Definition 2.1 (Mesh)**.** *A* **mesh** $\mathcal{T} = \{K_1, \ldots, K_{n_{el}}\}$ *for a domain* Ω *is a collection of a finite number* n_{el} *of element domains* $K_1, \ldots, K_{n_{el}}$ *such that* $\mathring{K}_i \cap \mathring{K}_j = \emptyset$ *and* $\Omega = \cup_{i=1}^{N_{el}} K_i$.

The symbol $\mathring{\omega}$ for $\omega \subset \mathbb{R}^2$ denotes the *interior* of ω. A point $x \in \mathring{\omega}$ if we can find a disc D centered at x such that $D \subset \omega$, that is, D is completely included in ω, see Fig. 2.9.

Figure 2.8 shows an example of a mesh. When all elements in a mesh are triangles (or tetrahedra in 3D), the mesh is also called a **triangulation**. **Vertices and edges** of a triangulation are any of the vertices and/or edges of the triangles in the triangulation.

Definition 2.2 (Continuous P_1 finite element space)**.** *Given a mesh* $\mathcal{T}$ *for a domain* Ω*, the* **continuous** P_1 **finite element space** *over* $\mathcal{T}$ *is the space* $\mathcal{W}_h$ *of all functions that belong to* $C^0(\Omega)$ *and are a polynomial of degree* $r = 1$ *in each of the element domains.*

A polynomial $p(x_1, x_2)$ of degree less or equal than 1 in two dimensions has the form

$$p(x_1, x_2) = c_1 + c_2 x_1 + c_3 x_2$$

Therefore, if we know the value of p at three different non-colinear points in the plane, we can uniquely identify the coefficients c_1, c_2 and c_3.

Given any partition of Ω it may be very hard to exhibit a basis of the P_1 space, but there are some special meshes, called **conforming triangulations** that make it very simple.

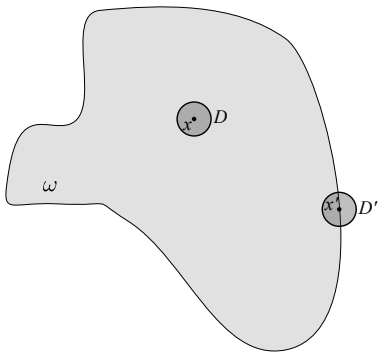

Figure 2.9 The point x belongs to the interior of ω, while x' does not.

Definition 2.3 (Conforming triangulation)**.** *A* **conforming triangulation** *of a polygonal domain* Ω *is a mesh for* Ω *such that the intersection of any two triangles* K *and* K' *is either (a) empty, or (b) a whole edge of* **both** K *and* K'*, or (c) a vertex of* **both** K *and* K'.

Figure 2.10 shows several meshes for a polygonal domain. The one labeled B is a conforming triangulation. The ones labeled A and C are not conforming.

Conforming triangulations are remarkable. Let $\mathcal{T}$ be a triangulation and let us number the vertices of the triangulation from 1 to n_V. Since every triangle has three vertices, by specifying the values $f_1, f_2, \ldots, f_{n_V}$ at all vertices of $\mathcal{T}$ one defines a *unique* $\mathbb{P}_1$-polynomial function f^K in each triangle K of $\mathcal{T}$. From these functions $\{f^K\}$, each defined in one triangle of $\mathcal{T}$, we can build a function f over Ω as

$$f(x) = f^K(x)\,, \quad \text{if } x \in K\,. \tag{2.28}$$

In other words, f takes at a point x the value corresponding to $f^K(x)$, if the point belongs to triangle K. All points belong to at least one triangle, so that $f(x)$ always exists. However, at some edges of a general triangulation the function may be multi-valued and thus f **may not be a continuous function**. This is easily understood by considering, in Fig. 2.10 (case A), the function f that is zero at all vertices except for the one and only interior vertex, at which the value is 1. This function, drawn in Fig. 2.11, is discontinuous along the edge over which the interior vertex is "hanging".

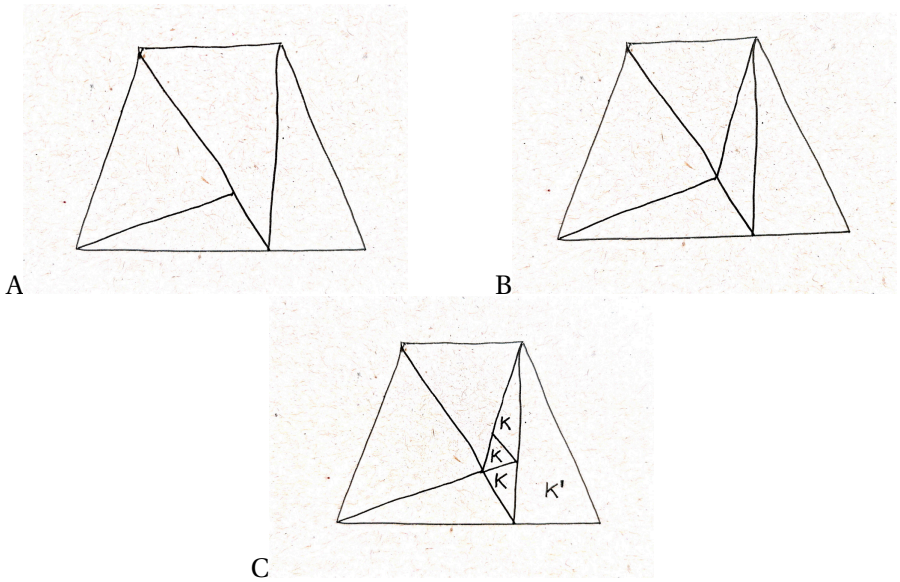

Figure 2.10 Examples of triangulations of a polygonal domain. B is conforming, A and C are not. The elements K in triangulation C that fail the definition of conforming triangulation when considered vis-à-vis K' are indicated.

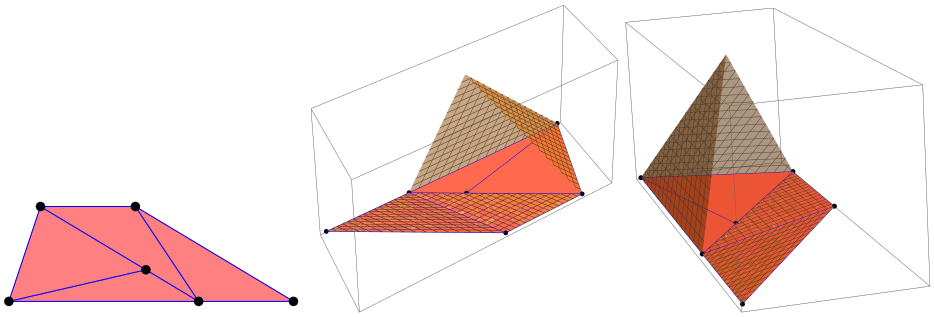

Figure 2.11 The non-conforming mesh in Fig. 2.10(Case A) is shown on the left, and two views of a piecewise linear polynomial function over it are displayed at the center and on the right. The function is obtained by setting the value of the interior vertex to 1, and to 0 at all the remaining vertices. Because the triangulation is not conforming, the resulting function is discontinuous along one of the interior edges.

Definition 2.4 (Hanging vertex)**.** *A* **hanging vertex** *in a triangulation $\mathcal{T}$ is a vertex v for which there exists a triangle K in $\mathcal{T}$ satisfying*

$$v \in K, \qquad \text{and} \qquad v \text{ is not a vertex of } K\,.$$

An alternative definition of **conforming triangulations** is **triangulations with no hanging vertices**.

The remarkable fact about conforming triangulations is the following

Theorem 2.2. *Let $\mathcal{T}$ be a conforming triangulation with n_V vertices. Given arbitrary scalar values $f_1, f_2, \ldots, f_{n_V}$ at the vertices, then*

- *the piecewise $\mathbb{P}_1$ function f defined by (2.28) is* **always continuous**, *and*
- *every function in the space $\mathcal{W}_h$ of all continuous piecewise $\mathbb{P}_1$ functions corresponds to a specific set of vertex values $(f_1, f_2, \ldots, f_{n_V}) \in \mathbb{R}^{n_V}$.*

As a consequence, the set of functions $\{N_a,\ a = 1, \ldots, n_V\}$, each one constructed as in (2.28), in which N_a takes the value 1 at vertex a and the value zero at all other vertices, **is a basis of** $\mathcal{W}_h$.

If the position of vertex i is $x_i = (x_{i1}, x_{i2})$, then the **Krönecker-delta property** holds, i.e.,

$$N_j(x_i) = \delta_{ij} \tag{2.29}$$

with $\delta_{ij} = 1$ if $i = j$ and $= 0$ otherwise.

Any function w_h in $\mathcal{W}_h$ is a linear combination of the basis functions. The coefficients of the linear combination, or **degrees of freedom**, are the values at the vertices, which by such property are called **nodes of the P_1 finite element space**. We thus have $n_V = m$ and

$$w_h(x) = w_1 N_1(x) + w_2 N_2(x) + \ldots + w_m N_m(x)\,, \tag{2.30}$$

where

$$w_1 = w_h(x_1), \quad w_2 = w_h(x_2), \qquad \ldots \qquad w_m = w_h(x_m)\,. \tag{2.31}$$

Once more we observe the linear correspondence between the **column array** of nodal values $\mathsf{W} = (w_1, w_2, \ldots, w_m)^T$ and the **function** $w_h(x)$.

How is a triangulation handled within a code? The typical way in which triangulations are handled is by means of two basic arrays:

- The **vertex coordinates array**, denoted here by X. It is a matrix of n_V columns, such that each column is the coordinate vector of a vertex.
- The **list of vertices**, denoted here by LV. It is a matrix of n_T columns and 3 rows. Each column contains the three numbers identifying the three vertices of the corresponding triangle. This is also called the **connectivity array**.

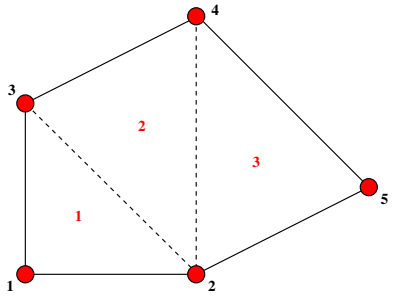

Figure 2.12 A simple conforming triangulation

Example 2.8 (A simple conforming triangulation) The triangulation of Fig. 2.12 has $n_V = 5$ vertices and $n_T = 3$ triangles and is conforming, as can easily be checked. With the specified numbering of triangles and vertices, the corresponding arrays are

$$\mathsf{X} = \begin{pmatrix} 4 & 8 & 4 & 8 & 12 \\ 2 & 2 & 6 & 8 & 4 \end{pmatrix} \tag{2.32}$$

$$\mathsf{LV} = \begin{pmatrix} 1 & 3 & 4 \\ 2 & 2 & 2 \\ 3 & 4 & 5 \end{pmatrix} \tag{2.33}$$

The basis functions $N_1, \ldots, N_5$ of the P_1 space corresponding to this triangulation can be seen in Fig. 2.13.

2.4.1.1 Barycentric or Area Coordinates

The **geometry** of a P_1 triangle K is determined by the positions its vertices $\mathbf{X}^1$, $\mathbf{X}^2$ and $\mathbf{X}^3$. It is the **only** triangle that has such vertices. It is also the **convex hull** of $\{\mathbf{X}^1, \mathbf{X}^2, \mathbf{X}^3\}$, defined as the convex linear combinations (CLC) of the vertex positions:

$$K = \mathscr{C}(\{\mathbf{X}^1, \mathbf{X}^2, \mathbf{X}^3\}) = \left\{ \sum_{j=1}^{d+1} \lambda_j \mathbf{X}^j \;\middle|\; \underbrace{0 \le \lambda_j \le 1 \;\forall j, \text{ and } \sum_{j=1}^{d+1} \lambda_j = 1}_{\hat{K}} \right\} \tag{2.34}$$

The set $\hat{K}$ defines a triangle, called **reference triangle**, illustrated in Fig. 2.14.
Remark: This definition of the geometry, in fact, works equally well in 2D ($d = 2$) and 3D ($d = 3$, in which case the triangle turns into a tetrahedron). It is independent of d. For each $\mathbf{x} \in K$ there exists a unique triplet $(\lambda_1, \lambda_2, \lambda_3) \in \hat{K}$ such that $\mathbf{x} = \sum_{j=1}^{3} \lambda_j \mathbf{X}^j$, thus

$$\mathbf{x} \in K \iff (\lambda_1, \lambda_2, \lambda_3) \in \hat{K}$$

is one-to-one, and thus a **reparameterization** (change of coordinates) of K. The parameters λ_i are called **area coordinates** of K (also **barycentric coordinates**).

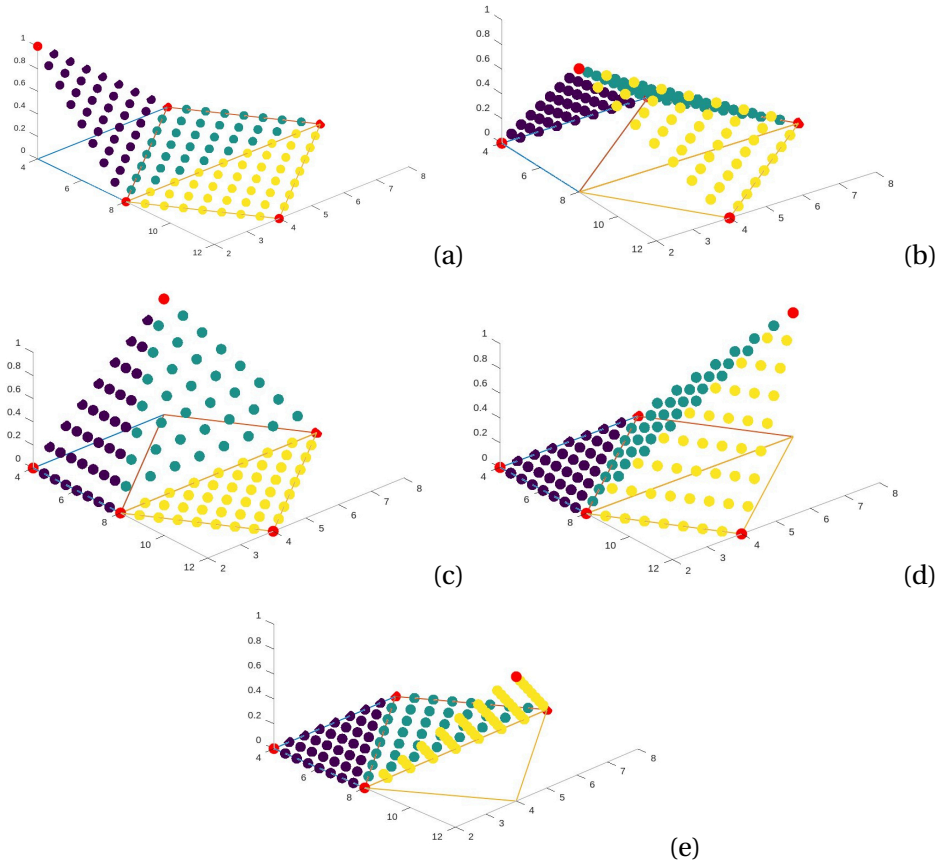

Figure 2.13 Basis functions of the triangulation of Example 2.8. (a) N_1, (b) N_2, ..., (e) N_5. The elevation corresponds to the value of the function. The nodal values are shown as red dots. Sample points belonging to triangle 1, 2 or 3 are shown in purple, green or yellow, respectively.

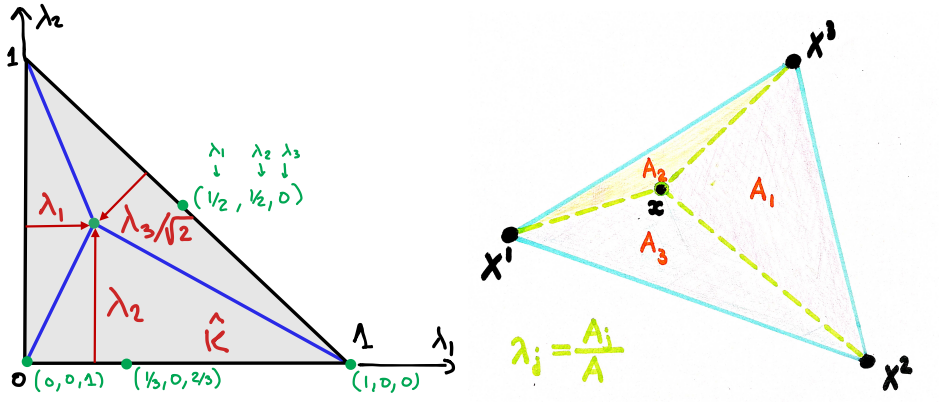

Figure 2.14 Left: Reference triangle $\hat{K}$ and domain of the barycentric coordinates. Right: Interpretation of barycentric coordinates in a general triangle as the fraction of the area of the triangle formed by a point x in the triangle and two of the vertices.

Barycentric coordinates have the following properties:

a) $\mathbf{x} = \mathbf{X}^j \iff \lambda_j = 1, \lambda_{k\neq j} = 0$. This is analogous to the Krönecker delta property.

b) $\mathbf{x}$ belongs to edge $\mathbf{X}^i\mathbf{X}^j$ iff $\lambda_i + \lambda_j = 1$, and as a result, $\lambda_{k\notin\{i,j\}} = 0$. In other words, λ_j is zero along the edge opposite to vertex $\mathbf{X}^j$.

c) $\mathbf{x}$ belongs to the (relative) interior of K iff all λ_j's are different from 0 and 1.

d) The barycentric coordinates satisfy that

$$\lambda_i = \frac{A_i}{A},$$

where A is the area of the triangle K and A_i is the area of the triangle formed by $\mathbf{x}$ and the two vertices $\mathbf{X}^j$ with $j \neq i$, see Fig. 2.14.

e) The inverse mapping to $(\lambda_1, \lambda_2, \lambda_3) \mapsto \mathbf{x} = \sum_{j=1}^3 \lambda_j \mathbf{X}^j$ is given by (in 2D, with $\mathbf{x} = (x_1, x_2)^T$)

$$\lambda_1(x_1, x_2) = \frac{1}{2A}\left[-(X_2^3 - X_2^2)(x_1 - X_1^2) + (X_1^3 - X_1^2)(x_2 - X_2^2)\right] \quad (2.35)$$

$$\lambda_2(x_1, x_2) = \frac{1}{2A}\left[-(X_2^1 - X_2^3)(x_1 - X_1^3) + (X_1^1 - X_1^3)(x_2 - X_2^3)\right] \quad (2.36)$$

$$\lambda_3(x_1, x_2) = \frac{1}{2A}\left[-(X_2^2 - X_2^1)(x_1 - X_1^1) + (X_1^2 - X_1^1)(x_2 - X_2^1)\right] \quad (2.37)$$

where $2A$ is twice the area of K,

$$2A = (X_1^2 - X_1^1)(X_1^3 - X_1^1) - (X_2^2 - X_2^1)(X_2^3 - X_2^1)\,.$$

It is a general convention that the *vertices are ordered either clockwise or counter-clockwise*. We are adopting the latter. Otherwise, A would be negative the area of K, but the other formulae would remain true.

f) The **barycenter B** of K corresponds to

$$\mathbf{B} = \frac{\mathbf{X}^1 + \mathbf{X}^2 + \mathbf{X}^3}{3} \leftrightarrow (\lambda_1, \lambda_2, \lambda_3) = \left(\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right)$$

g) The **edges midpoints** correspond to $(\lambda_1, \lambda_2, \lambda_3)$ equal to $(0, 1/2, 1/2)$, $(1/2, 0, 1/2)$ and $(1/2, 1/2, 0)$. Notice that the midpoints have been numbered according to the opposite vertex.

2.4.1.2 The P_1-Element and the Local-to-Global Map

The barycentric coordinates are not just another set of coordinates (instead of $x_1 - x_2$) that one could choose to parameterize the points of a triangle, for **triangular finite elements**,

$$N_1^e = \lambda_1, \quad N_2^e = \lambda_2 \quad \text{and} \quad N_3^e = \lambda_3, \qquad (2.38)$$

given by (2.35)-(2.37) above, are **the local basis of the P_1-finite element space** restricted to element e. In fact, they are **three polynomials of degree 1 and linearly independent**, and thus a basis of $\mathbb{P}_1$. Since they satisfy the Krönecker-delta property at the vertices, the **vertices are the nodes of this space**.

We can then define a P_1-element as $e = (\Omega_e, \{N_1^e, N_2^e, N_3^e\})$, where Ω_e is a triangle.

The gradient of the basis functions can be obtained by differentiation of (2.35)-(2.37) with respect to x_1 and x_2, which gives

$$\nabla N_1^e = \frac{1}{2A}\begin{pmatrix} X_2^2 - X_2^3 \\ X_1^3 - X_1^2 \end{pmatrix}, \tag{2.39}$$

$$\nabla N_2^e = \frac{1}{2A}\begin{pmatrix} X_2^3 - X_2^1 \\ X_1^1 - X_1^3 \end{pmatrix}, \tag{2.40}$$

$$\nabla N_3^e = \frac{1}{2A}\begin{pmatrix} X_2^1 - X_2^2 \\ X_1^2 - X_1^1 \end{pmatrix}. \tag{2.41}$$

The second derivatives are of course zero all over the element.

Now, **notice what happens if we take the local-to-global array equal to the list-of-vertices array**,

$$\mathsf{LG} = \mathsf{LV}\,. \tag{2.42}$$

This means that we consider the P_1 element with **the vertices as nodes and the triangles as element domains**.

Following exactly the same methodology that was developed for the 1D case, the global basis functions are defined as

$$N_A(x_1, x_2) = \mathop{\mathring{\sum}}_{\{(a,e)|\mathsf{LG}(a,e)=A\}} N_a^e(x_1, x_2)\,. \tag{2.43}$$

Remember, the summation is only performed when $x = (x_1, x_2)$ is **interior** to some triangle in the mesh. The value of N_A at the mesh vertices and edges is not equal to the sum above, but as the continuous extension (if it exists) from the element interiors. This is formalized in the following definition.

Definition 2.5 (Broken Sum)**.** *Let $\mathcal{T} = \{K_1, \ldots, K_{n_{el}}\}$ be a mesh for a domain $\Omega \subset \mathbb{R}^d$. Let $\mathcal{W}_h$ be a space of scalar-valued functions over Ω such that if $f_h \in \mathcal{W}_h$, then f_h is continuous in the interior of each element domain. The broken sum $\mathring{+}: \mathcal{W}_h \times \mathcal{W}_h \to \mathcal{W}_h$ is defined for $f_h, g_h \in \mathcal{W}_h$ by*

$$(f_h \mathring{+} g_h)(x) = f_h(x) + g_h(x), \qquad x \in \bigcup_{i=1}^{n_{el}} \mathring{K}_i \tag{2.44a}$$

and

$$(f_h \mathring{+} g_h)(x) = \lim_{y \to x} f_h(y) + g_h(y), \qquad \textit{otherwise.} \tag{2.44b}$$

Let us see how this works in the triangulation of Figure 2.12.

Example 2.9 (Using the list of vertices as local-to-global map) Going back to the triangulation in Example 2.8, with LG equal to LV given in (2.33), i.e.,

$$\mathsf{LG} = \mathsf{LV} = \begin{pmatrix} 1 & 3 & 4 \\ 2 & 2 & 2 \\ 3 & 4 & 5 \end{pmatrix}, \tag{2.45}$$

the explicit expressions for the global basis functions (corresponding to (2.43)) are

$$N_1 = N_1^1 \tag{2.46}$$
$$N_2 = N_2^1 \mathring{+} N_2^2 \mathring{+} N_2^3 \tag{2.47}$$
$$N_3 = N_3^1 \mathring{+} N_1^2 \tag{2.48}$$
$$N_4 = N_3^2 \mathring{+} N_1^3 \tag{2.49}$$
$$N_5 = N_3^3 \tag{2.50}$$

Take for example N_3, which is depicted in Fig. 2.15. We know that N_3 is different from zero just in elements 1 and 2 because in LG the number 3 only appears in columns 1 and 2. Inside element $e = 1$ the function N_3 coincides with the N_3^1, depicted in purple in the figure, because 3 appears in row 3 of column $e = 1$ of LG. Similarly, in element $e = 2$ the function N_3 coincides with N_1^2, depicted in green in the figure, because 3 appears in row 1 of column $e = 2$ of LG. In column $e = 3$ of LG the number 3 does not appear. This means that the function N_3 is identically zero in element $e = 3$, as shown in yellow in Fig. 2.15. By the magic of conforming triangulations, the three pieces fit together in such a way that the resulting function N_3 is a continuous function. **In fact, the functions N_1-N_5 defined by (2.46)-(2.46) are exactly the same as those defined in Theorem 2.2 and depicted in Figure 2.13.**

The fundamental message is that, if the arrays X and LV of a conforming triangulation of a domain Ω are available, then the global basis functions obtained from (2.43) by taking LG = LV generate the space $\mathcal{W}_h$ of P_1 continuous finite elements.

2.4.2 Elemental Computations with the Simplest C^0 Space

In what follows, we proceed and use the basis functions just developed to solve a 2D diffusion problem in any polygonal domain by a variational numerical method.

2.4.2.1 The Element Stiffness Matrix

Remember that our goal is to solve the diffusion Problem 2.2 based on variational equation (2.21) with $\mathcal{W}_h$ a finite element space over a conforming triangulation

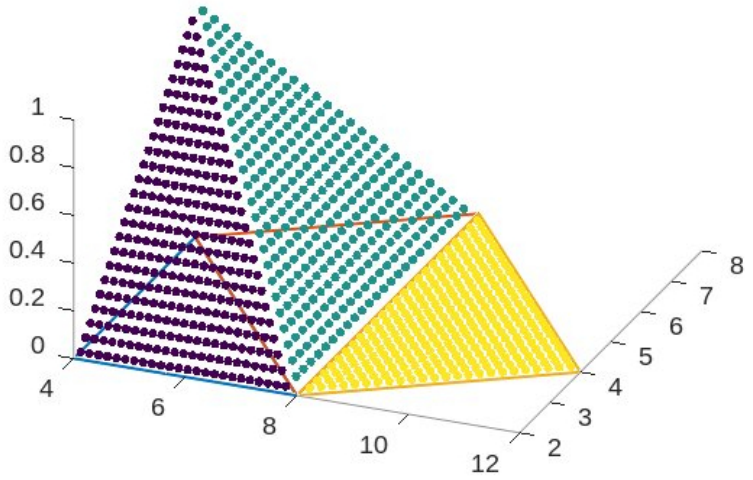

Figure 2.15 The basis function N_3 sampled only at interior points. Sample points belonging to triangle 1, 2 or 3 are shown in purple, green or yellow, respectively.

for Ω, so as to obtain the only function $u_h \in \mathscr{S}_h$ that satisfies

$$\int_\Omega (K\nabla u_h)\cdot\nabla v_h \, d\Omega = \int_\Omega f\, v_h \, d\Omega + \int_{\partial\Omega_N} H\, v_h \, d\Gamma \tag{2.51}$$

for all $v_h \in \mathscr{V}_h$.

The element stiffness matrix of the diffusion problem for the P_1 triangular element is as follows. Let the indices a and b run from 1 to 3, the number of nodes per element. The three local basis functions at element e are barycentric coordinates, as introduced in (2.38). As a consequence, the element matrix results from

$$\mathsf{K}^e_{ab} = \int_{\Omega_e} (K\nabla N^e_b)\cdot\nabla N^e_a \, d\Omega \,. \tag{2.52}$$

Let us assume for simplicity that K is isotropic, so that $K(x) = k(x)\mathbb{I}$. Further, notice that ∇N^e_b is constant over Ω_e for any b, so that

$$\mathsf{K}^e_{ab} = \left(\int_{\Omega_e} k(x)\, d\Omega\right) \nabla N^e_b\cdot\nabla N^e_a = k_e\, A_e \nabla N^e_b\cdot\nabla N^e_a \,.$$

Notice that k_e is the average diffusion coefficient in the element, since A_e is the element's area.

☞ The matrix dN has the form

$$\mathsf{dN} = \begin{bmatrix} \frac{\partial N^e_1}{\partial x_1} & \frac{\partial N^e_2}{\partial x_1} & \frac{\partial N^e_3}{\partial x_1} \\ \frac{\partial N^e_1}{\partial x_2} & \frac{\partial N^e_2}{\partial x_2} & \frac{\partial N^e_3}{\partial x_2} \end{bmatrix}.$$

Consider, for each element e, the array dN which contains the partial derivatives of the basis functions and is defined by

$$\mathsf{dN}_{ib} = \frac{\partial N^e_b}{\partial x_i} \,,$$

that is, one column per basis function, with $i = 1,2$. Then K^e is given by

$$\mathsf{K}^e = k_e\, A_e\, \mathsf{dN}^T\, \mathsf{dN} \,.$$

In Octave/MATLAB code, given the array of nodal coordinates of element e,

$$\mathsf{xe} = \begin{pmatrix} X_1^1 & X_1^2 & X_1^3 \\ X_2^1 & X_2^2 & X_2^3 \end{pmatrix}$$

in which each column corresponds to one of the three nodes of the element, and given k_e, the computation of K^e is as simple as

```
dN=[xe(2,2)-xe(2,3),xe(2,3)-xe(2,1),xe(2,1)-xe(2,2);...
      xe(1,3)-xe(1,2),xe(1,1)-xe(1,3),xe(1,2)-xe(1,1)];
Ae2=dN(2,3)*dN(1,2)-dN(1,3)*dN(2,2);
dN=dN/Ae2;
Ke=Ae2/2*ke*dN'*dN;
```

2.4.2.2 The Element Load Vector

Let us for now assume that either Dirichlet conditions are imposed all over the boundary or the Neumann datum H is zero. Then the element load vector is

$$\mathsf{F}_a^e = \int_{\Omega_e} f\, N_a^e \, d\Omega \,.$$

The result of this integral depends of course on the function $f(x)$. Let us assume that $f = f_e$ is constant over the element, then $f N_a^e$ is a polynomial of degree 1 in x_1 and x_2 which has as integral

$$\mathsf{F}_a^e = f_e \int_{\Omega_e} N_a^e \, d\Omega = \frac{f_e A_e}{3}, \qquad \forall\, a = 1,2,3. \tag{2.53}$$

The computation of the element load vector is thus immediate

```
Fe=Ae2*fe*ones(3,1)/6;
```

Another possibility is to assume that f is affine within the element, in which case the three nodal values need to be provided. Whatever the values are, it is clear that the product $f\, N_a^e$ is a polynomial of degree 2 in the variables x_1 and x_2.

To compute F^e we can use the following well-known integration rule: *The integral of a quadratic polynomial over a triangle is equal to $\frac{1}{3}$ times the triangle's area times the sum of the values of the polynomial at the three edge midpoints.*

Let $f_e = (f_{e,1}, f_{e,2}, f_{e,3})$ be an array containing the three nodal values of f_e. Then the integration rule above states that

$$\begin{aligned} \mathsf{F}_a^e &= \int_{\Omega_e} f\, N_a^e \, d\Omega \\ &= \frac{A_e}{3}\left(\frac{f_{e,1}+f_{e,2}}{2}\,\frac{\delta_{a1}+\delta_{a2}}{2} + \frac{f_{e,2}+f_{e,3}}{2}\,\frac{\delta_{a2}+\delta_{a3}}{2} + \frac{f_{e,3}+f_{e,1}}{2}\,\frac{\delta_{a3}+\delta_{a1}}{2} \right) \end{aligned}$$

Particularizing for $a = 1,2,3$ we obtain the element load vector

$$\mathsf{F}^e = \frac{A_e}{12}\begin{pmatrix} 2f_{e,1}+f_{e,2}+f_{e,3} \\ f_{e,1}+2f_{e,2}+f_{e,3} \\ f_{e,1}+f_{e,2}+2f_{e,3} \end{pmatrix} \tag{2.54}$$

which would lead to the code (notice that `fe` is stored as a row array)

```
auxmat=[2,1,1;1,2,1;1,1,2];
Fe=(Ae2/24)*auxmat*fe';
```

2.4.3 Solving Problems with Dirichlet Boundaries

Let us work out the procedure and code that solves the diffusion problem with P_1 finite elements, assuming that we only have Dirichlet boundary conditions. We assume, as before, that a mesh is provided by means of a **list of coordinates** X and a **list of vertices** LV. This enables us to construct $\mathcal{W}_h$ as the continuous P_1 finite element space over it.

The trial and test spaces are then constructed as

$$\tilde{\mathcal{S}}_h = \{w_h \in \mathcal{W}_h \mid w_h(x) = g(x) \quad \forall x \in \partial\Omega_D\}$$
$$\tilde{\mathcal{V}}_h = \{w_h \in \mathcal{W}_h \mid w_h(x) = 0 \quad \forall x \in \partial\Omega_D\}.$$

As in §1.3.1.1, the trial space is set to be the subset of functions in $\mathcal{W}_h$ that satisfy the essential boundary conditions, and the test space is the direction of the trial space. There is a caveat here though: functions in $\mathcal{W}_h$ are affine on each edge in $\partial\Omega$, but g may not be! As a result, the space $\tilde{\mathcal{S}}_h$ may have *no* functions in it...

Fortunately, it is not necessary to satisfy essential boundary conditions exactly for convergence, but it is enough to approximate them. A suitable way to do it in this case is by requiring w_h to be equal to g at each vertex (and hence node for P_1 element) on $\partial\Omega$, namely,

$$\mathcal{S}_h = \{w_h \in \mathcal{W}_h \mid w^a = g(\mathbf{X}^a) \text{ for each vertex } \mathbf{X}^a \in \partial\Omega_D\}$$
$$\mathcal{V}_h = \{w_h \in \mathcal{W}_h \mid w^a = 0 \text{ for each vertex } \mathbf{X}^a \in \partial\Omega_D\}$$

Once more, we defined the test space $\mathcal{V}_h$ as the direction of $\mathcal{S}_h$. Notice, however, that $\mathcal{V}_h = \tilde{\mathcal{V}}_h$.

Therefore, we can assume that a **list of boundary nodes**, denoted by η_g is provided, together with a **list of boundary values**, GG.

Also, we can approximate k and f as piecewise constant, with value k_e and f_e in each triangle.

Then we can build a function that computes the elementary stiffness matrix and load vector, for example:

```
function [Ke, Fe]=elementKandF(xe,ke,fe)
  dN=[xe(2,2)-xe(2,3),xe(2,3)-xe(2,1),xe(2,1)-xe(2,2);...
      xe(1,3)-xe(1,2),xe(1,1)-xe(1,3),xe(1,2)-xe(1,1)];
  Ae2=dN(2,3)*dN(1,2)-dN(1,3)*dN(2,2);
  dN=dN/Ae2;
  Ke=Ae2/2*ke*dN'*dN;
  Fe=Ae2*fe*ones(3,1)/6;
end
```

It only remains to **assemble** the element contributions and impose the Dirichlet boundary conditions to end up with the global stiffness matrix and load vector.

Here we will use a trick that simplifies the coding: Instead of taking care of the boundary conditions during the assembly procedure, we will *assemble the global*