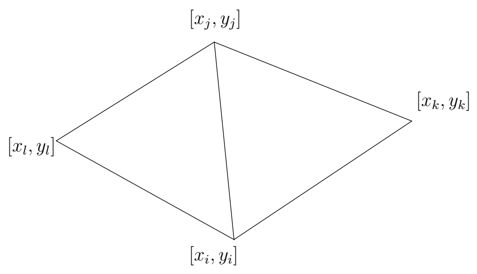


Figure 1.5: Notation for the evaluation of the diagonal entries (left) and off-diagonal entries (right) of the stiff matrix

Now, let us go back to the evaluation of S_{ii}. Let $T_i^j,\ j = 1, \dots, n$ be triangles having the node $\mathbf{v}_i$ as a vertex. Obviously, $\mathrm{supp}(\varphi_i) = \cup_{j=1}^n T_i^j$. We assume that $\mathbf{v}_i$ has coordinates $[x_0, y_0]$. Moreover, the vertices of $T_i^j,\ j = 1, \dots, n$ have coordinates $[x_j, y_j],\ j = 1, \dots, n$, indexed counter-clock-wise, see Figure 1.5, left.

Finally, taking into account (1.15) we can evaluate S_{ii} by the following subroutine:

```
subroutine STIFF_DIAGn(n, x(0:n), y(0:n), sii )
  integer, intent(in) :: n           ! input parameter
  real, intent(in) :: x(0:n), y(0:n) ! input array
  real, intent(out) :: sii           ! output parameter
  real :: D, Dx, Dy                  ! local variables
  integer :: j, j1                   ! local variables

  sii = 0.
  do j=1,n
     j1 = mod(j, n) + 1
     call GRADn(x(0), y(0), x(j), y(j), x(j1), y(j1), Dx, Dy)
     D = x(0)*(y(j)-y(j1) ) + x(j)*(y(j1)-y(0) ) + x(j1)*(y(0)-y(j) )

     sii = sii + (Dx*Dx + Dy*Dy)*D/2  ! D/2 = area of triangle
  end do
end subroutine STIFF_DIAGn
```

Comparing the subroutines `GRADn` and `STIFF_DIAGn`, we observe that the values `Dx` and `Dy` are divided by `D` in `GRADn`, then we square this value and multiply by `D` in `STIFF_DIAGn`. It is more efficient to avoid the division and the following multiplication by the same value. Hence, we can modify the subroutines `GRADn` and `STIFF_DIAGn` by their more efficient variants `GRAD` and `STIFF_DIAG` defined bellow.

```
subroutine GRAD (x1,y1, x2, y2, x3, y3, Dx, Dy)
```

```
   real, intent(in) :: x1,y1, x2, y2, x3, y3
   real, intent(out) :: Dx, Dy

   Dx = (y2 - y3)
   Dy = (x3 - x2)
 end subroutine GRAD
```

and

```
  subroutine STIFF_DIAG( n, x, y, sii )
   integer, intent(in) :: n
   real, intent(in) :: x(0:n), y(0:n)
   real, intent(out) :: sii
   real :: D, Dx, Dy     ! local variables
   integer :: j, j1
   sii = 0.
   do j=1,n
      j1 = mod(j+1, n) + 1
      call GRAD(x(0), y(0), x(j), y(j), x(j1), y(j1), Dx, Dy)

      D = x(0)*(y(j)-y(j1) ) + x(j)*(y(j1)-y(0) ) + x(j1)*(y(0)-y(j) )
      sii = sii + (Dx*Dx + Dy*Dy)/D/2  ! D/2 = area of triangle
   end do
 end subroutine STIFF_DIAG
```

Moreover, we have to evaluate the off-diagonal terms of the stiffness matrix, i.e., S_{ij} where $\mathbf{v}_i$ and $\mathbf{v}_j$ are connected by an edge of $\mathscr{T}_h$. We denote the coordinates of vertices of triangles sharing the edge $\mathbf{v}_i\mathbf{v}_j$ according to Figure 1.5, right. Then we define the subroutine for the evaluation of the off-diagonal terms of the stiffness matrix:

```
  subroutine STIFF_OFFDIAG(xi, yi, xj, yj, xk, yk, xl, yl, sij )
   real, intent(in) :: xi, yi, xj, yj, xk, yk, xl, yl
   real, intent(out) :: sij
   real :: D, Dxi, Dyi, Dxi, Dyj     ! local variables

   sij = 0.
   call GRAD(xi, yi, xk, yk, xj, yj, Dxi, Dyi)
   call GRAD(xj, yj, xi, yi, xk, yk, Dxj, Dyj)

   D = xi*(yk-yj) + xk*(yj-yi) + xj*(yi-yk)
   sij = sij + (Dxi*Dxj + Dyi*Dyj)/D/2  ! D/2 = area of triangle

   call GRAD(xi, yi, xj, yj, xl, yl, Dxi, Dyi)
   call GRAD(xj, yj, xl, yl, xi, yi, Dxj, Dyj)
   D = xi*(yj-yl) + xj*(yl-yi) + xl*(yi-yj)
   sij = sij + (Dxi*Dxj + Dyi*Dyj)/D/2  ! D/2 = area of triangle
```

```
end subroutine STIFF_OFFDIAG
```

Finally, in order to evaluate all entries of the stiffness matrix $\mathbb{S}$ we should call either subroutine `STIFF_DIAG` or `STIFF_OFFDIAG` for all $S_{ij} \neq 0$. However, this requires the setting of appropriate data structures, namely some link to the nodes connected by an edge to a given node (see 1.5, left) and some link between elements sharing a face (see 1.5, right.) Since we are going to introduce a better algorithm in the following section, we shall skip further considerations.

Setting of the stiffness matrix – optimized algorithm

We can notice that when we evaluate all non-vanishing terms of matrix $\mathbb{S}$, we call subroutine `GRAD` for each $T \in \mathscr{T}_h$ several times. It is more efficient to go over all $T \in \mathscr{T}_h$ only one time. This means that we compute the gradients of all possible linear functions on T vanishing in two vertices of T and equal to 1 in the remaining node. Then we evaluate the corresponding entries of the stiffness matrix.

Here we introduce a more efficient algorithm for the evaluation of $\mathbb{S}$. It evaluates all entries of $\mathbb{S}$ which correspond to inner vertices of $\mathscr{T}_h$ given by (1.10). First, we define a subroutine, which evaluates the gradient of three possible restrictions of the test functions on a given triangle.

```
subroutine GRADIENTS( x, y, Dphi )
  real, intent(in) :: x(1:3), y(1:3)
  real, intent(out) :: Dphi(1:3, 1:2)
     ! Dphi(i,j) = derivative of i-th function with respect x_j

  Dphi(1,1) = y(2) - y(3)
  Dphi(1,2) = x(3) - x(2)

  Dphi(2,1) = y(3) - y(1)
  Dphi(2,2) = x(1) - x(3)

  Dphi(3,1) = y(1) - y(2)
  Dphi(3,2) = x(2) - x(1)
end subroutine GRADIENTS
```

Then we have the following subroutine for the evaluation of all entries of $\mathbb{S}$. Let us recall that `xp(:), yp(:), node(:,:)` are global arrays storing the coordinates of vertices and the indices of vertices of triangles, cf. the beginning of Section 1.1.4. Hence, we have access to them from the subroutines without their specification as input parameters.

```
subroutine STIFF(M, N, S )
  integer, intent(in) :: M   ! = number of triangles of triangulation
  integer, intent(in) :: N   ! = number of vertices of triangulation
  real, dimension(1:N, 1:N), intent(out) :: S  ! = stiff matrix
```

```
   real :: Dphi(1:3, 1:2) , D  ! local variables
   integer :: i,j,ki, kj

   S(1:N, 1:N) = 0

   do k=1,M
      x(1:3) =  xp(node(k, 1:3) )
      y(1:3) =  yp(node(k, 1:3) )
      D = x(1)*(y(2)-y(3)) + x(2)*(y(3)-y(1)) + x(3)*(y(1)-y(2))
      call GRADIENTS( x(1:3), y(1:3), Dphi(1:3, 1:2) )

      do ki=1,3
         i = node(k,ki)
         if( i <= N) then          !! inner vertex?
           do kj=1,3
              j = node(k,kj)
              if( j <= N) then     !! inner vertex?
                S(i,j) = S(i,j) + (Dphi(ki,1)* Dphi(kj,1)  &
                                +  Dphi(ki,2)* Dphi(kj,2) )/D/2
              endif
           enddo
         endif
      enddo
   enddo
 end subroutine STIFF
```

Remark 1.1. *The subroutine* `STIFF` *evaluates only entries of the stiff matrix corresponding to the inner vertices of $\mathscr{T}_h$. This is assured by two* `if then` *conditions in subroutine* `STIFF`*. If more general boundary conditions than (1.2) are given also the entries corresponding to the boundary vertices of $\mathscr{T}_h$ have to be evaluated. In this case,* `N` *denotes the number of all vertices including the boundary ones, see Section 1.1.5.*

Setting of the right-hand side

Here we present an algorithm which evaluates the entries of the vector $\boldsymbol{g}$ given by (1.11). Let $i = 1, \dots, N$, we have

$$g_i = (g, \varphi_i) = \sum_{T \subset \mathrm{supp}(\varphi_i)} \int_T g\, \varphi_i \,\mathrm{d}\mathbf{x}, \tag{1.19}$$

where $\mathrm{supp}(\varphi_i)$ is shown in Figure 1.2. Generally, the integrals in (1.19) are evaluated with the aid of numerical quadratures, which are discussed in Section 1.5. Since we considered in this section the piecewise linear approximation, we will consider only the following simple quadrature rule

$$\int_T g(\mathbf{x})\, \varphi(\mathbf{x}) \,\mathrm{d}\mathbf{x} \approx \frac{1}{3}|T| \sum_{l=1}^{3} \varphi(\mathbf{x}_l) g(\mathbf{x}_l), \tag{1.20}$$

where $\mathbf{x}_l,\ l = 1, 2, 3$ are the vertices of a triangle $T \in \mathscr{T}_h$. This quadrature rule integrates linear functions exactly, hence it has first order accuracy.

The application of (1.20) to the evaluation of (1.19) leads to a very simple relation, since $\varphi_i(x_0, y_0) = 1$ and $\varphi_i(x_l, y_l) = 0,\ l = 1, \dots, n$, see Figure 1.5, left. Then we have

$$g_i \approx \frac{1}{3}\mathrm{meas}(\mathrm{supp}(\varphi_i))g(\mathbf{v}_i), \quad i = 1, \dots, N \tag{1.21}$$

where $\mathbf{v}_i,\ i = 1, \dots, N$ are inner vertices of $\mathscr{T}_h$.

Finally, let us note that the numerical quadrature (1.20) is generally not sufficiently accurate.

Solution of the linear algebraic system (1.12)

Once we have evaluated the entries of matrix $\mathbb{S}$ and vector $\boldsymbol{g}$, we have to solve the linear algebraic system (1.12). In the case of the Laplace problem (1.1) the situation is very simple. First we introduce the following term

Definition 1.1. *Let $\mathscr{T}_h$ be a mesh on Ω. Let T and T' be a pair of triangles from $\mathscr{T}_h$ sharing an edge $\mathbf{v}_i\mathbf{v}_j$. Let $\mathbf{v}_i, \mathbf{v}_j, \mathbf{v}_k$ and $\mathbf{v}_j, \mathbf{v}_i, \mathbf{v}_l$ be the vertices of T and T', respectively. We denote by* $\mathrm{SumAngles}(T, T')$ *the sum of angles of T and T' adjacent to vertices $\mathbf{v}_k$ and $\mathbf{v}_l$, respectively. We say that $\mathscr{T}_h$ is of the Delaunay type if* $\mathrm{SumAngles}(T, T') \le \pi$ *for each pair of triangles T, T' from $\mathscr{T}_h$ sharing an edge.*

It is possible to show that if $\mathscr{T}_h$ is of Delaunay type then the corresponding stiffness matrix $\mathbb{S}$ is symmetric and diagonally dominant. In this case an arbitrary direct or iterative solver can be used with success for the solution of (1.12).

Reconstruction of the approximate solution

Once we have found the vector $\boldsymbol{u}$ which is the solution of (1.12), we will reconstruct the piecewise linear function $u_h \in V_h$ from $\boldsymbol{u}$ using (1.8). However, in practice we need not know an explicit relation for u_h. Usually, we require a visualization of the solution and/or an evaluation of some characteristics. Some basic visualization techniques are mentioned in Section 1.6.

1.1.5 Treatment of general boundary conditions

In the previous sections, the homogeneous Dirichlet boundary condition was considered. However, in practice, more general boundary conditions have to be taken into account.

Let Ω be a bounded polygonal domain in $\mathbb{R}^d,\ d = 2, 3$, with a boundary $\partial\Omega$ which consists of two disjoint parts $\partial\Omega_D$ and $\partial\Omega_N$. We seek a function $u : \Omega \to \mathbb{R}$ such that

$$\begin{aligned} -\Delta u(x) &= g(x), \quad x \in \Omega, && (1.22)\\ u(x) &= u_D(x), \quad x \in \partial\Omega_D, && (1.23)\\ \nabla u(x) \cdot \boldsymbol{n} &= g_N(x), \quad x \in \partial\Omega_N, && (1.24) \end{aligned}$$

where $g \in L^2(\Omega)$, $g_N \in L^2(\partial\Omega)$ and u_D is a trace of some $u^* \in H^1(\Omega)$.

Let

$$V := \{v \in H^1(\Omega), v|_{\partial\Omega_D} = 0 \text{ in the sense of traces }\} \tag{1.25}$$

then the *week formulation* of (1.22) – (1.24) reads

$$\text{find } u \in H^1(\Omega) : \quad \begin{aligned} & u - u^* \in V, \\ & (\nabla u, \nabla v) = (g, v) + (g_N, v)_{L^2(\partial\Omega_N)} \quad \forall v \in V. \end{aligned} \tag{1.26}$$

Let $\mathscr{T}_h$ be a mesh reflecting that $\partial\Omega$ consists of two disjoint parts $\partial\Omega_D$ and $\partial\Omega_N$. Let X_h be a finite dimensional subspace of $H^1(\Omega)$ constructed over the mesh $\mathscr{T}_h$ and $V_h \subset X_h$ its subspace such that $v_h|_{\partial\Omega_D} = 0 \; \forall v_h \in V_h$. The *finite element* approximation (1.26) can be written as

$$\text{find } u_h \in X_h : \quad \begin{aligned} & (u_h - u_D, v_h)_{L^2(\partial\Omega_D)} = 0 \quad \forall v_h \in X_h, \\ & (\nabla u_h, \nabla v_h) = (g, v_h) + (g_N, v_h)_{L^2(\partial\Omega_N)} \quad \forall v_h \in V_h. \end{aligned} \tag{1.27}$$

The numerical implementation of (1.27) is more complicated than the implementation of (1.4). In this place we skip the evaluation of the boundary integral on the right hand side of (1.27) and refer to Section 1.4.6. We focus on satisfying the Dirichlet boundary condition on $\partial\Omega_D$.

We use the following notation for vertices of $\mathscr{T}_h$ which differs from the notation introduced in the previous sections. We denote by $\{\mathbf{v}_i\}_{i=1}^{N}$ the set of all vertices of $T \in \mathscr{T}_h$ lying inside of Ω or on $\partial\Omega_N$ and by $\{\mathbf{v}_i\}_{i=N+1}^{N+N_b}$ a set of all vertices of $T \in \mathscr{T}_h$ lying on $\partial\Omega_D$.

Similarly as in the previous, we restrict to the <u>piecewise linear</u> approximations. Therefore, each function from X_h is uniquely given by its values in $\{\mathbf{v}_i\}_{i=1}^{N+N_b}$ and each function from V_h is uniquely given by its values at $\{\mathbf{v}_i\}_{i=1}^{N}$. We denote by $\varphi_i \in X_h$ the basis functions corresponding to $\mathbf{v}_i$, $i = 1, \dots, N + N_b$ given by

$$\varphi_i \in X_h, \quad \varphi_i(\mathbf{v}_j) = \delta_{ij}, \quad i, j = 1, \dots, N + N_b. \tag{1.28}$$

First, we introduce a piecewise linear approximation of u_D from the boundary condition. We define a function $u_{D,h} \in X_h \setminus V_h$ by

$$(u_{D,h} - u_D, v_h)_{L^2(\partial\Omega_D)} = 0 \quad \forall v_h \in X_h \setminus V_h. \tag{1.29}$$

Such $u_{D,h}$ can be simply constructed since X_h is a finite dimensional spaces and then (1.29) represents a system of linear algebraic equations. Namely, we have

$$u_{D,h}(\mathbf{x}) = \sum_{j=N+1}^{N_b} u_{D,j}\varphi_j(\mathbf{x}), \tag{1.30}$$

therefore, we have from (1.29) – (1.30) the following system of linear algebraic equations

$$\sum_{j=N+1}^{N_b} u_{D,j}(\varphi_j, \varphi_i)_{L^2(\partial\Omega_D)} = (u_D, \varphi_i)_{L^2(\partial\Omega_D)}, \quad i = N + 1, \dots, N_b. \tag{1.31}$$

The evaluation of terms $(\varphi_j, \varphi_i)_{L^2(\partial\Omega_D)}$, $i, j = N+1, \dots, N_b$ is simple since these terms are non-vanishing only if $\mathbf{v}_i$ and $\mathbf{v}_i$ are connected by an edge on $\partial\Omega_D$. Moreover, the evaluation of $(u_D, \varphi_i)_{L^2(\partial\Omega_D)}$, $i = N+1, \dots, N_b$ reduces to an integration over two edges on $\partial\Omega_D$ having $\mathbf{v}_i$ as an end-point.

The solution of (1.27) can be written in the form

$$u_h(\mathbf{x}) = \sum_{j=1}^{N} u_j \varphi_i(\mathbf{x}) + \sum_{j=N+1}^{N_b} u_{D,j} \varphi_j(\mathbf{x}) = \sum_{j=1}^{N} u_j \varphi_i(\mathbf{x}) + u_{D,h}, \tag{1.32}$$

where the coefficients $u_{D,j}$, $j = N+1, \dots, N_b$ are known and the coefficients u_j, $j = 1, \dots, N$ are unknown. Inserting (1.32) into (1.27) and putting $v_h := \varphi_i$, we have

$$\sum_{j=1}^{N} u_j (\nabla\varphi_j, \nabla\varphi_i) = (g, \varphi_i) + (g_N, \varphi_i)_{L^2(\partial\Omega_N)} - (\nabla u_{D,h}, \nabla\varphi_i), \ i = 1, \dots, N, \tag{1.33}$$

which is very close to the problem (1.9), where the right-hand-side contains additional terms arising from the Dirichlet and the Neumann boundary conditions. Moreover, the symbol N has a slightly different meanings in (1.10) and (1.33), in the former case it denotes the number of interior vertices, in the latter case the number of inner and "Neumann" nodes. However, all subroutines presented in Section 1.1.4 can be employed with these minor changes.

1.2 General FE solution of the model problem

In Section 1.1 we described the implementation of the conforming piecewise linear finite element approximation of problem (1.1) – (1.2). Here we consider a higher degree of polynomial approximation. The implementation is more complicated since the gradient of test function is not constant over $T \in \mathscr{T}_h$.

It is more comfortable to employ the concept of the reference element. This approach allows us to deal with more general elements than triangles, e.g., quadrilaterals, curvilinear elements, etc.

1.2.1 Definition of reference elements

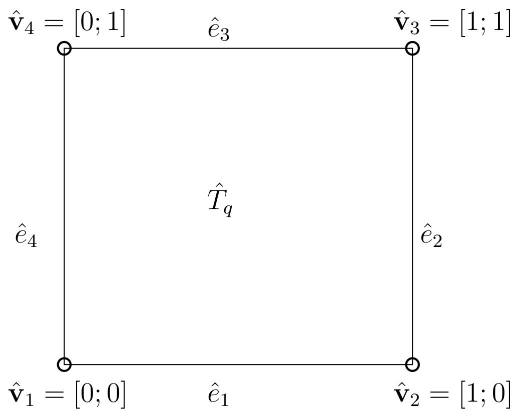
In these lecture notes, we consider

- the *reference simplex* (triangle for $d = 2$ and tetrahedron for $d = 3$)

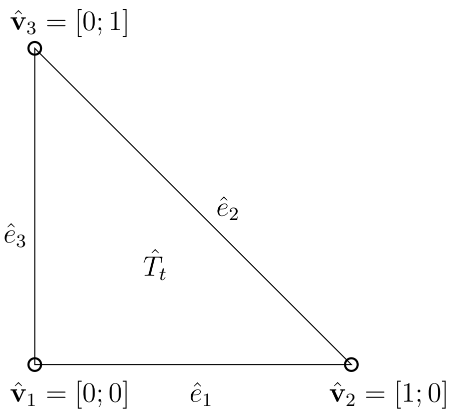
$$\widehat{T}_t := \left\{ \hat{\mathbf{x}} = (x_1, \dots, x_d)), \ x_i \geq 0, \ i = 1, \dots, d, \ \sum_{i=1}^{d} x_i \leq 1 \right\} \tag{1.34}$$

- the *reference parallelogram* (square for $d = 2$ and cube for $d = 3$)

$$\widehat{T}_q := \{ \hat{\mathbf{x}} = (x_1, \dots, x_d)), \ 0 \leq x_i \leq 1, \ i = 1, \dots, d, \} . \tag{1.35}$$



$\hat{\mathbf{v}}_4 = [0;1]$
$\hat{e}_3$
$\hat{\mathbf{v}}_3 = [1;1]$
$\widehat{T}_q$
$\hat{e}_4$
$\hat{e}_2$
$\hat{\mathbf{v}}_1 = [0;0]$
$\hat{e}_1$
$\hat{\mathbf{v}}_2 = [1;0]$



$\hat{\mathbf{v}}_3 = [0;1]$
$\hat{e}_2$
$\hat{e}_3$
$\widehat{T}_t$
$\hat{\mathbf{v}}_1 = [0;0]$
$\hat{e}_1$
$\hat{\mathbf{v}}_2 = [1;0]$

Figure 1.6: Two-dimensional reference elements: quadrilateral $\widehat{T}_q$ (left) and triangle $\widehat{T}_t$ (right) with the vertices $\hat{\mathbf{v}}_i$ and the edges $\hat{e}_i$

For simplicity, $\widehat{T}$ denotes either the reference simplex or the reference hexagon based on the context. We call $\widehat{T}$ the *reference element.*

Moreover, we denote by $\hat{\mathbf{v}}_1, \dots, \hat{\mathbf{v}}_{d+1}$ the vertices and by $\hat{e}_1, \dots, \hat{e}_{d+1}$ the edges of $\widehat{T}_t$, similarly by $\hat{\mathbf{v}}_1, \dots, \hat{\mathbf{v}}_{2^d}$ the vertices and by $\hat{e}_1, \dots, \hat{e}_{2^d}$ the edges of $\widehat{T}_q$. For $d = 2$, we assume that the numbering is oriented counterclockwise, see Figure 1.6, which shows the two-dimensional reference triangle and the reference quadrilateral.

We assume that for each element $T \in \mathscr{T}_h$ there exist a mapping

$$F_T = F_T(\hat{\mathbf{x}}) = (F_{T,1}(\hat{\mathbf{x}}), \dots, F_{T,d}(\hat{\mathbf{x}})) : \widehat{T} \to \mathbb{R}^d \quad \text{such that} \quad F_T(\widehat{T}) = T. \tag{1.36}$$

If F_T is an affine (linear) mapping then T is either a triangle or a parallelogram based on the type of $\widehat{T}$. Generally, the construction of the mappings F_T is simple for several types of elements:

- *Triangles*: Let T be a triangle defined by its vertices $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$. Then the linear mapping

$$F_T(\hat{\mathbf{x}}) = F_T(\hat{x}_1, \hat{x}_2) = \mathbf{x}_1 + \hat{x}_1(\mathbf{x}_2 - \mathbf{x}_1) + \hat{x}_2(\mathbf{x}_3 - \mathbf{x}_1) \tag{1.37}$$

  maps $\widehat{T}_t$ on T and moreover, $F_T(0,0) = \mathbf{x}_1$, $F_T(1,0) = \mathbf{x}_2$ and $F_T(0,1) = \mathbf{x}_3$.

- *Tetrahedron*: Let T be a tetrahedron defined by its vertices $\mathbf{x}_1$, $\mathbf{x}_2$, $\mathbf{x}_3$ and $\mathbf{x}_4$ Then the linear mapping

$$F_T(\hat{\mathbf{x}}) = F_T(\hat{x}_1, \hat{x}_2, \hat{x}_3) = \mathbf{x}_1 + \hat{x}_1(\mathbf{x}_2 - \mathbf{x}_1) + \hat{x}_2(\mathbf{x}_3 - \mathbf{x}_1) + \hat{x}_3(\mathbf{x}_4 - \mathbf{x}_1) \tag{1.38}$$

  maps $\widehat{T}_q$ on T and moreover, $F_T(0,0,0) = \mathbf{x}_1$, $F_T(1,0,0) = \mathbf{x}_2$, $F_T(0,1,0) = \mathbf{x}_3$ and $F_T(0,0,1) = \mathbf{x}_4$.

- *Quadrilaterals*: Let T be a quadrilateral defined by its vertices $\mathbf{x}_1$, $\mathbf{x}_2$, $\mathbf{x}_3$ and $\mathbf{x}_4$. Then the mapping

$$F_T(\hat{\mathbf{x}}) = F_T(\hat{x}_1, \hat{x}_2) = \mathbf{x}_1 + \hat{x}_1(\mathbf{x}_2 - \mathbf{x}_1) + \hat{x}_2(\mathbf{x}_3 - \mathbf{x}_1) + \hat{x}_1\hat{x}_2(\mathbf{x}_4 - \mathbf{x}_2 - \mathbf{x}_3 + \mathbf{x}_1) \tag{1.39}$$

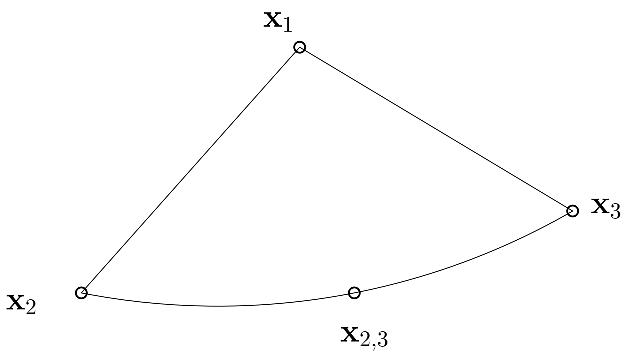


Figure 1.7: One-edge P^2-curvilinear triangles

maps $\widehat{T}_q$ on T and moreover, $F_T(0,0) = \mathbf{x}_1$, $F_T(1,0) = \mathbf{x}_2$, $F_T(0,1) = \mathbf{x}_3$ and $F_T(1,1) = \mathbf{x}_4$. Let us note that if T is a parallelogram then $\mathbf{x}_4 - \mathbf{x}_2 - \mathbf{x}_3 + \mathbf{x}_1 = \mathbf{0}$ and (1.39) is identical with (1.37).

- *One-edge P^2-curvilinear triangles*: This types of elements are often used for an approximation of non-polygonal boundaries. Let T be a one-edge P^2-curvilinear triangle, which is defined by its vertices $\mathbf{x}_1$, $\mathbf{x}_2$, $\mathbf{x}_3$ and the node $\mathbf{x}_{2,3}$ lying "not far" from $(\mathbf{x}_2 + \mathbf{x}_3)/2$. Element T is bounded by two straight lines $\mathbf{x}_1\mathbf{x}_2$ and $\mathbf{x}_1\mathbf{x}_3$ and one quadratic curve passing $\mathbf{x}_2, \mathbf{x}_{2,3}, \mathbf{x}_3$, see Figure 1.7. Then the quadratic mapping

$$F_T(\hat{x}_1, \hat{x}_2) = \mathbf{x}_1 + \hat{x}_1(\mathbf{x}_2 - \mathbf{x}_1) + \hat{x}_2(\mathbf{x}_3 - \mathbf{x}_1) + 4\hat{x}_1\hat{x}_2(\mathbf{x}_{2,3} - (\mathbf{x}_2 + \mathbf{x}_3)/2) \quad (1.40)$$

  maps $\widehat{T}_q$ on T. Moreover, $F_T(0,0) = \mathbf{x}_1$, $F_T(1,0) = \mathbf{x}_2$, $F_T(0,1) = \mathbf{x}_3$ and $F_T(1/2,1/2) = \mathbf{x}_{2,3}$. Let us note that if $\mathbf{x}_{2,3} = (\mathbf{x}_2 + \mathbf{x}_3)/2$ then T is a triangle and (1.40) is identical with (1.37).

It is possible to define mappings F_K for more general elements. Let us note that in the following sections we require that F_K is invertible and continuously differentiable on $\widehat{T}$. The existence of F_K^{-1} follows from the assumption that the *Jacobi matrix*

$$J_{F_T}(\hat{\mathbf{x}}) := \frac{D}{D\hat{\mathbf{x}}} F_T(\hat{\mathbf{x}}) \qquad (1.41)$$

does not change its sign on $\widehat{T}$. Very often, only polynomial mappings are considered for practical reasons.

1.2.2 Evaluation of integrals on the reference element

The concept of the reference elements is based in the Theorem of integration by substitution, which we present in the following form.

Theorem 1.1. *Let T be the image of $F_T(\widehat{T})$ where $\widehat{T}$ is the reference element and F_T is a continuously differentiable mapping. Let $f(\mathbf{x}) : T \to \mathbb{R}$ be an integrable function defined on T. We define the function $\hat{f} : \widehat{T} \to \mathbb{R}$ by $\hat{f}(\hat{\mathbf{x}}) = f(\mathbf{x})$ where $\mathbf{x} = F_T(\hat{\mathbf{x}})$. Then*

$$\int_T f(\mathbf{x})\,\mathrm{d}\mathbf{x} = \int_{\widehat{T}} \hat{f}(\hat{\mathbf{x}})|\det J_{F_T}(\hat{\mathbf{x}})|\,\mathrm{d}\hat{\mathbf{x}}, \tag{1.42}$$

where J_{F_T} is the Jacobi matrix of F_T.

Let us note that if F_T is a linear mapping (i.e., T is a simplex or a parallelogram) then J_F is a constant matrix and $|\det J_{F_T}| = 2|T|$, where $|T|$ denotes the d-dimensional Lebesgue measure of T.

Theorem 1.1 gives us a tool for the evaluation of the various type of integrals in the implementation of finite element and discontinuous Galerkin methods. Instead of integration over T we integrate over $\widehat{T}$ which is generally easier. This is demonstrated in the following paragraphs.

Let $T \in \mathscr{T}_h$ and φ be a function defined on T. Our aim is to compute

$$\int_T \frac{\partial \varphi(\mathbf{x})}{\partial x_i}\,\mathrm{d}\mathbf{x}. \tag{1.43}$$

Let $\hat{\varphi} : \widehat{T} \to \mathbb{R}$ be defined as usual by $\hat{\varphi}(\hat{\mathbf{x}}) = \varphi(\mathbf{x})$ where $\mathbf{x} = F_T(\hat{\mathbf{x}})$,

$$F_T = (F_{T,1}, \dots, F_{T,d})$$

and $T = F_T(\widehat{T})$. Moreover, we have $\hat{\mathbf{x}} = F_T^{-1}(\mathbf{x})$ where

$$F_T^{-1} = (F_{T,1}^{-1}, \dots, F_{T,d}^{-1})$$

is the inverse mapping of F_T. With the aid of Theorem 1.1 and the chain rule, we have

$$\begin{aligned} \int_T \frac{\partial}{\partial x_i}\varphi(\mathbf{x})\,\mathrm{d}\mathbf{x} &= \int_{\widehat{T}} \frac{\partial}{\partial x_i}\hat{\varphi}(\hat{\mathbf{x}})|\det J_{F_T}(\hat{\mathbf{x}})|\,\mathrm{d}\hat{\mathbf{x}} \qquad (1.44)\\ = \int_{\widehat{T}} \sum_{k=1}^{d} \frac{\partial \hat{\varphi}(\hat{\mathbf{x}})}{\partial \hat{x}_k}\frac{\partial \hat{x}_k}{\partial x_i}|\det J_{F_T}(\hat{\mathbf{x}})|\,\mathrm{d}\hat{\mathbf{x}} &= \int_{\widehat{T}} \sum_{k=1}^{d} \frac{\partial \hat{\varphi}(\hat{\mathbf{x}})}{\partial \hat{x}_k}\frac{\partial F_{T,k}^{-1}}{\partial x_i}|\det J_{F_T}(\hat{\mathbf{x}})|\,\mathrm{d}\hat{\mathbf{x}}. \end{aligned}$$

Therefore, we have to evaluate the partial derivative of F_T^{-1}. It is possible to use the following formulas. Let $i, k = 1, \dots, d$, then

$$\begin{aligned} x_k &= F_{T,k}(F_T^{-1}(\mathbf{x})), \qquad \Big|\frac{\partial}{\partial x_i} \\ \frac{\partial}{\partial x_i}x_k &= \frac{\partial}{\partial x_i}F_{T,k}(F_T^{-1}(\mathbf{x})), \\ \delta_{ik} &= \sum_{l=1}^{d} \frac{\partial F_{T,k}}{\partial \hat{x}_l}\frac{\partial F_{T,l}^{-1}}{\partial x_i}, \\ \delta_{ik} &= \sum_{l=1}^{d} (J_{F_T})_{kl}\,(J_{F_T^{-1}})_{li}, \qquad (1.45) \end{aligned}$$

where $(J_{F_T})_{kl},\ k,l = 1,\dots,d$ and $(J_{F_T^{-1}})_{li},\ l,i = 1,\dots,d$ denote entries of the Jacobi matrices J_{F_T} and $J_{F_T^{-1}}$, respectively. Relation (1.45) implies

$$\mathbb{I} = J_{F_T} J_{F_T^{-1}} \qquad \Longrightarrow \qquad J_{F_T^{-1}} = (J_{F_T})^{-1}, \tag{1.46}$$

where $\mathbb{I}$ is the identity matrix. Therefore, instead of evaluation of the Jacobi matrix of F_T^{-1} it is sufficient to evaluate $(J_{F_T})^{-1}$ = the inversion of J_{F_T}. This is simple, since J_{F_T} is (usually) 2×2 or 3×3 matrix.

Therefore, from (1.44) and (1.46) we have

$$\begin{aligned}\int_T \frac{\partial}{\partial x_i}\varphi(\mathbf{x})\,\mathrm{d}\mathbf{x} &= \int_{\widehat{T}} \sum_{k=1}^{d} \frac{\partial \hat\varphi(\hat{\mathbf{x}})}{\partial \hat x_k} (J_{F_T}(\hat{\mathbf{x}})^{-1})_{ki} |\det J_{F_T}(\hat{\mathbf{x}})|\,\mathrm{d}\hat{\mathbf{x}} \\ &= \int_{\widehat{T}} \sum_{k=1}^{d} \frac{\partial \hat\varphi(\hat{\mathbf{x}})}{\partial \hat x_k} (J_{F_T}(\hat{\mathbf{x}})^{-\mathrm{T}})_{ik} |\det J_{F_T}(\hat{\mathbf{x}})|\,\mathrm{d}\hat{\mathbf{x}} \\ &= \int_{\widehat{T}} \left(J_{F_T}(\hat{\mathbf{x}})^{-\mathrm{T}} \widehat{\nabla}\hat\varphi(\hat{\mathbf{x}})\right)_i |\det J_{F_T}(\hat{\mathbf{x}})|\,\mathrm{d}\hat{\mathbf{x}},\end{aligned} \tag{1.47}$$

where $J_{F_T}^{-\mathrm{T}}$ is the transposed matrix to $J_{F_T}^{-1}$, $(J_{F_T}^{-\mathrm{T}}\widehat{\nabla}\hat\varphi)_i$ denotes the i-th component of vector $J_{F_T}^{-\mathrm{T}}\widehat{\nabla}\hat\varphi$ arising from the matrix-vector product of $J_{F_T}^{-\mathrm{T}}$ by $\widehat{\nabla}\hat\varphi$ and $\widehat{\nabla}$ is the reference gradient operator given by

$$\widehat{\nabla} := \left(\frac{\partial}{\partial \hat x_1}, \dots, \frac{\partial}{\partial \hat x_d}\right). \tag{1.48}$$

In the same manner it is possible to derive, e.g.,

$$\int_T \frac{\partial \varphi_a(\mathbf{x})}{\partial x_i}\frac{\partial \varphi_b(\mathbf{x})}{\partial x_j}\,\mathrm{d}\mathbf{x} = \int_{\widehat{T}} \left(J_{F_T}(\hat{\mathbf{x}})^{-\mathrm{T}}\widehat{\nabla}\hat\varphi_a(\hat{\mathbf{x}})\right)_i \left(J_{F_T}(\hat{\mathbf{x}})^{-\mathrm{T}}\widehat{\nabla}\hat\varphi_b(\hat{\mathbf{x}})\right)_j |\det J_{F_T}(\hat{\mathbf{x}})|\,\mathrm{d}\hat{\mathbf{x}}$$

or more simply

$$\int_T \frac{\partial \varphi_a}{\partial x_i}\frac{\partial \varphi_b}{\partial x_j}\,\mathrm{d}\mathbf{x} = \int_{\widehat{T}} \left(J_{F_T}^{-\mathrm{T}}\widehat{\nabla}\hat\varphi_a\right)_i \left(J_{F_T}^{-\mathrm{T}}\widehat{\nabla}\hat\varphi_b\right)_j |\det J_{F_T}|\,\mathrm{d}\hat{\mathbf{x}}, \tag{1.49}$$

which is type of integral appearing in the evaluation of the stiffness matrix.

1.2.3 Grids and functional spaces

Again, we consider the model Poisson problem (1.1) – (1.2), the weak formulation (1.3) and the finite element discretization (1.4). In contrast to Section 1.1, we consider more general partitions of Ω and finite element spaces. Let a grid $\mathscr{T}_h$ be a set of non-overlapping elements $T \in \mathscr{T}_h$ such that

$$\overline{\Omega} = \bigcup_{T\in\mathscr{T}_h} T. \tag{1.50}$$

We consider two following type of grids.

Triangular grid

Each element $T \in \mathscr{T}_h$ is an image of the reference triangle (for $d = 2$) or the reference tetrahedron (for $d = 3$) given by a polynomial mapping $F_T : \widehat{T}_t \to \mathbb{R}$, $T \in \mathscr{T}_h$. If F_T is a linear mapping then T is a triangle (for $d = 2$) or a tetrahedron (for $d = 3$). If F_T has the form (1.40) then T is an one-edge P^2-curvilinear triangle. The approximate solution is sought in the functional space

$$\begin{aligned} V_h^{t,p} &:= \{v_h,\ v_h \in C(\overline{\Omega}) \cap H_0^1(\Omega),\ v_h|_T \circ F_T \in P^{t,p}(\widehat{T}_t)\ \forall T \in \mathscr{T}_h\}, \qquad (1.51)\\ P^{t,p}(\omega) &:= \{\hat{z} : \omega \to \mathbb{R},\ \hat{z}(\hat{x}_1, \dots, \hat{x}_d) = \sum_{i_1,\dots,i_d=0}^{i_1+\dots+i_d \le p} a_{i_1,\dots,i_d} \hat{x}_1^{i_1} \dots \hat{x}_d^{i_d},\ a_{i_1,\dots,i_d} \in \mathbb{R}\}, \end{aligned}$$

where $\omega \subset \mathbb{R}^d$ is an arbitrary domain. Let us note that the notation $v_h|_T \circ F_T \in P^{t,p}(\widehat{T}_t)$ means that there exists a function $\hat{v}_h \in P^{t,p}(\widehat{T}_t)$ such that

$$v_h(\mathbf{x}) = v_h(F_T(\hat{\mathbf{x}})) = \hat{v}_h(\hat{\mathbf{x}}) \qquad \forall \hat{\mathbf{x}} \in \widehat{T}_t. \qquad (1.52)$$

Obviously, if F_T is a linear mapping then $v_h|_T \in P^{t,p}(T)$. Otherwise, (e.g., T is a curvilinear triangle), $v_h|_T \notin P^{t,p}(T)$.

The spaces $P^{t,p}(\widehat{T}_t),\ p = 1, 2, 3$ can be expressed (for $d = 2$) as

$$\begin{aligned} P^{t,1}(\widehat{T}_t) &= \mathrm{span}\{1,\ \hat{x}_1, \hat{x}_2\}, \qquad (1.53)\\ P^{t,2}(\widehat{T}_t) &= \mathrm{span}\{1,\ \hat{x}_1, \hat{x}_2,\ \hat{x}_1^2, \hat{x}_1\hat{x}_2, \hat{x}_2^2\},\\ P^{t,3}(\widehat{T}_t) &= \mathrm{span}\{1,\ \hat{x}_1, \hat{x}_2,\ \hat{x}_1^2, \hat{x}_1\hat{x}_2, \hat{x}_2^2,\ \hat{x}_1^3, \hat{x}_1^2\hat{x}_2, \hat{x}_1\hat{x}_2^2, \hat{x}_2^3\}, \end{aligned}$$

where span$\{S\}$ is the linear hull of the set $\{S\}$. Obviously, $\hat{N}^{t,p} := \dim P^{t,p}(\widehat{T}_t) = (p+1)(p+2)/2$ for $d = 2$ and $\hat{N}^{t,p} := \dim P^{t,p}(\widehat{T}_t) = (p+1)(p+2)(p+3)/6$ for $d = 3$.

Quadrilateral grids

Each element $T \in \mathscr{T}_h$ is an image of the reference square (for $d = 2$) or the reference cube (for $d = 3$) given by a polynomial mapping $F_T : \widehat{T}_q \to \mathbb{R}$, $T \in \mathscr{T}_h$. If F_T is a linear mapping then T is a parallelogram. The approximate solution is sought in the functional space

$$\begin{aligned} V_h^{q,p} &:= \{v_h,\ v_h \in C(\overline{\Omega}) \cap H_0^1(\Omega),\ v_h|_T \circ F_T \in P^{q,p}(\widehat{T}_q)\ \forall T \in \mathscr{T}_h\}, \qquad (1.54)\\ P^{q,p}(\omega) &:= \{\ \hat{z} : \omega \to \mathbb{R},\ \hat{z}(x_1, \dots, x_d) = \sum_{i_1,\dots,i_d=0}^{p} a_{i_1,\dots,i_d} x_1^{i_1} \dots x_d^{i_d},\ a_{i_1,\dots,i_d} \in \mathbb{R}\}, \end{aligned}$$

where $\omega \subset \mathbb{R}^d$ is an arbitrary domain.

The spaces $P^{q,p}(\widehat{T}_q),\ p = 1, 2, 3$ can be expressed (for $d = 2$) as

$$\begin{aligned} P^{q,1}(\widehat{T}_q) &= \mathrm{span}\{1,\ \hat{x}_1, \hat{x}_2, \hat{x}_1\hat{x}_2\}, \qquad (1.55)\\ P^{q,2}(\widehat{T}_q) &= \mathrm{span}\{1,\ \hat{x}_1, \hat{x}_2, \hat{x}_1\hat{x}_2,\ \hat{x}_1^2, \hat{x}_1^2\hat{x}_2, \hat{x}_1^2\hat{x}_2^2, \hat{x}_1\hat{x}_2^2, \hat{x}_2^2\},\\ P^{q,3}(\widehat{T}_q) &= \mathrm{span}\{1,\ \hat{x}_1, \hat{x}_2, \hat{x}_1\hat{x}_2,\ \hat{x}_1^2, \hat{x}_1^2\hat{x}_2, \hat{x}_1^2\hat{x}_2^2, \hat{x}_1\hat{x}_2^2, \hat{x}_2^2,\\ &\qquad \hat{x}_1^3, \hat{x}_1^3\hat{x}_2, \hat{x}_1^3\hat{x}_2^2, \hat{x}_1^3\hat{x}_2^3, \hat{x}_1^2\hat{x}_2^3, \hat{x}_1\hat{x}_2^3, \hat{x}_2^3\}. \end{aligned}$$

Obviously, $\hat{N}^{q,p} := \dim P^{q,p}(\widehat{T}_q) = (p+1)^d$ for $d = 2, 3$.

Since most of the following considerations are valid for triangular as well as quadrilateral grids we call $\mathscr{T}_h$ a triangulation in both cases. Moreover, the corresponding reference element is denoted by $\widehat{T}$, $P^p(\widehat{T})$ the corresponding space of polynomial functions on $\widehat{T}$ with the dimension $\hat{N}$ and the corresponding finite element space is denoted by V_h.

1.2.4 Implementation

Basis of V_h

Let N denote the dimension of the space V_h and $B_h := \{\varphi_i(\mathbf{x})\}_{i=1}^N$ be its basis. The construction of B_h will be discussed in Section 1.3. Here we only note that we construct the set of *reference shape functions* $\hat{B} := \{\hat{\varphi}_j,\ \hat{\varphi}_j : \widehat{T} \to \mathbb{R}\}_{j=1}^{\hat{N}}$ such that

- $\hat{\varphi}_j(\hat{\mathbf{x}}) \in P^p(\widehat{T}),\ j = 1, \dots, \hat{N}$, $\hat{N}$ is the dimension of $P^p(\widehat{T})$.
- $\hat{\varphi}_j(\hat{\mathbf{x}}),\ j = 1, \dots, \hat{N}$ are linearly independent,
- $\mathrm{span}(\hat{B}) = P^p(\widehat{T})$.

We call $\hat{B}$ the *reference basis.*

Moreover, in virtue of (1.51) or (1.54), the basis functions $\varphi_i \in B_h$ satisfy: for each $T \in \mathscr{T}_h,\ T \subset \mathrm{supp}(\varphi_i)$ there exists a function $\hat{\varphi}_{i,T} \in \hat{B}$ such that

$$\varphi_i(\mathbf{x})|_T = \varphi_i(F_T(\hat{\mathbf{x}}))|_T = \hat{\varphi}_{i,T}(\hat{\mathbf{x}}) \qquad \forall \hat{\mathbf{x}} \in \widehat{T},\ i = 1, \dots, N. \tag{1.56}$$

More details are given in Section 1.3.

Use of the concept of the reference elements

Our aim is to evaluate integrals of the type

$$\int_T \frac{\partial \varphi_i}{\partial x_k} \frac{\partial \varphi_j}{\partial x_l} \, \mathrm{d}\mathbf{x}, \qquad i, j = 1, \dots, N,\ k, l = 1, \dots, d,\ T \in \mathscr{T}_h, \tag{1.57}$$

which appears, e.g., in the definition of the stiffness matrix.

The use of the approach from Section 1.1 requires an evaluation (and storing) of $\nabla \varphi_i|_T$ for all $i = 1, \dots, N$ and all $T \in \mathscr{T}_h$. Let us note, that $\nabla \varphi_i|_T = 0$ for a lot of $T \in \mathscr{T}_h$ since supports of test functions contain only a few elements from $\mathscr{T}_h$.

It is more efficient to employ the concept of the reference elements, namely relation (1.49), where the integration is carried out on the reference element $\widehat{T}$.

Therefore, using (1.49), we can evaluate (1.57) with the aid of

$$\int_T \frac{\partial \varphi_i}{\partial x_k} \frac{\partial \varphi_j}{\partial x_l} \, \mathrm{d}\mathbf{x} = \int_{\widehat{T}} \left(J_{F_T}^{-\mathrm{T}} \widehat{\nabla} \hat{\varphi}_{i,T} \right)_k \left(J_{F_T}^{-\mathrm{T}} \widehat{\nabla} \hat{\varphi}_{j,T} \right)_l |\det J_{F_T}| \, \mathrm{d}\hat{\mathbf{x}}. \tag{1.58}$$

Data structures

In order to evaluate the right-hand side of (1.58) for all $i, j = 1, \dots, N$, $k, l = 1, \dots, d$, $T \in \mathscr{T}_h$ it is enough to evaluate and store the following data.

(S1) for each $T \in \mathscr{T}_h$, the determinant of the Jacobi matrix $\det J_{F_T}$ and the transposed matrix of the inversion of the Jacobi matrix $J_{F_T}^{-\mathrm{T}}$,

(S2) for each $\hat{\varphi}_i$, $i = 1, \dots \hat{N}$, the gradient $\widehat{\nabla}\hat{\varphi}_i$ on $\widehat{T}$,

(S3) for each pair (φ_i, T), $T \in \mathscr{T}_h$, $T \subset \mathrm{supp}(\varphi_i)$, $i = 1, \dots, N$, an index $j \in \{1, \dots, \hat{N}\}$ such that $\varphi_i(F_T(\hat{\mathbf{x}}))|_T = \hat{\varphi}_j(\hat{\mathbf{x}})$.

Let us note that (S1) is given by the geometry of the mesh, (S2) by the reference shape functions and (S3) by a combination of both. However, in (S3) we have to store only one integer index for any acceptable pair (φ_i, T).

1.3 Basis and shape functions

In this section, we describe two possible ways (among others) of the construction of the basis of spaces $V_h^{t,p}$ and $V_h^{q,p}$ given by (1.51) and (1.54), respectively. In Section 1.3.1, we describe the construction of Lagrangian basis, which is easy for determination and also for implementation. However, in the case when we use, e.g., different degrees of polynomial approximations at different elements (*hp*-methods), the efficiency of this approach is low. Therefore, in Section 1.3.2 we present the construction of the Lobatto basis, which is generally more efficient.

We have already mentioned in Section 1.2.4 that we define the *reference shape functions* $\hat{B} := \{\hat{\varphi}_j\}_{j=1}^{\hat{N}}$ defined on the reference element $\widehat{T}$. Then the basis of V_h is generated by $\hat{B}$. We present the construction of the shape functions for $d = 2$. For $d = 3$, it is possible to use a similar (but technically more complicated) approach.

1.3.1 The reference Lagrangian shape functions

The reference Lagrangian shape functions are defined with the aid of a set of the reference Lagrangian nodes $\hat{\mathbf{z}}_i = (\hat{z}_{i,1}, \hat{z}_{i,2})$, $i = 1, \dots, \hat{N}$ within the reference element. Then we simply put $\hat{\varphi}_i(\hat{\mathbf{z}}_k) = \delta_{ij}$, $i, j = 1, \dots, \hat{N}$. In the following, we introduce the reference Lagrangian nodes and the reference Lagrangian shape function for the reference triangle and the reference square separately.

The reference Lagrangian shape functions on the square

Definition 1.2. *(Lagrangian nodes) Let $p \geq 1$, then the Lagrangian nodes on the reference square $\widehat{T}_q$ corresponding to the polynomial degree p are the set of nodes*

$$\{\hat{\mathbf{z}}_k^{q,p}\}_{k=1}^{\hat{N}^{q,p}}, \ \hat{\mathbf{z}}_k^{q,p} = (\hat{z}_{k,1}^{q,p}, \hat{z}_{k,2}^{q,p}) \tag{1.59}$$

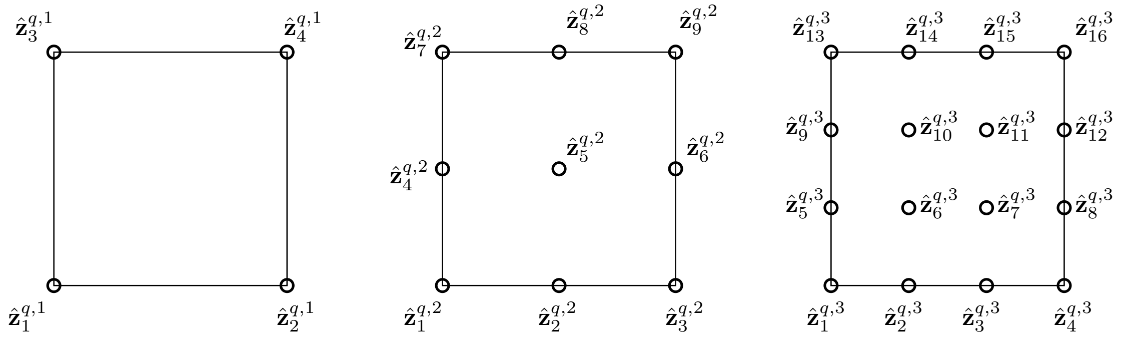

$\hat{\mathbf{z}}_3^{q,1}$
$\hat{\mathbf{z}}_4^{q,1}$
$\hat{\mathbf{z}}_1^{q,1}$
$\hat{\mathbf{z}}_2^{q,1}$
$\hat{\mathbf{z}}_7^{q,2}$
$\hat{\mathbf{z}}_8^{q,2}$
$\hat{\mathbf{z}}_9^{q,2}$
$\hat{\mathbf{z}}_4^{q,2}$
$\hat{\mathbf{z}}_5^{q,2}$
$\hat{\mathbf{z}}_6^{q,2}$
$\hat{\mathbf{z}}_1^{q,2}$
$\hat{\mathbf{z}}_2^{q,2}$
$\hat{\mathbf{z}}_3^{q,2}$
$\hat{\mathbf{z}}_{13}^{q,3}$
$\hat{\mathbf{z}}_{14}^{q,3}$
$\hat{\mathbf{z}}_{15}^{q,3}$
$\hat{\mathbf{z}}_{16}^{q,3}$
$\hat{\mathbf{z}}_9^{q,3}$
$\hat{\mathbf{z}}_{10}^{q,3}$
$\hat{\mathbf{z}}_{11}^{q,3}$
$\hat{\mathbf{z}}_{12}^{q,3}$
$\hat{\mathbf{z}}_5^{q,3}$
$\hat{\mathbf{z}}_6^{q,3}$
$\hat{\mathbf{z}}_7^{q,3}$
$\hat{\mathbf{z}}_8^{q,3}$
$\hat{\mathbf{z}}_1^{q,3}$
$\hat{\mathbf{z}}_2^{q,3}$
$\hat{\mathbf{z}}_3^{q,3}$
$\hat{\mathbf{z}}_4^{q,3}$

Figure 1.8: The reference Lagrangian nodes on the reference square $\widehat{T}_q$, $p = 1$ (left), $p = 2$ (center) and $p = 3$ (right)

where

$$\hat{z}^{q,p}_{j(p+1)+i+1,1} = i/p, \quad \hat{z}^{q,p}_{j(p+1)+i+1,2} = j/p, \qquad i, j = 0, \dots, p. \tag{1.60}$$

Obviously, $\hat{N}^{q,p} = (p+1)^2$.

Figure 1.8 shows the Lagrangian nodes for $p = 1, 2, 3$. We define the *reference Lagrangian shape functions* $\{\hat{\varphi}^{q,p}_l\}_{l=1}^{\hat{N}^{q,p}}$ for $p \geq 1$ on the reference square $\widehat{T}_q$ by the relation

$$\hat{\varphi}^{q,p}_l \in P^{q,p}(\widehat{T}_q),\ \hat{\varphi}^{q,p}_l(\hat{\mathbf{z}}^{q,p}_k) = \delta_{lk},\ l, k = 1, \dots, \hat{N}^{q,p}. \tag{1.61}$$

It is possible to derive the following explicit relations for the *Lagrangian shape functions on the reference square* $\widehat{T}_q$

$$\begin{array}{ll}
p = 1 & \hat{\varphi}^{q,1}_1 = (1-\lambda_1)(1-\lambda_2),\ \hat{\varphi}^{q,1}_2 = \lambda_1(1-\lambda_2), \\
 & \hat{\varphi}^{q,1}_3 = \lambda_2(1-\lambda_1),\ \hat{\varphi}^{q,1}_4 = \lambda_1\lambda_2 \\
p = 2 & \hat{\varphi}^{q,2}_1 = 4(1-\lambda_1)(\frac{1}{2}-\lambda_1)(1-\lambda_2)(\frac{1}{2}-\lambda_2), \\
 & \hat{\varphi}^{q,2}_2 = 8\lambda_1(1-\lambda_1)(1-\lambda_2)(\frac{1}{2}-\lambda_2), \\
 & \hat{\varphi}^{q,2}_3 = -4\lambda_1(\frac{1}{2}-\lambda_1)(1-\lambda_2)(\frac{1}{2}-\lambda_2), \\
 & \hat{\varphi}^{q,2}_4 = 8(1-\lambda_1)(\frac{1}{2}-\lambda_1)\lambda_2(1-\lambda_2), \\
 & \hat{\varphi}^{q,2}_5 = 16\lambda_1(1-\lambda_1)\lambda_2(1-\lambda_2), \\
 & \hat{\varphi}^{q,2}_6 = -8\lambda_1(\frac{1}{2}-\lambda_1)\lambda_2(1-\lambda_2), \\
 & \hat{\varphi}^{q,2}_7 = -4(1-\lambda_1)(\frac{1}{2}-\lambda_1)\lambda_2(\frac{1}{2}-\lambda_2), \\
 & \hat{\varphi}^{q,2}_8 = -8\lambda_1(1-\lambda_1)\lambda_2(\frac{1}{2}-\lambda_2), \\
 & \hat{\varphi}^{q,2}_9 = 4\lambda_1(\frac{1}{2}-\lambda_1)\lambda_2(\frac{1}{2}-\lambda_2),
\end{array} \tag{1.62}$$

where $\lambda_1 = \hat{x}_1$ and $\lambda_2 = \hat{x}_2$ are the barycentric coordinates on $\widehat{T}_q$.

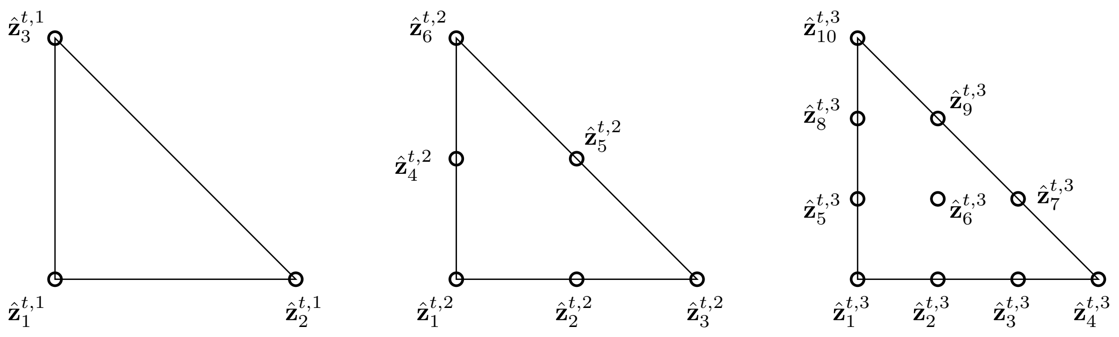

$\hat{\mathbf{z}}_3^{t,1}$
$\hat{\mathbf{z}}_1^{t,1}$
$\hat{\mathbf{z}}_2^{t,1}$
$\hat{\mathbf{z}}_6^{t,2}$
$\hat{\mathbf{z}}_5^{t,2}$
$\hat{\mathbf{z}}_4^{t,2}$
$\hat{\mathbf{z}}_1^{t,2}$
$\hat{\mathbf{z}}_2^{t,2}$
$\hat{\mathbf{z}}_3^{t,2}$
$\hat{\mathbf{z}}_{10}^{t,3}$
$\hat{\mathbf{z}}_9^{t,3}$
$\hat{\mathbf{z}}_8^{t,3}$
$\hat{\mathbf{z}}_7^{t,3}$
$\hat{\mathbf{z}}_5^{t,3}$
$\hat{\mathbf{z}}_6^{t,3}$
$\hat{\mathbf{z}}_1^{t,3}$
$\hat{\mathbf{z}}_2^{t,3}$
$\hat{\mathbf{z}}_3^{t,3}$
$\hat{\mathbf{z}}_4^{t,3}$

Figure 1.9: Lagrangian nodes on the reference triangle $\widehat{T}_t$, $p = 1$ (left), $p = 2$ (center) and $p = 3$ (right)

The reference Lagrangian shape functions on the triangle

Definition 1.3. *(Lagrangian nodes) Let $p \geq 1$, then the reference Lagrangian nodes on the reference triangle $\widehat{T}_t$ corresponding to the polynomial degree p are the set of nodes*

$$\{\hat{\mathbf{z}}_k^{t,p}\}_{k=1}^{\hat{N}^{t,p}},\ \hat{\mathbf{z}}_k^{t,p} = (\hat{z}_{k,1}^{t,p}, \hat{z}_{k,2}^{t,p}) \tag{1.63}$$

generated by the following algorithm

```
$k := 0$
 for $i = 0, \ldots, p$
    for $j = 0, \ldots, p - i$
      $k := k + 1$
      $\hat{z}_{k,1}^{t,p} = j/p,$
      $\hat{z}_{k,2}^{t,p} = i/p.$
```
(1.64)

Obviously, $\hat{N}^{t,p} = (p+1)(p+2)/2$.

Figure 1.9 shows the Lagrangian nodes for $p = 1, 2, 3$. We define the *reference Lagrangian shape functions* $\{\hat{\varphi}_l^{t,p}\}_{l=1}^{\hat{N}^{t,p}}$ by the relation

$$\hat{\varphi}_l^{t,p} \in P^{t,p}(\widehat{T}_t),\ \hat{\varphi}_l^{t,p}(\hat{\mathbf{z}}_k^{t,p}) = \delta_{lk},\ l, k = 1, \ldots, \hat{N}^{t,p}. \tag{1.65}$$

It is possible to derive the following explicit relation for the *Lagrangian shape functions* *on the reference triangle* $\widehat{T}_t$

$$p=1 \qquad \hat{\varphi}_1^{t,1}=\lambda_1,\ \hat{\varphi}_2^{t,1}=\lambda_2,\ \hat{\varphi}_3^{t,1}=\lambda_3, \tag{1.66}$$

$$\begin{aligned}
p=2 \qquad & \hat{\varphi}_2^{t,2}=\lambda_1(2\lambda_1-1),\ \hat{\varphi}_2^{t,2}=4\lambda_1\lambda_2,\ \hat{\varphi}_3^{t,2}=\lambda_2(2\lambda_2-1),\\
& \hat{\varphi}_4^{t,2}=4\lambda_2\lambda_3,\ \hat{\varphi}_5^{t,2}=\lambda_3(2\lambda_3-1),\ \hat{\varphi}_6^{t,2}=4\lambda_3\lambda_1,
\end{aligned}$$

$$\begin{aligned}
p=3 \qquad & \hat{\varphi}_1^{t,3}=\lambda_1(3\lambda_1-1)(3\lambda_1-2)/2,\ \hat{\varphi}_2^{t,3}=9\lambda_1\lambda_2(3\lambda_1-1)/2,\\
& \hat{\varphi}_3^{t,3}=9\lambda_1\lambda_2(3\lambda_2-1)/2,\ \hat{\varphi}_4^{t,3}=\lambda_2(3\lambda_2-1)(3\lambda_2-2)/2,\\
& \hat{\varphi}_5^{t,3}=9\lambda_3\lambda_1(3\lambda_1-2)/2,\ \hat{\varphi}_6^{t,3}=27\lambda_1\lambda_2\lambda_3,\\
& \hat{\varphi}_7^{t,3}=9\lambda_2\lambda_3(3\lambda_2-1)/2,\ \hat{\varphi}_8^{t,3}=9\lambda_3\lambda_1(3\lambda_2-1)/2,\\
& \hat{\varphi}_9^{t,3}=9\lambda_2\lambda_3(3\lambda_2-2)/2,\ \hat{\varphi}_{10}^{t,3}=\lambda_3(3\lambda_3-1)(3\lambda_3-2),
\end{aligned}$$

$$\begin{aligned}
p=4 \qquad & \hat{\varphi}_1^{t,4}=6\lambda_1(4\lambda_1-1)(4\lambda_1-2)(4\lambda_1-3),\\
& \hat{\varphi}_2^{t,4}=8\lambda_1\lambda_2(4\lambda_1-1)(4\lambda_1-2)/3,\\
& \hat{\varphi}_3^{t,4}=4\lambda_1\lambda_2(4\lambda_1-1)(4\lambda_2-1),\\
& \hat{\varphi}_4^{t,4}=8\lambda_1\lambda_2(4\lambda_2-1)(4\lambda_2-2)/3,\\
& \hat{\varphi}_5^{t,4}=6\lambda_2(4\lambda_2-1)(4\lambda_2-2)(4\lambda_2-3),\\
& \hat{\varphi}_6^{t,4}=8\lambda_3\lambda_1(4\lambda_1-1)(4\lambda_3-2)/3,\\
& \hat{\varphi}_7^{t,4}=32\lambda_1\lambda_2\lambda_3(4\lambda_1-1),\\
& \hat{\varphi}_8^{t,4}=32\lambda_1\lambda_2\lambda_3(4\lambda_2-1),\\
& \hat{\varphi}_9^{t,4}=8\lambda_2\lambda_3(4\lambda_2-1)(4\lambda_2-2)/3,\\
& \hat{\varphi}_{10}^{t,4}=4\lambda_3\lambda_1(4\lambda_3-1)(4\lambda_1-1),\\
& \hat{\varphi}_{11}^{t,4}=32\lambda_1\lambda_2\lambda_3(4\lambda_3-1),\\
& \hat{\varphi}_{12}^{t,4}=4\lambda_2\lambda_3(4\lambda_2-1)(4\lambda_3-1),\\
& \hat{\varphi}_{13}^{t,4}=8\lambda_3\lambda_1(4\lambda_3-1)(4\lambda_3-2)/3,\\
& \hat{\varphi}_{14}^{t,4}=8\lambda_2\lambda_3(4\lambda_3-1)(4\lambda_3-2)/3,\\
& \hat{\varphi}_{15}^{t,4}=6\lambda_3(4\lambda_3-1)(4\lambda_3-2)(4\lambda_3-3).
\end{aligned}$$

Here $\lambda_1=\hat{x}_1$, $\lambda_2=\hat{x}_2$ and $\lambda_3=1-\hat{x}_1-\hat{x}_2$ are the barycentric coordinates on $\widehat{T}_t$.

1.3.2 The reference Lobatto shape functions

The disadvantage of the Lagrangian shape functions is that they are not hierarchical, which means that

$$\{\hat{\varphi}_k^{q,p}\}_{k=1}^{\hat{N}^{q,p}} \not\subset \{\hat{\varphi}_k^{q,p+1}\}_{k=1}^{\hat{N}^{q,p+1}} \quad \text{and} \quad \{\hat{\varphi}_k^{t,p}\}_{k=1}^{\hat{N}^{t,p}} \not\subset \{\hat{\varphi}_k^{t,p+1}\}_{k=1}^{\hat{N}^{t,p+1}}, \quad p=1,2,\dots. \tag{1.67}$$

This causes no trouble in the case when our aim is to implement FEM for one given degree of polynomial approximation only. In the case, when we need to implement several degrees of polynomial approximation (e.g., *hp*-FEM), then the use of the Lagrangian shape functions requires an evaluation and storing data structure (S2) for a lot of test functions.

Therefore, it is more efficient to use, e.g., *Lobatto shape functions*, which are hierarchical, since they satisfy the conditions

$$\{\hat{\varphi}_k^{q,p}\}_{k=1}^{\hat{N}^{q,p}} \subset \{\hat{\varphi}_k^{q,p+1}\}_{k=1}^{\hat{N}^{q,p+1}} \quad \text{and} \quad \{\hat{\varphi}_k^{t,p}\}_{k=1}^{\hat{N}^{t,p}} \subset \{\hat{\varphi}_k^{t,p+1}\}_{k=1}^{\hat{N}^{t,p+1}}, \quad p = 1, 2, \ldots, \tag{1.68}$$

for quadrilaterals and triangles, respectively.

Now, we recall two well-known sets of functions.

Definition 1.4. *(Legendre polynomials) Legendre polynomials $L_n(x),\ n = 0, 1, \ldots$ are eigenvectors of the Legendre operator,*

$$-\frac{\mathrm{d}}{\mathrm{d}x}\left[(1-x^2)\frac{\mathrm{d}}{\mathrm{d}x}L_n(x)\right] = n(n+1)L_n, \quad x \in (-1, 1). \tag{1.69}$$

Legendre polynomials can be evaluated using the recursive formula

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

Legendre polynomials form an orthogonal basis of the space $L^2(-1, 1)$, namely

$$\int_{-1}^{1} L_k(x)L_m(x)\,\mathrm{d}x = \begin{cases} \frac{2}{2k+1} & \text{for } k = m, \\ 0 & \text{otherwise.} \end{cases} \tag{1.70}$$

Definition 1.5. *(Lobatto functions on $(-1, 1)$) Let us define functions*

$$\begin{aligned} \tilde{\ell}_0(x) &= \frac{1-x}{2}, \\ \tilde{\ell}_1(x) &= \frac{x+1}{2}, \\ \tilde{\ell}_k(x) &= \frac{1}{\|L_{k-1}\|_2}\int_{-1}^{x} L_{k-1}(\xi)\mathrm{d}\xi, \quad k = 2, 3, \ldots. \end{aligned} \tag{1.71}$$

From (1.70) we have $\|L_{k-1}\|_2 = \sqrt{2/(2k-1)}$. Obviously $\tilde{\ell}_k(-1) = 0$ and $\tilde{\ell}_k(1) = 0$ for $k \geq 2$ since $L_{k-1},\ k \geq 2$ are orthogonal to $L_0 = 1$, i.e.,

$$\int_{-1}^{1} L_k(x)\ \mathrm{d}x = \int_{-1}^{1} L_k(x)L_0(x)\ \mathrm{d}x = 0, \quad k = 2, 3, \ldots. \tag{1.72}$$

Therefore, the Lobatto functions $\tilde{\ell}_k,\ k = 0, \ldots, p$ form a basis of the space $P_p(-1, 1)$.

For the purposes of our reference elements, we have to transform the Lobatto functions from the interval $(-1, 1)$ onto $(0, 1)$.

Definition 1.6. *(Lobatto functions on* $(0,1)$*) Let us define the functions*

$$\ell_k(x) \;=\; \tilde{\ell}_k\left(\frac{x+1}{2}\right), \qquad k=0,1,\dots, \tag{1.73}$$

where $\tilde{\ell}_k,\ k=0,1,\dots,$ *are given by (1.71).*

Obviously $\ell_k(0)=0$, $\ell_k(1)=0$ for $k\geq 2$ and the Lobatto functions $\ell_k,\ k=0,\dots,p$ form a basis of the space $P_p(0,1)$.

For the definition of the shape function on triangles, it is convenient to decompose the higher-order Lobatto functions $\ell_k,\ k\geq 2$ from Definition 1.6 into products of the form

$$\ell_k(x)=\ell_0(x)\ell_1(x)\varphi_{k-2}(x), \quad k=2,3,\dots, \tag{1.74}$$

where $\varphi_{k-2}\in P^{k-2}([0,1]),\ k=2,3,\dots$ are the *kernel functions.*

Let us recall that we denote by $\{\hat{\mathbf{v}}_i\}$ the vertices and by $\{\hat{e}_i\}$ the edges of the reference element $\widehat{T}$.

The reference Lobatto shape functions on the square

The Lobatto shape functions on the quadrilateral $\widehat{T}_q$ are constructed as tensor products of the one dimensional Lobatto functions. They are grouped into 3 subsets according to which node they belong.

The *Lobatto vertex functions* $\hat{\varphi}^{q,p}_{\hat{\mathbf{v}}_l},\ l=1,\dots,4$ assigned to vertices $\hat{\mathbf{v}}_l,\ l=1,\dots,4$ are equal to one at $\hat{\mathbf{v}}_l$ and vanishing at all remaining vertices. They are chosen bilinear, defined via products of Lobatto functions as follows

$$\begin{aligned}
\hat{\varphi}^{q,p}_{\hat{\mathbf{v}}_1}(\hat{x}_1,\hat{x}_2) &= \ell_0(\hat{x}_1)\ell_0(\hat{x}_2),\\
\hat{\varphi}^{q,p}_{\hat{\mathbf{v}}_2}(\hat{x}_1,\hat{x}_2) &= \ell_1(\hat{x}_1)\ell_0(\hat{x}_2),\\
\hat{\varphi}^{q,p}_{\hat{\mathbf{v}}_3}(\hat{x}_1,\hat{x}_2) &= \ell_1(\hat{x}_1)\ell_1(\hat{x}_2),\\
\hat{\varphi}^{q,p}_{\hat{\mathbf{v}}_4}(\hat{x}_1,\hat{x}_2) &= \ell_0(\hat{x}_1)\ell_1(\hat{x}_2),
\end{aligned} \tag{1.75}$$

for $(\hat{x}_1,\hat{x}_2)\in\widehat{T}_q$.

For $p\geq 2$, the *Lobatto edge functions* $\hat{\varphi}^{q,p}_{\hat{e}_j,k},\ k=2,\dots,p,\ j=1,\dots,4$ are associated with the corresponding edge $\hat{e}_j$. Their trace on $\hat{e}_j$ coincides with the Lobatto functions ℓ_k and their trace vanishes on all remaining edges:

$$\begin{aligned}
\hat{\varphi}^{q,p}_{\hat{e}_1,k}(\hat{x}_1,\hat{x}_2) &= \ell_k(\hat{x}_1)\ell_0(\hat{x}_2), \quad 2\leq k\leq p,\\
\hat{\varphi}^{q,p}_{\hat{e}_2,k}(\hat{x}_1,\hat{x}_2) &= \ell_1(\hat{x}_1)\ell_k(\hat{x}_2), \quad 2\leq k\leq p,\\
\hat{\varphi}^{q,p}_{\hat{e}_3,k}(\hat{x}_1,\hat{x}_2) &= \ell_k(\hat{x}_1)\ell_1(\hat{x}_2), \quad 2\leq k\leq p,\\
\hat{\varphi}^{q,p}_{\hat{e}_4,k}(\hat{x}_1,\hat{x}_2) &= \ell_0(\hat{x}_1)\ell_k(\hat{x}_2), \quad 2\leq k\leq p,
\end{aligned} \tag{1.76}$$

for $(\hat{x}_1,\hat{x}_2)\in\widehat{T}_q$. Therefore, $p-2$ shape functions correspond to each edge of $\widehat{T}_q$.

For $p\geq 3$, the shape functions are completed by *Lobatto bubble functions* $\hat{\varphi}^{q,p}_{b,n_1,n_2,}$ which vanish everywhere on the boundary of $\widehat{T}_q$

$$\hat{\varphi}^{q,p}_{b,n_1,n_2,}(\hat{x}_1,\hat{x}_2)=\ell_{n_1}(\hat{x}_1)\ell_{n_2}(\hat{x}_2), \qquad 2\leq n_1\leq p,\ 2\leq n_2\leq p, \tag{1.77}$$

for $(\hat{x}_1,\hat{x}_2)\in\widehat{T}_q$. The total number of the Lobatto shape functions on $\widehat{T}_q$ is summarized in Table 1.1.

Node type	Pol. degree	# of shape functions	# of objects
Vertex	$p \geq 1$	1	4
Edge	$p \geq 2$	$p-1$	4
Bubble	$p \geq 3$	$(p-1)^2$	1

Table 1.1: Lobatto shape functions on $\widehat{T}_q$

.

The reference Lobatto shape functions on the triangle

The Lobatto basis of the space V_h consists again of vertex, edge and bubble shape functions defined on $\widehat{T}_t$.

The *Lobatto vertex functions* $\hat{\varphi}^{t,p}_{\hat{\mathbf{v}}_1}$, $\hat{\varphi}^{t,p}_{\hat{\mathbf{v}}_2}$, $\hat{\varphi}^{t,p}_{\hat{\mathbf{v}}_3}$, are assigned to vertices $\hat{\mathbf{v}}_1$, $\hat{\mathbf{v}}_2$, $\hat{\mathbf{v}}_3$. Each function is equal to one at the corresponding vertex and vanishes at the remaining two vertices. Vertex functions are linear, defined by following formulas

$$\hat{\varphi}^{t,p}_{\hat{\mathbf{v}}_k}(\hat{x}_1, \hat{x}_2) = \lambda_k, \qquad k = 1, 2, 3, \tag{1.78}$$

where $\lambda_1 = \hat{x}_1$, $\lambda_2 = \hat{x}_2$ and $\lambda_3 = 1 - \hat{x}_1 - \hat{x}_2$ are the barycentric coordinates on $\widehat{T}_t$.

For $p \geq 2$, the *Lobatto edge functions* $\hat{\varphi}^{t,p}_{\hat{e}_j,k}$, $k = 2, \dots, p$, $j = 1, 2, 3$ coincide with one-dimensional Lobatto functions on corresponding edges and vanish on all remaining edges. They can be written using the kernel functions φ_{k-2}, defined by (1.74), in the form

$$\begin{aligned} \hat{\varphi}^{t,p}_{\hat{e}_1,k}(\hat{x}_1, \hat{x}_2) &= \lambda_1 \lambda_2 \varphi_{k-2}(\lambda_2 - \lambda_1), \quad 2 \leq k \leq p, \\ \hat{\varphi}^{t,p}_{\hat{e}_2,k}(\hat{x}_1, \hat{x}_2) &= \lambda_2 \lambda_3 \varphi_{k-2}(\lambda_3 - \lambda_2), \quad 2 \leq k \leq p, \\ \hat{\varphi}^{t,p}_{\hat{e}_3,k}(\hat{x}_1, \hat{x}_2) &= \lambda_3 \lambda_1 \varphi_{k-2}(\lambda_1 - \lambda_3), \quad 2 \leq k \leq p. \end{aligned} \tag{1.79}$$

For $p \geq 3$, the *Lobatto bubble shape functions* $\hat{\varphi}^{t,p}_{b,n1,n_2}$ complete the basis on $\widehat{T}_t$. These functions vanish on the whole element boundary. They will be defined using affine coordinates and kernel functions as follows

$$\hat{\varphi}^{t,p}_{b,n1,n_2} = \lambda_1 \lambda_2 \lambda_3 \varphi_{n_1-1}(\lambda_3 - \lambda_2) \varphi_{n_2-1}(\lambda_2 - \lambda_1), \quad 1 \leq n_1, 1 \leq n_2,\ n_1 + n_2 \leq p-1, \tag{1.80}$$

where φ_{n_1-1}, φ_{n_2-1} are the kernel functions defined by (1.74). The total number of Lobatto shape functions on a triangular reference element is summarized in Table 1.2.

Remark 1.2. *On both reference domains $\widehat{T}_q$, $\widehat{T}_t$ shape functions coincide on edges with Lobatto functions ℓ_k, $k = 0, 1, \dots$. Moreover, the bubble shape functions do not affect a possible compatibility between quadrilateral and triangular meshes. This allows us to combine triangular and quadrilateral elements in one hybrid mesh, see [1], [3].*

Node type	Pol. degree	# of shape functions	# of objects
Vertex	$p \geq 1$	1	3
Edge	$p \geq 2$	$p-1$	3
Bubble	$p \geq 3$	$(p-1)(p-2)/2$	1

Table 1.2: Lobatto shape functions on $\widehat{T}_t$

.

1.3.3 Global basis functions on V_h

In Sections 1.3.1 – 1.3.2 we introduced two possible sets of shape functions defined on the reference element $\widehat{T}$.

Here we describe the construction of a global basis of the finite-dimensional space V_h, which is carried out by gluing together reference shape functions (transformed to the physical element by the mapping F_T). This is done in such a way, that the resulting basis functions satisfy conformity requirements of the space $V_h \subset H^1(\Omega)$.

Remark 1.3. *In order to fulfill the conformity requirement of global continuity of basis functions, the orientation of physical mesh edges has to be taken into account. For two-dimensional problems, it is possible to index edges of the reference element as well as physical elements counterclockwise. Then the orientation of the reference element edge $\hat{e} = \hat{\mathbf{v}}_i\hat{\mathbf{v}}_j \subset \partial\widehat{T}$ coincides with the orientation of the edge $e = \mathbf{v}_i\mathbf{v}_j \subset \partial T$ such that $\mathbf{v}_i = F_T(\hat{\mathbf{v}}_i)$ and $\mathbf{v}_j = F_T(\hat{\mathbf{v}}_j)$. In other case, when the orientations differ, all reference edge functions have to be transformed by a suitable mapping, which inverts the parameterization of the edge $\hat{e}$.*

In order to avoid a complication with a notation, we assume that $\mathscr{T}_h$ is a triangular grid. All statements are valid also for quadrilateral grid, only the superscript ^{t} has to be replaced by the superscript ^{q} and the number of vertices and edges of $T \in \mathscr{T}_h$ has to be increased from 3 to 4.

Global Lagrangian shape functions

Let $p \geq 1$ be a given degree of polynomial approximation and $\hat{\mathbf{z}}_i^{t,p},\ i = 1,\dots,\hat{N}$ be the reference Lagrangian nodes defined by (1.64). For each $T \in \mathscr{T}_h$, we define the nodes

$$\mathbf{z}_{T,i}^{t,p} = F_T(\hat{\mathbf{z}}_i^{t,p}),\ i = 1,\dots,\hat{N}, \tag{1.81}$$

where F_T maps $\widehat{T}_t$ onto T. Let us note that some of these nodes coincide, namely $\mathbf{z}_{T,i}^{t,p} = \mathbf{z}_{T',i'}^{t,p}$ for $T \neq T'$ sharing a vertex or an edge.

Moreover, we define the set of *Lagrangian nodes*

$$\mathscr{L}^{t,p} := \left\{ \mathbf{z} \in \Omega \text{ such that } \exists (T,i) \in \mathscr{T}_h \times \{1,\dots,\hat{N}\} \text{ satisfying } \mathbf{z} = \mathbf{z}_{T,i}^{t,p} \right\}. \tag{1.82}$$

This means that $\mathscr{L}^{t,p}$ contains all $\mathbf{z}_{T,i}^{t,p},\ i = 1,\dots,\hat{N},\ T \in \mathscr{T}_h$ lying inside of Ω (we exclude nodes lying on $\partial\Omega$) and each node appears in $\mathscr{L}^{t,p}$ only one time. We index the set $\mathscr{L}^{t,p}$ by $\mathscr{L}^{t,p} = \{\mathbf{z}_j^{t,p},\ j = 1,\dots,N_L\}$.

The *global Lagrangian basis functions* $\varphi_i^{t,p},\ i = 1, \ldots, N_L$ are defined by

$$\varphi_i^{t,p}(\mathbf{x}) \in V_h,\ \varphi_i^{t,p}(\mathbf{z}_j^{t,p}) = \delta_{ij}, \quad i, j = 1, \ldots, N_L. \tag{1.83}$$

Let us note that each global Lagrangian basis function $\varphi_i^{t,p},\ i = 1, \ldots, N_L$ can be, due to (1.83), classified as

- *vertex basis function* if the corresponding Lagrangian node is a vertex of $\mathscr{T}_h$, its support is formed by a patch of elements $T \in \mathscr{T}_h$ having this vertex in common,
- *edge basis function* if the corresponding Lagrangian node lies on an edge of $\mathscr{T}_h$, its support is formed by a patch of two elements $T \in \mathscr{T}_h$ sharing this edge,
- *bubble basis function* if the corresponding Lagrangian node lies in the interior of $T \in \mathscr{T}_h$, which is also its support.

The global Lagrangian basis functions are defined in the following way.

Definition 1.7. *(Vertex basis function) Let $\mathbf{z}_i^{t,p}$ be a Lagrangian node which is identified with the vertex $\mathbf{v}_l$ of $\mathscr{T}_h$. Then the corresponding vertex Lagrangian basis function $\varphi_i^{t,p}$ is a continuous function defined on Ω which equals to one at $\mathbf{v}_l$ and vanishes outside of the patch S_l formed by all elements $T \in \mathscr{T}_h$ sharing the vertex $\mathbf{v}_l$. For each element $T_j \in S_l$,*

$$\varphi_i^{t,p}(\mathbf{x})|_{T_j} = \hat{\varphi}_k^{t,p}(\hat{\mathbf{x}}), \qquad \mathbf{x} = F_T(\hat{\mathbf{x}}), \tag{1.84}$$

where $\hat{\varphi}_k^{t,p}$ is the reference Lagrangian shape function such that $\hat{\mathbf{z}}_k = F_T^{-1}(\mathbf{z}_i^{t,p})$.

Definition 1.8. *(Edge basis function) Let $\mathbf{z}_i^{t,p}$ be a Lagrangian node which lies on an edge e_l of $\mathscr{T}_h$. Then the corresponding edge Lagrangian basis function $\varphi_i^{t,p}$ is a continuous function defined on Ω which vanishes outside of the patch S_l formed by all elements sharing the edge e_l. For each element $T_j \in S_l$,*

$$\varphi_i^{t,p}(\mathbf{x})|_{T_j} = \hat{\varphi}_k^{t,p}(\hat{\mathbf{x}}), \qquad \mathbf{x} = F_T(\hat{\mathbf{x}}), \tag{1.85}$$

where $\hat{\varphi}_k^{t,p}$ is the reference Lagrangian shape function such that $\hat{\mathbf{z}}_k = F_T^{-1}(\mathbf{z}_i^{t,p})$.

Definition 1.9. *(Bubble basis function) Let $\mathbf{z}_i^{t,p}$ be a Lagrangian node which lies in the interior of an element $T \in \mathscr{T}_h$. Then the corresponding bubble Lagrangian basis function $\varphi_i^{t,p}$ is a continuous function defined on Ω which vanishes outside of T and*

$$\varphi_i^{t,p}(\mathbf{x})|_{T} = \hat{\varphi}_k^{t,p}(\hat{\mathbf{x}}), \qquad \mathbf{x} = F_T(\hat{\mathbf{x}}), \tag{1.86}$$

where $\hat{\varphi}_k^{t,p}$ is the reference Lagrangian shape function such that $\hat{\mathbf{z}}_k = F_T^{-1}(\mathbf{z}_i^{t,p})$.

Global Lobatto shape functions

Let $p \geq 1$ be a given degree of polynomial approximation and the global basis B of V_h consists of three types of basis functions:

- *Vertex basis functions* $\varphi^{t,p}_{\mathbf{v}_l}$, $l = 1,\dots,N$, which are associated to each inner vertex $\mathbf{v}_l$, $l = 1,\dots,N$ of $\mathscr{T}_h$. The support of $\varphi^{t,p}_{\mathbf{v}_l}$ is formed by a patch of elements $T \in \mathscr{T}_h$ having $\mathbf{v}_l$ as a vertex.

- *Edge basis functions* $\varphi^{t,p}_{e_l,k}$, $2 \le k \le p$, $l = 1,\dots,E$, which are associated to each inner edge e_l, $l = 1,\dots,E$ of $\mathscr{T}_h$. The support of $\varphi^{t,p}_{e_l,k}$, $2 \le k \le p$ is formed by the patch of two elements from $T \in \mathscr{T}_h$ sharing the face e_l.

- *Bubble basis functions* $\varphi^{t,p}_{T,n1,n_2}$, $1 \le n_1$, $1 \le n_2$, $n_1 + n_2 \le p-1$, $T \in \mathscr{T}_h$, which are associated to each element $T \in \mathscr{T}_h$. Their support consists only of the one element T.

The global Lobatto basis functions are defined in the following way.

Definition 1.10. *(Vertex basis function) The vertex basis function $\varphi^{t,p}_{\mathbf{v}_l}$ associated with a mesh vertex $\mathbf{v}_l$, $l = 1,\dots,N$ is a continuous function defined on Ω which equals to one at $\mathbf{v}_l$ and vanishes outside of the patch S_l formed by all elements $T \in \mathscr{T}_h$ sharing the vertex $\mathbf{v}_l$. For each element $T_i \in S_l$,*

$$\varphi^{t,p}_{\mathbf{v}_l}(\mathbf{x})|_{T_i} = \hat{\varphi}^{t,p}_{\hat{\mathbf{v}}_k}(\hat{\mathbf{x}}), \qquad \mathbf{x} = F_T(\hat{\mathbf{x}}), \tag{1.87}$$

where $\hat{\varphi}^{t,p}_{\hat{\mathbf{v}}_k}$ is the reference vertex shape function associated to the vertex $\hat{\mathbf{v}}_k$ of $\widehat{T}_t$ such that $\hat{\mathbf{v}}_k = F_T^{-1}(\mathbf{v}_l)$.

Definition 1.11. *(Edge basis function) The edge basis function $\varphi^{t,p}_{e_l,k}$ associated with a mesh edge e_l and $k = 2,\dots,p$, is a continuous function defined on Ω, which vanishes outside of the patch S_l formed by all elements sharing the edge e_l. For each element $T \in S_l$,*

$$\varphi^{t,p}_{e_l,k}(\mathbf{x})|_{T_i} = \hat{\varphi}^{t,p}_{\hat{e}_j,k}(\hat{\mathbf{x}}), \qquad \mathbf{x} = F_T(\hat{\mathbf{x}}), \tag{1.88}$$

where $\hat{\varphi}^{t,p}_{\hat{e}_j,k}$ is the reference edge shape function associated to the edge $\hat{e}_j$ of $\widehat{T}_t$ such that $\hat{e}_j = F_T^{-1}(e_l)$.

Definition 1.12. *(Bubble basis function) The bubble basis function $\varphi^{t,p}_{T,n_1,n_2}$ associated with $T \in \mathscr{T}_h$ and $1 \le n_1$, $1 \le n_2$, $n_1 + n_2 \le p-1$, $k = 2,\dots,p$, is a continuous function defined on Ω which vanishes outside of T,*

$$\varphi^{t,p}_{T,n_1,n_2}(\mathbf{x})|_{T} = \hat{\varphi}^{t,p}_{b,n_1,n_2}(\hat{\mathbf{x}}), \qquad \mathbf{x} = F_T(\hat{\mathbf{x}}), \tag{1.89}$$

where $\hat{\varphi}^{t,p}_{b,n_1,n_2}$ is the reference bubble shape function associated to $\widehat{T}_t$.

Remark 1.4. *We already mentioned that the use of hierarchical finite elements is advantageous in the case where different degrees of polynomial approximation are used on different elements. However, in the case of conforming finite element methods, the basis functions have to be continuous. Therefore, it is necessary to construct a transition from one degree of polynomial approximation to the another, see [1].*

1.4 Discontinuous finite elements

In the previous sections of this chapter, we discussed implementations of conforming finite elements methods, where the finite element spaces consist of continuous functions. In this section we focus on an implementation of methods based on discontinuous approximations, namely the *discontinuous Galerkin (DG) method*.

Although the DG formulation is more complicated than for the conforming FEM since additional terms appear there, its implementation is usually simpler since the test functions are chosen separately for each $T \in \mathscr{T}_h$.

1.4.1 DG discretization of the model problem

In the following we recall the DG discretization of the model problem (1.1) – (1.2). We consider here the IIPG variant of the DG method since it has the simplest formulation.

Let $\mathscr{T}_h$ ($h > 0$) be a partition of the closure $\overline{\Omega}$ of the domain Ω into a finite number of closed d-dimensional elements, where we admit a combination of triangular and quadrilateral elements including hanging nodes. For simplicity, we write again,

$$\overline{\Omega} = \bigcup_{T \in \mathscr{T}_h} T. \tag{1.90}$$

Moreover, $\mathscr{F}_h$ denotes the set of all edges (faces) of $\mathscr{T}_h$.

We assume that for each element $T \in \mathscr{T}_h$ there exists a mapping

$$F_T = F_T(\hat{\mathbf{x}}) = (F_{T,1}(\hat{\mathbf{x}}), \dots, F_{T,d}(\hat{\mathbf{x}})) : \widehat{T} \to \mathbb{R}^d \quad \text{such that} \quad F_T(\widehat{T}) = T, \tag{1.91}$$

where $\widehat{T}$ is either the reference triangle $\widehat{T}_t$ or the reference quadrilateral $\widehat{T}_q$ given by (1.34) or (1.35), respectively.

We define the space of *discontinuous piecewise polynomial functions* S_{hp} by

$$S_{hp} = \{v_h,\ v_h \in L^2(\Omega),\ v_h|_T \circ F_T \in P^p(\widehat{T})\ \forall T \in \mathscr{T}_h\}, \tag{1.92}$$

where $\widehat{T}$ denotes either $\widehat{T}_t$ or $\widehat{T}_q$ depending on the type of T. Furthermore, P^p is the space of polynomials of degree $\leq p$ on $\widehat{T}$ defined by (1.51) and/or (1.54).

Since DG methods use a discontinuous approximation, it is possible to define the space $P^p(\widehat{T})$ in (1.92) in several manners, namely

- $P^{t,p}(\widehat{T}_t)$ for triangles and $P^{t,p}(\widehat{T}_q)$ for quadrilaterals,
- $P^{q,p}(\widehat{T}_t)$ for triangles and $P^{q,p}(\widehat{T}_q)$ for quadrilaterals,
- $P^{t,p}(\widehat{T}_t)$ for triangles and $P^{q,p}(\widehat{T}_q)$ for quadrilaterals,

In this text we consider the first variant, the others can be obtain with only a small modification.

Moreover, $\boldsymbol{n}_e$ denotes a unit normal vector to $e \in \mathscr{F}_h$, its orientation is arbitrary but fixed. Furthermore, the symbols $\langle v_h \rangle_e$ and $[|v_h|]_{Ee}$ denote the mean value and the jump in direction of $\boldsymbol{n}_e$ of $v_h \in S_{hp}$ on inner face $e \in \mathscr{F}_h$ ($e \subset \Omega$). For $e \subset \partial\Omega$, we put $[|v_h|]_{Ee} = \langle v_h \rangle_e = v_h|_e$. Finally, if $\boldsymbol{n}_e$ or $[| \cdot |]_{Ee}$ or $\langle \cdot \rangle_e$ are arguments of $\int_e \dots \, \mathrm{d}\sigma$, $e \in \mathscr{F}_h$, we omit very often the subscript e and write simply $\boldsymbol{n}$ or $[| \cdot |]_E$ or $\langle \cdot \rangle$, respectively.

Now, we are ready to introduce the DG approximation of the model problem.

Definition 1.13. *We say that $u_h \in S_{hp}$ is the* approximate DG solution *of (1.1) – (1.2) if*

$$\sum_{T \in \mathscr{T}_h} \int_T \nabla u_h \cdot \nabla v_h \, \mathrm{d}x - \sum_{e \in \mathscr{F}_h} \int_e \langle \nabla u_h \rangle \cdot \boldsymbol{n} [|v_h|]_E \, \mathrm{d}\sigma \tag{1.93}$$
$$+ \sum_{e \in \mathscr{F}_h} \int_e \rho [|u_h|]_E [|v_h|]_E \, \mathrm{d}\sigma = \int_\Omega g \, v_h \, \mathrm{d}x \quad \forall v_h \in S_{hp},$$

where ρ denotes the penalty parameter.

1.4.2 The *hp*-discontinuous Galerkin method

Let us assume that the elements of $\mathscr{T}_h$ are indexed by

$$\mathscr{T}_h = \{T_\mu, \ \mu = 1, \dots, M\}. \tag{1.94}$$

DGM allows the use of different polynomial degrees over different elements without the necessity to construct a cross-edge transition from one degree of polynomial approximation to the another. In this section, we will consider this general case since its treatment is relatively simple. We assign a positive integer p_μ (local polynomial degree) to each $T_\mu \in \mathscr{T}_h$. In the case when one uses the same degree of polynomial approximation in the whole Ω, we have $p_\mu = p \ \forall T_\mu \in \mathscr{T}_h$. We define the set

$$\mathsf{p} = \{p_\mu, T_\mu \in \mathscr{T}_h\}. \tag{1.95}$$

Over the triangulation $\mathscr{T}_h$, we define the finite dimensional space of discontinuous piecewise polynomial functions associated with the vector p by

$$S_{h\mathsf{p}} = \{v; \ v \in L^2(\Omega), \ v|_{T_\mu} \circ F_K \in P_{p_\mu}(\widehat{T}) \ \forall \mu = 1, \dots, M\}, \tag{1.96}$$

where $P^{p_\mu}(\widehat{T})$ denotes the space of all polynomials on $\widehat{T}$ of degree $\leq p_\mu$.

Now we can simply formulate the *hp*-approximation of the model problem.

Definition 1.14. *We say that $u_h \in S_{h\mathsf{p}}$ is the hp-*approximate DG solution *of (1.1) – (1.2) if*

$$\sum_{T \in \mathscr{T}_h} \int_T \nabla u_h \cdot \nabla v_h \, \mathrm{d}x - \sum_{e \in \mathscr{F}_h} \int_e \langle \nabla u_h \rangle \cdot \boldsymbol{n} [|v_h|]_E \, \mathrm{d}\sigma \tag{1.97}$$
$$+ \sum_{e \in \mathscr{F}_h} \int_e \rho [|u_h|]_E [|v_h|]_E \, \mathrm{d}\sigma = \int_\Omega g \, v_h \, \mathrm{d}x \quad \forall v_h \in S_{h\mathsf{p}},$$

where ρ denotes the penalty parameter.

p_μ	1	2	3	4	5
$d = 2$	3	6	10	15	21
$d = 3$	4	10	20	35	56

Table 1.3: Values of dof_μ for $p_\mu = 1, \ldots, 5$ and $d = 2, 3$

In comparison with (1.4), the face integrals have to be implemented. On the other hand, the implementation of DGM is easier since a discontinuous approximation is used. Therefore, it is possible to construct basis functions with a support consisting of one $T \in \mathscr{T}_h$.

In the following we describe the construction of basis functions of S_{hp}. First, we introduce a (global) basis of the space $S_{h\mathsf{p}}$ as a composition of local shapes constructed for each $T \in \mathscr{T}_h$ separately. Then, we introduce the construction of local shape functions. Finally, we explain the evaluation of the face integrals.

1.4.3 Definition of the DG basis

Since $S_{h\mathsf{p}}$ is the space of discontinuous piecewise polynomial functions, it is possible to consider a set of linearly independent polynomial functions on T_μ for each $T_\mu \in \mathscr{T}_h$

$$\begin{aligned} B_\mu \;=\; \Big\{ & \varphi_{\mu,j};\ \varphi_{\mu,j} \in S_{h\mathsf{p}},\ \mathrm{supp}(\varphi_{\mu,j}) \subseteq T_\mu, \\ & \varphi_{\mu,j} \text{ are linearly independent for } j = 1, \ldots, \mathsf{dof}_\mu \Big\}, \end{aligned} \tag{1.98}$$

where

$$\mathsf{dof}_\mu = \begin{cases} \frac{1}{2}\prod_{j=1}^{2}(p_\mu + j) & \text{for } d = 2 \\ \\ \frac{1}{6}\prod_{j=1}^{3}(p_\mu + j) & \text{for } d = 3 \end{cases}, \quad \mu = 1, \ldots, M, \tag{1.99}$$

denotes the number of *local degrees of freedom* for each element $T_\mu \in \mathscr{T}_h$. The values of dof_μ are shown in Table 1.3 for $p_\mu = 1, \ldots, 5$ and $d = 2, 3$. We call B_μ the *local basis* on T_μ. For the construction of the basis B_μ, $\mu = 1, \ldots, M$, see Section 1.4.4.

A composition of the local bases B_μ, $\mu = 1, \ldots, M$ defines a basis of $S_{h\mathsf{p}}$, i.e.,

$$B = \{\varphi_j;\ \varphi_j \in S_{h\mathsf{p}},\ j = 1, \ldots, \mathsf{dof}\}. \tag{1.100}$$

By dof (i.e. degrees of freedom), we denote the dimension of $S_{h\mathsf{p}}$ (=number of elements of the basis B) which is equal to

$$\mathsf{dof} = \sum_{\mu=1}^{M} \mathsf{dof}_\mu, \tag{1.101}$$

where dof_μ is given by (1.99).

Therefore, a function $u_h \in S_{h\mathsf{p}}$ can be written in the form

$$u_h(x) = \sum_{\mu=1}^{M} \sum_{j=1}^{\mathsf{dof}_\mu} u_{\mu,j}\varphi_{\mu,j}(x), \quad x \in \Omega, \tag{1.102}$$

where $u_{\mu,j} \in \mathbb{R},\ j = 1, \ldots, \mathsf{dof}_\mu,\ \mu = 1, \ldots, M$. Moreover, for $u_h \in S_{h\mathsf{p}}$, we define the vector of its basis coefficients by

$$\boldsymbol{u} = \{u_{\mu,j}\}_{j=1,\ldots,\mathsf{dof}_\mu}^{\mu=1,\ldots,M} \in \mathbb{R}^{\mathsf{dof}}. \tag{1.103}$$

Therefore, using (1.102) – (1.103) we have an isomorphism

$$u_h \in S_{h\mathsf{p}} \quad \longleftrightarrow \quad \boldsymbol{u} \in \mathbb{R}^{\mathsf{dof}}. \tag{1.104}$$

1.4.4 Construction of the DG basis functions

In order to achieve reasonable efficiency, hierarchical basis functions should be employed. We put

$$\bar{p} = \max_{\mu=1,\ldots,M} p_\mu.$$

Let $P^{\bar{p}}(\widehat{T})$ denote the space of all polynomials on $\widehat{T}$ of degree $\leq p_{\bar{p}}$. Let $\hat{S}_{\bar{p}}$ be a hierarchical basis of $P^{\bar{p}}(\widehat{T})$ which will be specified later. We call $\hat{S}_{\bar{p}}$ the *reference basis* and its elements the *reference shape functions*.

Furthermore, let $F_\mu := F_{T_\mu},\ \mu = 1, \ldots, M$, be the mapping introduced by (1.91) such that $F_\mu(\widehat{T}) = T_\mu$. We put

$$B_\mu := \{\varphi_{\mu,j};\ \varphi_{\mu,j}(\mathbf{x}) = \varphi_{\mu,j}(F_\mu(\hat{\mathbf{x}})) = \hat{\varphi}_j(\hat{\mathbf{x}}),\ \hat{\mathbf{x}} \in \widehat{T},\ j = 1, \ldots, \mathsf{dof}_\mu\}, \tag{1.105}$$

which defines a basis B_μ introduced in (1.98) for each element $T_\mu \in \mathscr{T}_h$ separately. Finally, (1.100) defines the *global basis* of $S_{h\mathsf{p}}$.

Reference shape functions

It is possible to use an arbitrary set of (hierarchical or non-hierarchical) shape functions for the definition of $\hat{S}_{\bar{p}}$. An example are the Lagrangian and Lobatto shape functions introduced in Sections 1.3.1 and 1.3.2, respectively. However, since $S_{h\mathsf{p}}$ is the space of discontinuous functions, it is not necessary to construct vertex, edge and bubble functions in order to fulfil the continuity of the basis functions.

Therefore, it is possible to use the *Taylor shape functions* given by

$$\left\{ \hat{\varphi}_{i_1,\ldots,i_d}(\hat{x}_1, \ldots, \hat{x}_d) = \Pi_{j=1}^d (\hat{x}_j - \hat{x}_j^c)^{i_j}; \quad i_1, \ldots, i_d \geq 0,\ \sum_{j=1}^d i_j \leq \bar{p} \right\}, \tag{1.106}$$

where $(\hat{x}_1^c, \ldots, \hat{x}_d^c)$ is the barycentre of $\widehat{T}$. (It is possible to use also $\hat{x}_1^c = \cdots = \hat{x}_d^c = 0$). The advantage of Taylor shape functions is a simpler implementation since we need not distinguish among the vertex, edge and bubble functions, one subroutine is sufficient.

However, a direct use of the Taylor shape functions is not too practical, since these functions are not normalized which can cause problems in the solution of the corresponding linear algebraic systems. Moreover, in many application, namely in evolution problems, it is advantageous to employ an L^2-orthonormal basis because the corresponding linear systems have favorable computational properties. Let us note that a suitable preconditioning can reduce many drawbacks following from the possible use of non-normalized shape functions.

Therefore, we employ the local character of the shape functions and construct a basis of $\hat{S}_{\bar{p}}$ which is *orthonormal* with respect to the *L^2-scalar product.* The simplest way how to obtain a L^2-orthonormal basis is the performance of the Gram-Schmidt orthogonalization process applied to the functions from (1.106). Then we obtain the orthonormal set of the reference shape functions

$$\hat{S}_{\bar{p}} := \{\hat{\varphi}_j,\ j = 1, \dots, \mathsf{dof}_{\bar{\mu}}\}, \tag{1.107}$$

where $\mathsf{dof}_{\bar{\mu}}$ is given by (1.99) with $p_{\bar{\mu}} := \bar{p}$.

It is a well known fact that the Gram-Schmidt orthogonalization is an unstable algorithm. However, in practice, this approach works, since $\mathsf{dof}_{\bar{\mu}}$ is usually not too large a number (between 10 and 200) and moreover, the instability of the Gram-Schmidt process causes a small violation of the L^2-orthogonality of the reference shape functions which do not affect good computational properties of the corresponding linear algebraic systems.

The Gram-Schmidt orthogonalization on the reference element can be carried out symbolically by some suitable software (e.g., Maple) or numerically. Our numerical experiments based on the numerical realization of the Gram-Schmidt orthogonalization work with success for $\bar{p} = 10 \ \Rightarrow\ \mathsf{dof}_{\bar{\mu}} = 45$ (for $d = 2$).

Finally, let us note that the global basis of $S_{h\mathsf{p}}$ obtained from (1.100), (1.105) and (1.107) is orthonormal if the mappings $F_\mu := F_{T_\mu},\ \mu = 1, \dots, M$ are linear. Otherwise, some violations of the L^2-orthogonality are present. However, the mappings $F_\mu,\ \mu = 1, \dots, M$ are close to linear mappings, hence these violations are small and (again) do not affect good computational properties of the corresponding linear algebraic systems.

1.4.5 Evaluation of volume integrals

The evaluation of the volume integrals in (1.97) is very simple. Similarly as in the conforming FEM, we have to compute the terms

$$\sum_{T\in\mathscr{T}_h} \int_T \nabla\varphi_{\mu,i}\cdot\nabla\varphi_{\nu,j}\,\mathrm{d}x, \qquad i = 1, \dots, \mathsf{dof}_\mu,\ j = 1, \dots, \mathsf{dof}_\nu,\ \mu, \nu = 1, \dots, M, \tag{1.108}$$

where $\varphi_{\mu,i}, \varphi_{\nu,j} \in B$ (= basis of $S_{h\mathsf{p}}$). Since each $\varphi_{\mu,i}$ vanishes outside of T_μ, we have

$$\sum_{T\in\mathscr{T}_h} \int_T \nabla\varphi_{\mu,i}\cdot\nabla\varphi_{\nu,j}\,\mathrm{d}x, = \delta_{\mu,\nu} \int_{T_\mu} \nabla\varphi_{\mu,i}\cdot\nabla\varphi_{\mu,j}\,\mathrm{d}x, \tag{1.109}$$

for all $i = 1, \dots, \mathsf{dof}_\mu,\ j = 1, \dots, \mathsf{dof}_\nu,\ \mu, \nu = 1, \dots, M$. Therefore, the stiffness matrix $\mathbb{S}$ is block diagonal with blocks $\mathbb{S}_\mu,\ \mu = 1, \dots, M$ given by

$$\mathbb{S}_\mu = \{S_{\mu,i,j}\}_{i,j=1}^{\mathsf{dof}_\mu}, \quad S_{\mu,i,j} = \int_{T_\mu} \nabla\varphi_{\mu,i}\cdot\nabla\varphi_{\mu,j}\,\mathrm{d}x, \quad i, j = 1, \dots, \mathsf{dof}_\mu. \tag{1.110}$$

These integrals can be simply evaluated wit the aid of (1.58), namely

$$\int_{T_\mu} \nabla\varphi_{\mu,i}\cdot\nabla\varphi_{\mu,j}\,\mathrm{d}x = \int_{\widehat{T}} \sum_{l=1}^{d} \left(J_{F_{T_\mu}}^{-\mathrm{T}}\widehat{\nabla}\hat{\varphi}_i\right)_l \left(J_{F_{T_\mu}}^{-\mathrm{T}}\widehat{\nabla}\hat{\varphi}_j\right)_l |\det J_{F_{T_\mu}}|\,\mathrm{d}\hat{\mathbf{x}}, \tag{1.111}$$

where $\hat{\varphi}_i,\ \hat{\varphi}_j$ are the reference shape functions from $\hat{S}_{\bar{p}}$ given by (1.107). In comparison to the conforming FEM, the situation is simpler, since we know that $\varphi_{\mu,i}$ corresponds to $\hat{\varphi}_i$, $i = 1,\dots,\mathsf{dof}_\mu$, compare with (1.56).

1.4.6 Evaluation of face integrals

In this section we deal with the evaluation of the face integrals in (1.97). Let us note that the necessity to evaluate face integrals appears also in the conforming finite element method, e.g., for the implementation of general boundary conditions, see Section 1.1.5. For simplicity, we restrict only to the case $d = 2$.

Let $e \in \mathscr{F}_h$ be an edge of $T \in \mathscr{T}_h$. We call e a physical edge. Our aim is to evaluate the integrals

$$\int_e f(\mathbf{x})\,\mathrm{d}\sigma, \qquad \int_e \boldsymbol{f}(\mathbf{x})\cdot\boldsymbol{n}\,\mathrm{d}\sigma, \tag{1.112}$$

where $\boldsymbol{n}$ is the normal vector to e and $f : e \to \mathbb{R}$, $\boldsymbol{f} : e \to \mathbb{R}^2$ are given functions. Let us recall the definition of the face integral. Let $\psi = (\psi_1,\psi_2) : [0,1] \to e$ be a parameterization of the edge e. Then

$$\int_e f(\mathbf{x})\,\mathrm{d}\sigma = \int_0^1 f(\psi(t))\sqrt{\left(\psi_1(t)'\right)^2 + \left(\psi_2(t)'\right)^2}\mathrm{d}t, \tag{1.113}$$

where $\psi_i(t)'$, $i = 1,2$ denotes the derivative of $\psi_i(t)$ with respect to t.

In order to evaluate the integrals (1.112), we use the approach based on the reference element. Let $\hat{e}$ be an edge of the reference element $\widehat{T}$ such that $T = F_T(\widehat{T})$ and $e = F_T(\hat{e})$. We call $\hat{e}$ the reference edge. Let

$$\hat{\mathbf{x}}_{\hat{e}}(t) = (\hat{x}_{\hat{e},1}(t), \hat{x}_{\hat{e},2}(t)) : [0,1] \to \hat{e} \tag{1.114}$$

be a parameterization of the reference edge $\hat{e}$ preserving the counterclockwise orientation of the element. Namely, using the notation from Figure 1.6, we have

$$\text{if } \widehat{T} = \widehat{T}_t \text{ then} \qquad \begin{aligned} &\hat{\mathbf{x}}_{\hat{e}_1}(t) := (t,0), \quad t \in (0,1),\\ &\hat{\mathbf{x}}_{\hat{e}_2}(t) := (1-t,t), \quad t \in (0,1),\\ &\hat{\mathbf{x}}_{\hat{e}_3}(t) := (0,1-t), \quad t \in (0,1), \end{aligned} \tag{1.115}$$

$$\text{if } \widehat{T} = \widehat{T}_q \text{ then} \qquad \begin{aligned} &\hat{\mathbf{x}}_{\hat{e}_1}(t) := (t,0), \quad t \in (0,1),\\ &\hat{\mathbf{x}}_{\hat{e}_2}(t) := (1,t), \quad t \in (0,1),\\ &\hat{\mathbf{x}}_{\hat{e}_3}(t) := (1-t,1), \quad t \in (0,1),\\ &\hat{\mathbf{x}}_{\hat{e}_4}(t) := (0,1-t), \quad t \in (0,1). \end{aligned}$$

Moreover, we define the infinitesimal increase of $\hat{\mathbf{x}}$ by

$$\mathrm{d}\hat{\mathbf{x}}_{\hat{e}}(t) := \frac{\mathrm{d}}{\mathrm{d}t}\hat{\mathbf{x}}_{\hat{e}}(t) \in \mathbb{R}^2, \tag{1.116}$$

namely

$$\text{if } \widehat{T} = \widehat{T}_t \text{ then} \quad \begin{aligned} \mathrm{d}\hat{\mathbf{x}}_{\hat{e}_1} &:= (1,0), \quad t \in (0,1), \\ \mathrm{d}\hat{\mathbf{x}}_{\hat{e}_2} &:= (-1,1), \quad t \in (0,1), \\ \mathrm{d}\hat{\mathbf{x}}_{\hat{e}_3} &:= (0,-1), \quad t \in (0,1), \end{aligned} \tag{1.117}$$

$$\text{if } \widehat{T} = \widehat{T}_q \text{ then} \quad \begin{aligned} \mathrm{d}\hat{\mathbf{x}}_{\hat{e}_1} &:= (1,0), \quad t \in (0,1), \\ \mathrm{d}\hat{\mathbf{x}}_{\hat{e}_2} &:= (0,1), \quad t \in (0,1), \\ \mathrm{d}\hat{\mathbf{x}}_{\hat{e}_3} &:= (-1,0), \quad t \in (0,1), \\ \mathrm{d}\hat{\mathbf{x}}_{\hat{e}_4} &:= (0,-1), \quad t \in (0,1). \end{aligned}$$

Therefore, the physical edge e is parameterized by

$$\begin{aligned} e: \quad \mathbf{x} &= F_T(\hat{\mathbf{x}}_{\hat{e}}(t)) = (F_{T,1}(\hat{\mathbf{x}}_{\hat{e}}(t)), F_{T,2}(\hat{\mathbf{x}}_{\hat{e}}(t))) \\ &= (F_{T,1}(\hat{x}_{\hat{e},1}(t), \hat{x}_{\hat{e},2}(t)), F_{T,2}(\hat{x}_{\hat{e},1}(t), \hat{x}_{\hat{e},2}(t))), \quad t \in [0,1]. \end{aligned} \tag{1.118}$$

The first integral from (1.112) is given by

$$\begin{aligned} \int_e f(\mathbf{x})\,\mathrm{d}\sigma &= \int_0^1 f(F_T(\hat{\mathbf{x}}_{\hat{e}}(t))) \left(\sum_{i=1}^2 \left(\frac{\mathrm{d}}{\mathrm{d}t} F_{T,i}(\hat{\mathbf{x}}_{\hat{e}}(t))\right)^2\right)^{1/2} \mathrm{d}t \\ &= \int_0^1 f(F_T(\hat{\mathbf{x}}_{\hat{e}}(t))) \left(\sum_{i=1}^2 \left(\sum_{j=1}^2 \frac{\partial F_{T,i}(\hat{\mathbf{x}}_{\hat{e}}(t))}{\partial \hat{x}_j} \frac{\mathrm{d}}{\mathrm{d}t}\hat{x}_{e,j}(t)\right)^2\right)^{1/2} \mathrm{d}t \\ &= \int_0^1 f(F_T(\hat{\mathbf{x}}_{\hat{e}}(t)))\, |J_{F_T}(\hat{\mathbf{x}}_{\hat{e}}(t))\mathrm{d}\hat{\mathbf{x}}_{\hat{e}}|\ \mathrm{d}t, \end{aligned} \tag{1.119}$$

where J_{F_T} is the Jacobian matrix of the mapping F_T multiplied by the vector $\mathrm{d}\hat{\mathbf{x}}_e$ given by (1.116) and $|\cdot|$ is the Euclidean norm of the vector. Let us note that if F_T is a linear mapping then e is a straight edge and $|J_{F_T}(\hat{\mathbf{x}}_{\hat{e}}(t))\mathrm{d}\hat{\mathbf{x}}_{\hat{e}}(t)|$ is equal to its length.

Now, we focus on the second integral from (1.112). Let $\boldsymbol{r}_e$ be the tangential vector to e (if e is a straight line then $\boldsymbol{r}_e = e$). Using (1.118) and (1.119), we evaluate $\boldsymbol{r}_e$ at $\mathbf{x}(t) = F_T(\hat{\mathbf{x}}_{\hat{e}}(t)),\ t \in [0,1]$ by

$$\begin{aligned} \boldsymbol{r}_e(\mathbf{x}(t)) &= (r_{e,1}(\mathbf{x}(t)), r_{e,2}(\mathbf{x}(t))) \\ &:= \frac{\mathrm{d}}{\mathrm{d}t} F_T(\hat{\mathbf{x}}_{\hat{e}}(t)) = \left(J_{F_{T,1}}(\hat{\mathbf{x}}_{\hat{e}}(t))\mathrm{d}\hat{\mathbf{x}}_{\hat{e}}, J_{F_{T,2}}(\hat{\mathbf{x}}_{\hat{e}}(t))\mathrm{d}\hat{\mathbf{x}}_{\hat{e}}\right). \end{aligned} \tag{1.120}$$

Now, by rotation we obtain the normal vector $\boldsymbol{n}_e$ pointing outside of T, namely

$$\begin{aligned} \boldsymbol{n}_e(\mathbf{x}(t)) &= (n_{e,1}(\mathbf{x}(t)), n_{e,2}(\mathbf{x}(t))) : \\ & \quad n_{e,1}(\mathbf{x}(t)) = r_{e,2}(\mathbf{x}(t)),\ n_{e,2}(\mathbf{x}(t)) = -r_{e,1}(\mathbf{x}(t)), \end{aligned} \tag{1.121}$$

where $(r_{e,1}, r_{e,2})$ is the tangential vector to e given by (1.120). Here it is important that the counterclockwise orientation of the elements is considered. Therefore, from (1.120) and (1.121), we have

$$\boldsymbol{n}_e(\mathbf{x}(t)) \quad = \quad \left(J_{F_{T,2}}(\hat{\mathbf{x}}_{\hat{e}}(t))\mathrm{d}\hat{\mathbf{x}}_{\hat{e}}, -J_{F_{T,1}}(\hat{\mathbf{x}}_{\hat{e}}(t))\mathrm{d}\hat{\mathbf{x}}_{\hat{e}}\right). \tag{1.122}$$

Let us note that $\boldsymbol{n}_e(\mathbf{x}(t))$ is not normalized, it is necessary to divide it by $|\boldsymbol{n}_e(\mathbf{x}(t))| = |J_{F_T}(\hat{\mathbf{x}}_{\hat{e}}(t))\mathrm{d}\hat{\mathbf{x}}_{\hat{e}}|$. Finally, similarly as in (1.119), we obtain

$$\begin{aligned} \int_e \boldsymbol{f}(\mathbf{x}) \cdot \boldsymbol{n}\,\mathrm{d}\sigma \quad &= \quad \int_0^1 \boldsymbol{f}(F_T(\hat{\mathbf{x}}_{\hat{e}}(t))) \cdot \frac{\boldsymbol{n}_e(\mathbf{x}(t))}{|\boldsymbol{n}_e(\mathbf{x}(t))|}\,|J_{F_T}(\hat{\mathbf{x}}_{\hat{e}}(t))\mathrm{d}\hat{\mathbf{x}}_{\hat{e}}|\,\mathrm{d}t, \qquad (1.123) \\ &= \quad \int_0^1 \boldsymbol{f}(F_T(\hat{\mathbf{x}}_{\hat{e}}(t))) \cdot \boldsymbol{n}_e(\mathbf{x}(t))\,\mathrm{d}t, \end{aligned}$$

where $\boldsymbol{n}_e(\mathbf{x}(t))$ is given by (1.122). Let us note that if F_T is a linear mapping then e is a straight edge and $|\boldsymbol{n}_e(\mathbf{x}(t))|$ is equal to its length.

1.4.7 Data structures

In order to evaluate the integrals (1.111), (1.119) and (1.123), it is enough to evaluate and store the following data.

(S1) for each $T \in \mathscr{T}_h$, the determinant of the Jacobi matrix $\det J_{F_T}$ and the transposed matrix of the inversion of the Jacobi matrix J_{F_T},

(S2) the shape functions $\hat{\varphi}_i,\ i = 1, \dots \hat{N}$ with the gradients $\widehat{\nabla}\hat{\varphi}_i,\ i = 1, \dots \hat{N}$ on $\widehat{T}$.

Comparison with the data structures of the FEM implementation, we see that (S1) and (S2) are in fact the same, but (S3) is missing. This is caused by the fact that we have a direct connection between basis functions and corresponding shape functions, see relation (1.105).

1.5 Numerical quadratures

In the previous sections we assumed that we are able to evaluate all integrals (over the reference element or the reference edge) exactly. This is true for the model problem which is linear and therefore all integrands are polynomial functions. However, when a problem is nonlinear, it is necessary to use an approximation, namely suitable numerical quadratures. Even in the case when the problem is linear, the implementation with numerical quadratures is simpler.

There exist many numerical quadrature rules with many advantages and disadvantages. In the following we exhibit some basic quadrature rules for edge integrals and volume integrals over the reference square and over the reference triangle, which exhibit a reasonable compromise between accuracy and implementational complexity.

1.5.1 Edge quadratures

We consider the *Gauss quadrature rules* given by

$$\int_0^1 f(x)\,\mathrm{d}x \approx G_k(f) := \sum_{i=1}^{k} w_i f(x_i), \tag{1.124}$$

where $w_i,\ i = 1,\dots,k$ and $x_i,\ i = 1,\dots,k$ are the Gauss weights and Gauss nodes, respectively. Their values can be found in many textbooks, e.g., in [3]. Table 1.4 gives these values for $G_k,\ k = 1,\dots,12$.

1.5.2 Quadratures on quadrilaterals

In order to integrate over the reference square $\widehat{T}_q$, we employ the *bi-Gauss quadrature rules* given by

$$\int_{\widehat{T}_q} f(\mathbf{x})\,\mathrm{d}\mathbf{x} \approx G_k^{\mathrm{bi}}(f) := \sum_{i=1}^{k}\sum_{j=1}^{k} w_i w_j f(\mathbf{x}_{ij}), \tag{1.125}$$

where $w_i,\ i = 1,\dots,k$ are the Gauss weights introduced in (1.124) and

$$\mathbf{x}_{ij} = (x_i, x_j), \quad i,j = 1,\dots,k, \tag{1.126}$$

where $x_i,\ i = 1,\dots,k$ are the Gauss nodes introduced in (1.124). Obviously, the bi-Gauss quadrature rules are derived by integration with respect to the first variable and then with respect to the second one. Similarly, it is possible to derive a quadrature rule for the reference cube for $d = 3$.

1.5.3 Quadratures on triangles

In order to integrate over the reference triangle $\widehat{T}_t$, we employ the *Dunavant quadrature rules* given by

$$\int_{\widehat{T}_t} f(\mathbf{x})\,\mathrm{d}\mathbf{x} \approx D_k(f) := \sum_{i=1}^{m_k} w_i f(\mathbf{x}_i), \tag{1.127}$$

where m_k is the number of integration nodes, $w_i,\ i = 1,\dots,k$ are the Dunavant weights and $\mathbf{x}_i = (\lambda_{i,1}, \lambda_{i,2}, \lambda_{i,3}),\ i = 1,\dots,m_k$, are the Dunavant integration nodes. Their values can be found, e.g., in [2], Table 1.5 shows these values for $D_k,\ k = 1,\dots,7$.

1.5.4 Data structure

If the integrals appearing in FEM and/or DGM are evaluated with the aid of numerical quadratures, we should revise the data structures (S1), (S2), (S3) introduced in Sections 1.2.4 and 1.4.7. Therefore, instead of (S1) and (S2), it is enough to evaluate and store the following data.

(D1) for each $T \in \mathscr{T}_h$, the determinant of the Jacobi matrix $\det J_{F_T}$ and the transposed matrix of the inversion of the Jacobi matrix J_{F_T} in the *edge and volume quadrature nodes used*,

G_k	j	w_j	x_j	j	w_j	x_j
G_1	1	1.00000000000000	0.50000000000000			
G_2	1	0.50000000000000	0.21132486540519	2	0.50000000000000	0.78867513459481
G_3	1	0.27777777777778	0.11270166537926	2	0.44444444444444	0.50000000000000
G_3	3	0.27777777777778	0.88729833462074			
G_4	1	0.17392742256873	0.06943184420297	2	0.32607257743127	0.33000947820757
G_4	3	0.32607257743127	0.66999052179243	4	0.17392742256873	0.93056815579703
G_5	1	0.11846344252809	0.04691007703067	2	0.23931433524968	0.23076534494716
G_5	3	0.28444444444444	0.50000000000000	4	0.23931433524968	0.76923465505284
G_5	5	0.11846344252809	0.95308992296933			
G_6	1	0.08566224618959	0.03376524289842	2	0.18038078652407	0.16939530676687
G_6	3	0.23395696728635	0.38069040695840	4	0.23395696728635	0.61930959304160
G_6	5	0.18038078652407	0.83060469323313	6	0.08566224618959	0.96623475710158
G_7	1	0.06474248308443	0.02544604382862	2	0.13985269574464	0.12923440720030
G_7	3	0.19091502525256	0.29707742431130	4	0.20897959183673	0.50000000000000
G_7	5	0.19091502525256	0.70292257568870	6	0.13985269574464	0.87076559279970
G_7	7	0.06474248308443	0.97455395617138			
G_8	1	0.05061426814519	0.01985507175123	2	0.11119051722669	0.10166676129319
G_8	3	0.15685332293894	0.23723379504184	4	0.18134189168918	0.40828267875218
G_8	5	0.18134189168918	0.59171732124782	6	0.15685332293894	0.76276620495816
G_8	7	0.11119051722669	0.89833323870681	8	0.05061426814519	0.98014492824877
G_9	1	0.04063719418079	0.01591988024619	2	0.09032408034743	0.08198444633668
G_9	3	0.13030534820147	0.19331428364970	4	0.15617353852000	0.33787328829810
G_9	5	0.16511967750063	0.50000000000000	6	0.15617353852000	0.66212671170190
G_9	7	0.13030534820147	0.80668571635030	8	0.09032408034743	0.91801555366332
G_9	9	0.04063719418079	0.98408011975381			
G_{10}	1	0.03333567215434	0.01304673574141	2	0.07472567457529	0.06746831665551
G_{10}	3	0.10954318125799	0.16029521585049	4	0.13463335965500	0.28330230293538
G_{10}	5	0.14776211235738	0.42556283050918	6	0.14776211235738	0.57443716949082
G_{10}	7	0.13463335965500	0.71669769706462	8	0.10954318125799	0.83970478414951
G_{10}	9	0.07472567457529	0.93253168334449	10	0.03333567215434	0.98695326425859
G_{11}	1	0.02783428355809	0.01088567092697	2	0.06279018473245	0.05646870011595
G_{11}	3	0.09314510546387	0.13492399721298	4	0.11659688229600	0.24045193539659
G_{11}	5	0.13140227225512	0.36522842202383	6	0.13646254338895	0.50000000000000
G_{11}	7	0.13140227225512	0.63477157797617	8	0.11659688229600	0.75954806460341
G_{11}	9	0.09314510546387	0.86507600278702	10	0.06279018473245	0.94353129988405
G_{11}	11	0.02783428355809	0.98911432907303			
G_{12}	1	0.02358766819326	0.00921968287664	2	0.05346966299766	0.04794137181476
G_{12}	3	0.08003916427167	0.11504866290285	4	0.10158371336153	0.20634102285669
G_{12}	5	0.11674626826918	0.31608425050091	6	0.12457352290670	0.43738329574427
G_{12}	7	0.12457352290670	0.56261670425573	8	0.11674626826918	0.68391574949909
G_{12}	9	0.10158371336153	0.79365897714331	10	0.08003916427167	0.88495133709715
G_{12}	11	0.05346966299766	0.95205862818524	12	0.02358766819326	0.99078031712336

Table 1.4: Gauss quadratures $G_k,\ k = 1, \dots, 12$ with weights w_i and nodes x_i

D_k	j	w_j	$\lambda_{j,1}$	$\lambda_{j,2}$	$\lambda_{j,3}$
D_1	1	1.00000000000000	0.33333333333333	0.33333333333333	0.33333333333333
D_2	1	0.33333333333333	0.66666666666667	0.16666666666667	0.16666666666667
D_2	2	0.33333333333333	0.16666666666667	0.16666666666667	0.66666666666667
D_2	3	0.33333333333333	0.16666666666667	0.66666666666667	0.16666666666667
D_3	1	-.56250000000000	0.33333333333333	0.33333333333333	0.33333333333333
D_3	2	0.52083333333333	0.60000000000000	0.20000000000000	0.20000000000000
D_3	3	0.52083333333333	0.20000000000000	0.20000000000000	0.60000000000000
D_3	4	0.52083333333333	0.20000000000000	0.60000000000000	0.20000000000000
D_4	1	0.22338158967801	0.10810301816807	0.44594849091597	0.44594849091597
D_4	2	0.22338158967801	0.44594849091597	0.44594849091597	0.10810301816807
D_4	3	0.22338158967801	0.44594849091597	0.10810301816807	0.44594849091597
D_4	4	0.10995174365532	0.81684757298046	0.09157621350977	0.09157621350977
D_4	5	0.10995174365532	0.09157621350977	0.09157621350977	0.81684757298046
D_4	6	0.10995174365532	0.09157621350977	0.81684757298046	0.09157621350977
D_5	1	0.22500000000000	0.33333333333333	0.33333333333333	0.33333333333333
D_5	2	0.13239415278851	0.05971587178977	0.47014206410511	0.47014206410511
D_5	3	0.13239415278851	0.47014206410511	0.47014206410511	0.05971587178977
D_5	4	0.13239415278851	0.47014206410511	0.05971587178977	0.47014206410511
D_5	5	0.12593918054483	0.79742698535309	0.10128650732346	0.10128650732346
D_5	6	0.12593918054483	0.10128650732346	0.10128650732346	0.79742698535309
D_5	7	0.12593918054483	0.10128650732346	0.79742698535309	0.10128650732346
D_6	1	0.11678627572638	0.50142650965818	0.24928674517091	0.24928674517091
D_6	2	0.11678627572638	0.24928674517091	0.24928674517091	0.50142650965818
D_6	3	0.11678627572638	0.24928674517091	0.50142650965818	0.24928674517091
D_6	4	0.05084490637021	0.87382197101700	0.06308901449150	0.06308901449150
D_6	5	0.05084490637021	0.06308901449150	0.06308901449150	0.87382197101700
D_6	6	0.05084490637021	0.06308901449150	0.87382197101700	0.06308901449150
D_6	7	0.08285107561837	0.05314504984482	0.31035245103378	0.63650249912140
D_6	8	0.08285107561837	0.31035245103378	0.63650249912140	0.05314504984482
D_6	9	0.08285107561837	0.63650249912140	0.05314504984482	0.31035245103378
D_6	10	0.08285107561837	0.31035245103378	0.05314504984482	0.63650249912140
D_6	11	0.08285107561837	0.63650249912140	0.31035245103378	0.05314504984482
D_6	12	0.08285107561837	0.05314504984482	0.63650249912140	0.31035245103378
D_7	1	-.14957004446768	0.33333333333333	0.33333333333333	0.33333333333333
D_7	2	0.17561525743321	0.47930806784192	0.26034596607904	0.26034596607904
D_7	3	0.17561525743321	0.26034596607904	0.26034596607904	0.47930806784192
D_7	4	0.17561525743321	0.26034596607904	0.47930806784192	0.26034596607904
D_7	5	0.05334723560884	0.86973979419557	0.06513010290222	0.06513010290222
D_7	6	0.05334723560884	0.06513010290222	0.06513010290222	0.86973979419557
D_7	7	0.05334723560884	0.06513010290222	0.86973979419557	0.06513010290222
D_7	8	0.07711376089026	0.04869031542532	0.31286549600487	0.63844418856981
D_7	9	0.07711376089026	0.31286549600487	0.63844418856981	0.04869031542532
D_7	10	0.07711376089026	0.63844418856981	0.04869031542532	0.31286549600487
D_7	11	0.07711376089026	0.31286549600487	0.04869031542532	0.63844418856981
D_7	12	0.07711376089026	0.63844418856981	0.31286549600487	0.04869031542532
D_7	13	0.07711376089026	0.04869031542532	0.63844418856981	0.31286549600487

Table 1.5: Dunavant quadratures $D_k,\ k = 1, \dots, 7$ with weights w_i and nodes x_i

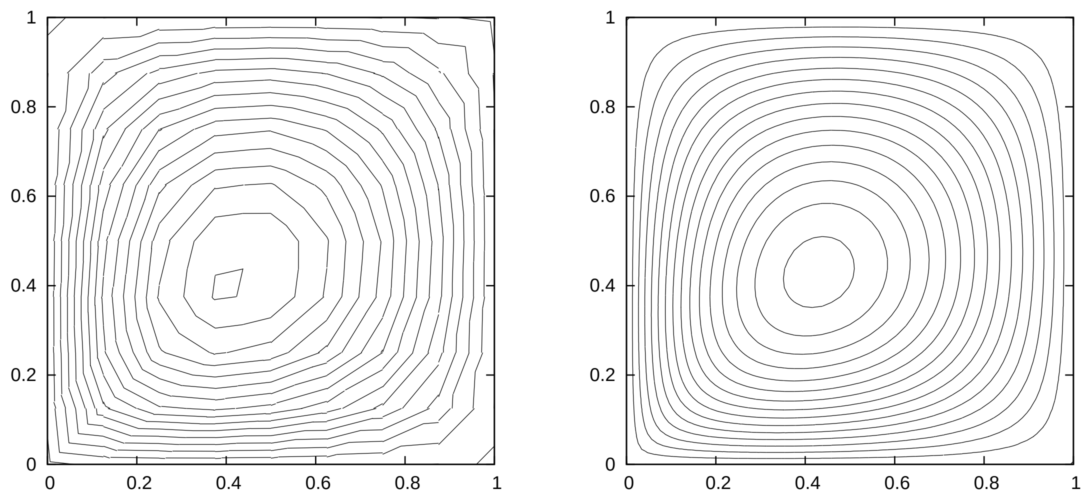


Figure 1.10: Basic graphical output: isolines, coarse grid (left) and fine grid (right)

(D2) the shape functions $\hat{\varphi}_i,\ i = 1, \dots \hat{N}$ with the gradients $\widehat{\nabla}\hat{\varphi}_i,\ i = 1, \dots \hat{N}$ on $\widehat{T}$ in the *edge and volume quadrature nodes used.*

However, there is a natural question which quadrature rules should be employed. For linear or nearly linear problem, it is sufficient to use quadrature rules such that integrate exactly polynomials of the degrees which appear in the volume and edge integrals. For strongly nonlinear problems, more accurate quadrature rule should be used, their degrees can be found empirically.

1.6 Basic visualization techniques

The result of a finite element computation is a (continuous or discontinuous) piecewise polynomial function $u_h \in V_h$. Usually, we are not interested of its analytical expression but we need its visualization. In this section, we discuss these aspects.

1.6.1 Types of visualization

There are several usual graphical outputs, which can be required based on the considered problem. Namely

- *isolines/isosurfaces of the solutions* are the lines (for $d = 2$) or surfaces (for $d = 3$) consisting of points of the computational domains where the approximate solution is equal to a given value, i.e., the set $\{\boldsymbol{x} \in \Omega;\ u_h(\boldsymbol{x}) = g\}$ is the isoline (isosurface) corresponding to the value g. Figure 1.10 shows the isolines of the piecewise linear approximation of the function

$$u(x_1, x_2) = 2(x_1^2+x_2^2)^{-1/4}x_1x_2(1-x_1)(1-x_2), \qquad (x_1, x_2) \in (0,1)\times(0,1) \quad (1.128)$$

for the values $g = 0.01n,\ n = 0, 1, \dots, 18.$

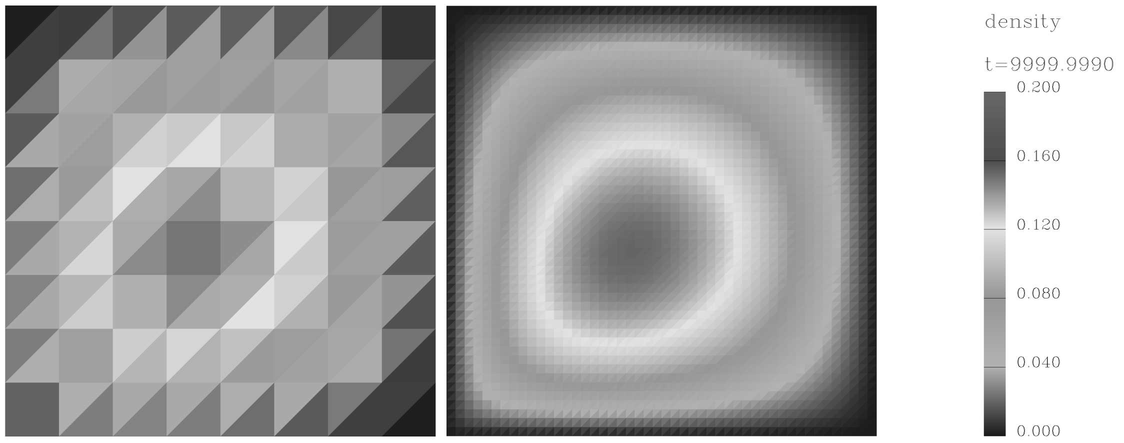


Figure 1.11: Basic graphical output: color maps, coarse grid (left) and fine grid (right)

- *color maps of the solution* is the visualization technique where each element of the mesh is drawn by a color which corresponds to a given value. Figure 1.11 shows the color maps of the function (1.128). Let us note that color maps very ofter smear details of approximate solution.

- *3D plot of the solution* (only for $d = 2$) is the graph of the surface

$$\{(x_1, x_2, x_3);\ x_3 = u_h(x_1, x_2),\ (x_1, x_2) \in \Omega\}.$$

  Figure 1.12 shows the 3D plots of the function (1.128). This graphical output contains the whole information of the approximate solution. On the other hand if the approximate solution is very complex this type of output is hard to see.

- *visualization on boundaries* is important, e.g., in aerodynamics, when we need to know a pressure distribution on the airplane surface.

- *cuts of the solution* help to see the solution inside of the computational domain, Figure 1.13 shows the diagonal cuts from the lower-left to upper-right corner of the function (1.128).

1.6.2 Software for visualization

Within this section we briefly describe types of software which can be used for the visualization of finite element solutions. Our aim is not to give a complete list of all available software but give to the readers an overview of some possibilities.

We suppose that the software for the visualization can be split into the following groups.

- *Commercial graphical software* usually represent a complete tool for visualization. Let us mention as an example the code Techplot which is able to draw a continuous piecewise linear or piecewise constant approximate solution for $d = 2$ as well as

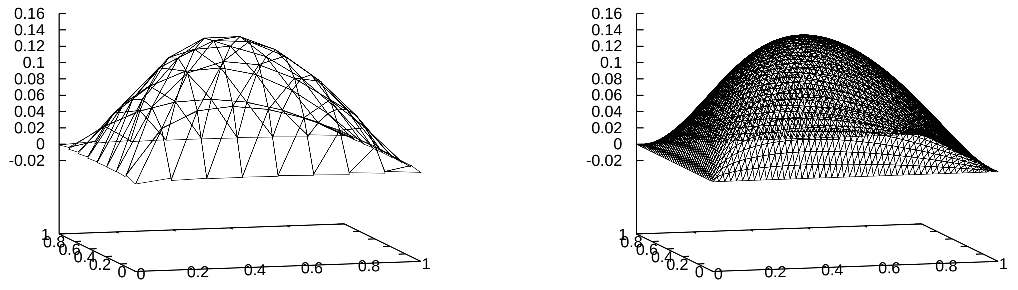


Figure 1.12: Basic graphical output: 3D plots, coarse grid (left) and fine grid (right)

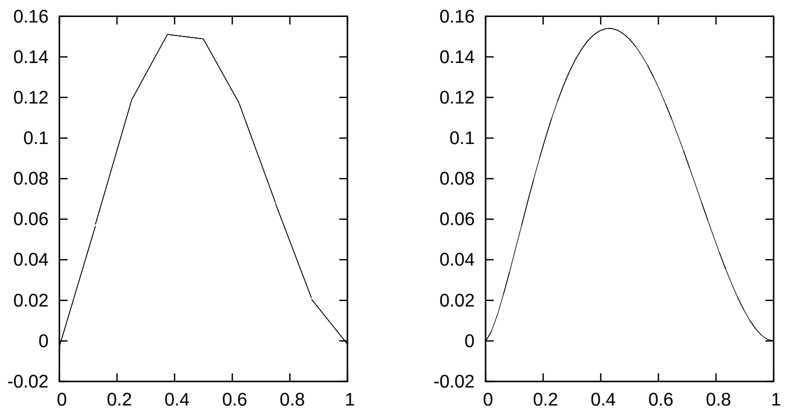


Figure 1.13: Basic graphical output: diagonal cuts, coarse grid (left) and fine grid (right)

$d = 3$. It allows a wide range of graphical tools, e.g., slices, isosurfaces, cuts, animations etc. However, Techplot uses some postprocessing procedures which smooths the given approximate solution. This is advantageous for the effect of visualization but it introduces some additional perturbations and/or "errors". The use of commercial software does not require any pre-computing, in generally. However, most of the commercial software does not allow a visualization of higher order approximations. In this case, some additional technique has to be used, see Section 1.6.3.

- *Free graphical libraries* represented, e.g., by PGPLOT, which is a set of Fortran 77 subroutines which are able to draw a given line, to fill a given polygon by a given color, etc. Therefore, the use of this library requires the creation of code calling the PGPLOT subroutines. This represents additional work but you can draw exactly what you want, no postprocessing perturbs the results. Figure 1.10, right, was obtained by PGPLOT.

- *Basic graphical code* as, e.g., gnuplot, is able to carried out basic graphical tasks as connecting a given set of nodes. This type of software can be used with success for wall distribution, cuts and isolines. However, a visualization of isolines on unstructured grids requires their pre-computing by some additional code. Figures 1.10, left, and 1.12 were created by gnuplot.

1.6.3 Visualization of higher order polynomial functions

In this section we discuss some possibilities how to visualize a higher order piecewise polynomial solution u_h of an abstract problem. These *visualization techniques* are in fact independent of whether $u_h \in V_h$ is continuous or $u_h \in S_{h\mathsf{p}}$ is discontinuous.

In order to visualize a piecewise polynomial function, we need to evaluate its value in given nodes from Ω. If we use the data structure introduced in Section 1.5.4, we can simply evaluate the solution in integrations nodes. However, it is not usually enough, since the Gauss and Dunavant quadrature nodes do not lay at end points of $(0, 1)$ or at the boundary of the reference triangle. Therefore, for the purposes of visualization it is suitable to evaluate the solution at the Lagrangian nodes introduced in Section 1.3.1. This means that we have to evaluate and store also

(D2a) the shape functions $\hat{\varphi}_i,\ i = 1, \dots \hat{N}$ on $\widehat{T}$ in the nodes used for a visualization.

All graphical codes are able to draw a line between two nodes. Therefore, it is possible to visualize without problems a piecewise linear approximate solution. A visualization of higher order approximation is more complicated. It is possible to calculate exactly the corresponding solution dot per dot, but it is usually very time consuming.

A more efficient and sufficiently accurate approach is the following. Let $T \in \mathscr{T}_h$ be an element and $u_h|_T$ a polynomial function of degree p. Then we split T into several sub-elements by connecting the Lagrangian nodes $F_T(\hat{\mathbf{z}}_i^p),\ i = 1, \dots, \hat{N}^p$, see Figure 1.14. We evaluate $u_h|_T$ at each of the Lagrangian nodes $F_T(\hat{\mathbf{z}}_i^p),\ i = 1, \dots, \hat{N}^p$ and carry out a piecewise linear visualization over this sub-grid.

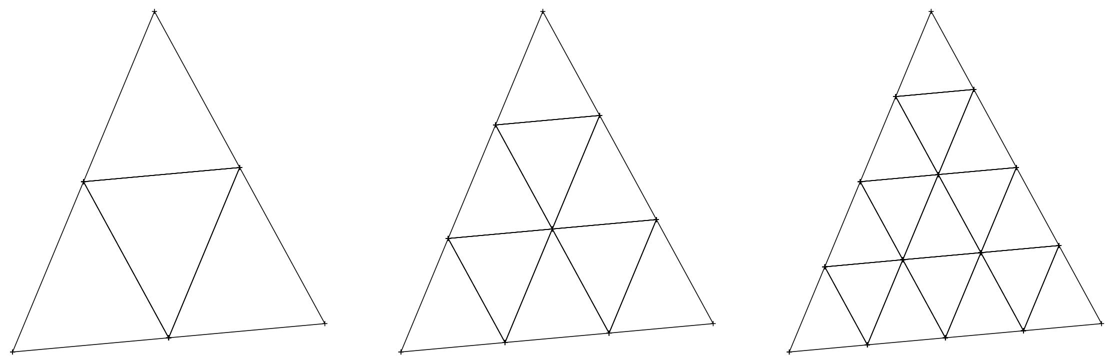


Figure 1.14: The sub-grids for the visualization on a triangle T for $p = 2$ (left), $p = 3$ (center) and $p = 4$ (right)

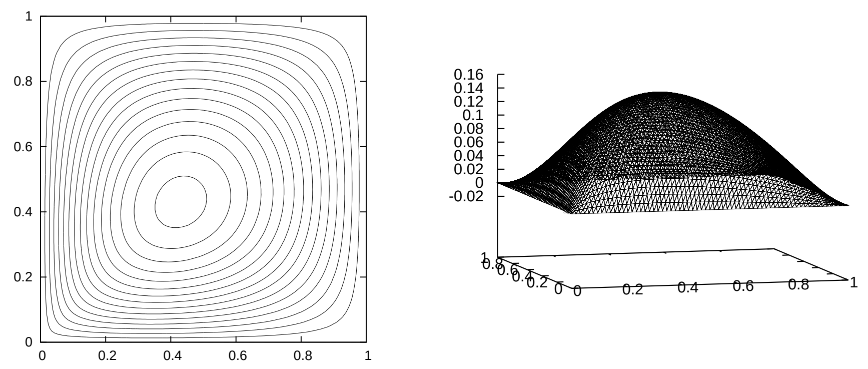


Figure 1.15: The visualization of the given function, isolines (left) and 3D plot (right)

Figure 1.15 shows the exact solution of a given problem which is visualized by isolines and a 3D plot. This problem was solved on a mesh consisting of four triangles (arising by splitting the square by its diagonals) using $P_k,\ k = 1, \ldots, 7$ approximations. Figures 1.16 – 1.22 show these approximate solutions using the visualization on the corresponding subgrids.

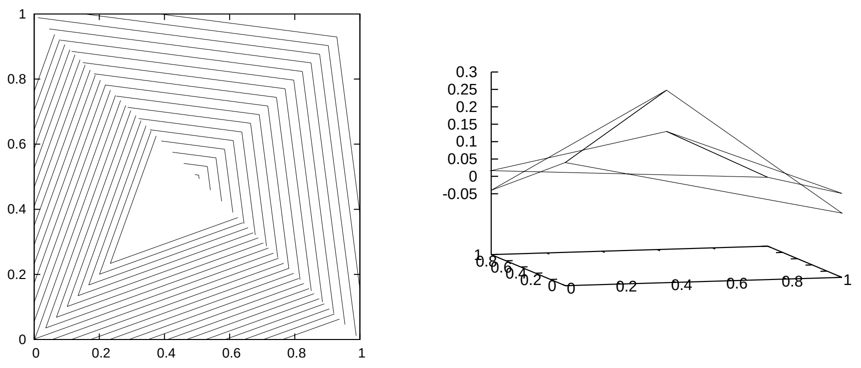


Figure 1.16: Visualization of the P_1 approximation, isolines (left) and 3D plot (right)

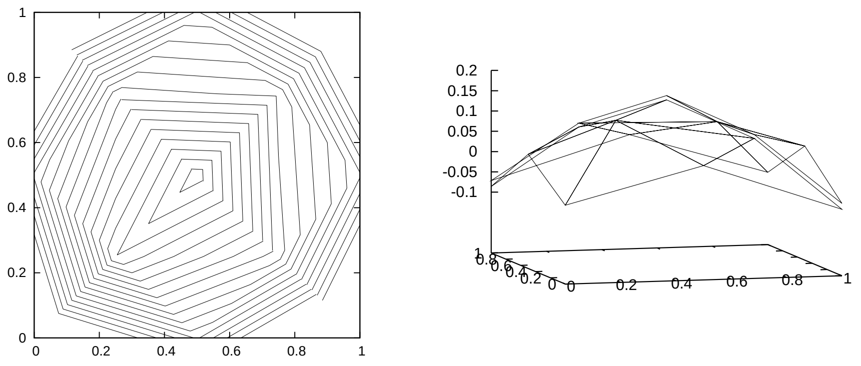


Figure 1.17: Visualization of the P_2 approximation, isolines (left) and 3D plot (right)

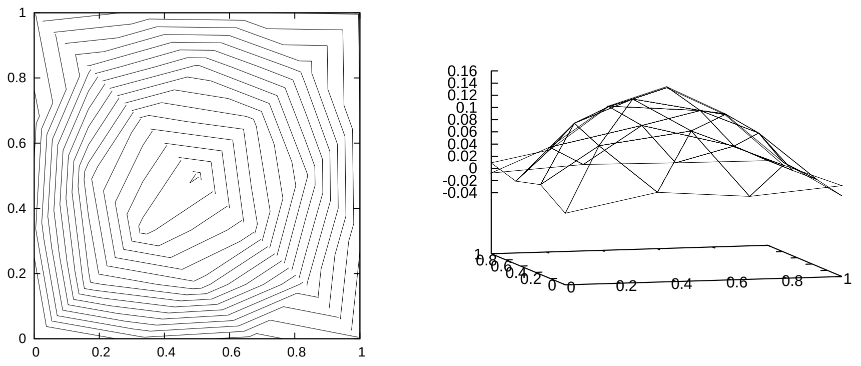


Figure 1.18: Visualization of the P_3 approximation, isolines (left) and 3D plot (right)

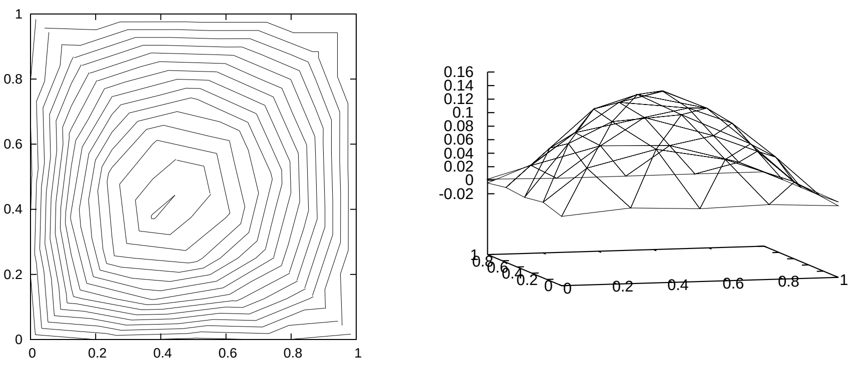


Figure 1.19: Visualization of the P_4 approximation, isolines (left) and 3D plot (right)

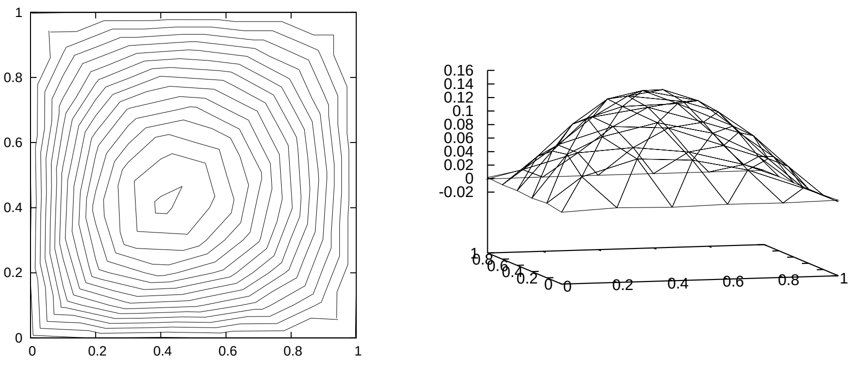


Figure 1.20: Visualization of the P_5 approximation, isolines (left) and 3D plot (right)

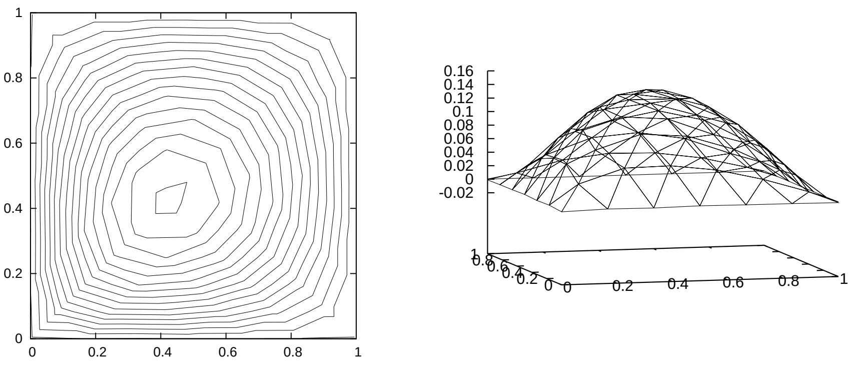


Figure 1.21: Visualization of the P_6 approximation, isolines (left) and 3D plot (right)

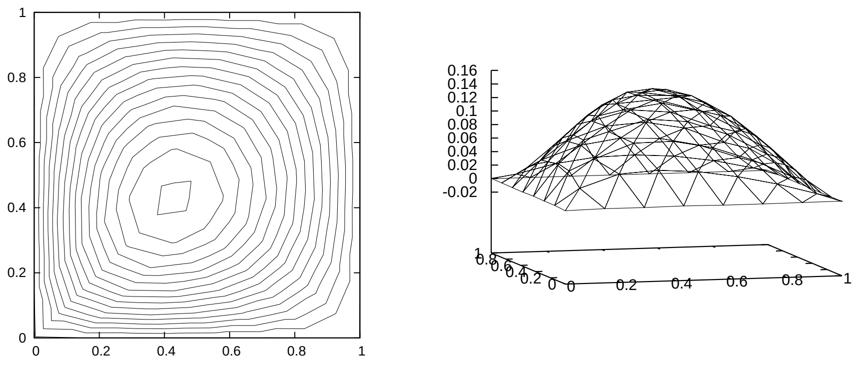


Figure 1.22: Visualization of the P_7 approximation, isolines (left) and 3D plot (right)

Bibliography

[1] L. Dubcová. *hp-FEM for coupled problems in fluid mechanics.* PhD thesis, Charles University Prague, 2010.

[2] D. A. Dunavant. High degree efficient symmetrical gaussian quadrature rules for the triangle. *Int. J. Numer. Methods Eng.*, 21:1129–1148, 1985.

[3] P. Šolín, K. Segeth, and I. Doležel. *Higher-Order Finite Element Methods.* Chapman & Hall/CRC Press, 2003.