belongs to $\mathscr{P}^e$. Its components in the local basis are

$$\phi_a^e = u_{\mathsf{LG}(a,e)}. \tag{1.121}$$

The function f^e is called the **restriction** of u to element e.

So each set of values for the global degrees of freedom define a set of values for the local degrees of freedom to describe the same function over an element. The process of obtaining the local degrees of freedom from the global ones through (1.121) is called **localization**.

Map between local and global degrees of freedom (1.121)**.**

Consider $u \in \mathscr{W}_h$, then

$$\begin{aligned} u(x) &= \sum_{A=1}^{m} u_A N_A(x) \\ &= \sum_{A=1}^{m} u_A \sum_{\{(a,e)|\mathsf{LG}(a,e)=A\}}^{\circ} N_a^e(x) && \text{from (1.116)} \\ &= \sum_{A=1}^{m} \sum_{\{(a,e)|\mathsf{LG}(a,e)=A\}}^{\circ} u_{\mathsf{LG}(a,e)} N_a^e(x) && \text{from definition of } \mathsf{LG} \\ &= \sum_{e=1}^{\overset{\circ}{n_{\mathrm{el}}}} \sum_{a=1}^{\overset{\circ}{k}} u_{\mathsf{LG}(a,e)} N_a^e(x) && \text{see below} \end{aligned}$$

The last step uses the fact that in spanning all values of A with the first sum, the two sums together effectively guarantee that all pairs (a,e) will be added exactly once, since $\{1,\ldots,m\}$ is precisely the range of LG, so its pre-image is the entire domain. This is again a consequence of the fact that every shape function contributes to exactly one global basis function, and all global basis functions are built in this way. It is also a consequence of defining global basis functions as sums of shape functions. Had global basis functions been defined as more general linear combinations of shape functions, each local degree of freedom would not be directly equal to a global degree of freedom.

So, because of the construction of the basis functions, it follows that: (a) the function u restricted to element e belongs to $\mathscr{P}^e$, (b) the values of the degrees of freedom of u restricted to element e are $\phi_a^e = u_{\mathsf{LG}(a,e)}$, so the local-to-global map also maps the local degrees of freedom to local ones.

1.4.4 Assembly of the Stiffness Matrix and Load Vector

The computation of the stiffness matrix and load vector generally involves the calculation of integrals over the domain, such as those involved in the bilinear form and linear functional. For example, for model Problem 1.3 with $b(x) = 0$,

$$a(u,v) = \int_\Omega \left[k(x)u'(x)v'(x) + c(x)u(x)v(x) \right] dx, \tag{1.122a}$$

$$\ell(v) = k(L)d_L v(L) + \int_\Omega f(x)v(x)\, dx. \tag{1.122b}$$

This task highlights key functions that elements provide: The decomposition of the domain into elements afford us the ability to decompose integrals over the domain as a sum of integrals over elements, construct *elemental stiffness matrices* and *elemental load vectors*, and *"assemble"* them over the mesh to form the stiffness matrix and load vector of the problem.

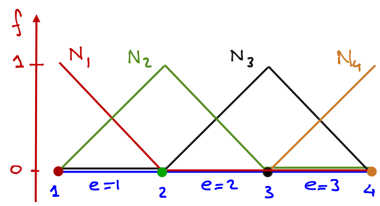


Figure 1.18

Let's have a brief look at the main ideas of what we will be discussing. Consider again the simplest space of continuous functions over a mesh with 3 elements and basis functions in Fig. 1.18. If, for example, we wanted to compute the stiffness matrix entry $K_{33} = a(N_3, N_3)$, we could split the integral in (1.122a) as

$$
\begin{aligned}
K_{33} &= \int_{\Omega} \left[k(x) N_3'(x) N_3'(x) + c(x) N_3(x) N_3(x) \right] \, dx \\
&= \underbrace{\int_{\Omega_2} \left[k(x) \left(N_2^2\right)'(x) \left(N_2^2\right)'(x) + c(x) N_2^2(x) N_2^2(x) \right] \, dx}_{K_{22}^2} \\
&\quad + \underbrace{\int_{\Omega_3} \left[k(x) \left(N_1^3\right)'(x) \left(N_1^3\right)'(x) + c(x) N_1^3(x) N_1^3(x) \right] \, dx}_{K_{11}^3}
\end{aligned}
\tag{1.123}
$$

There is no need to compute an integral over element $e = 1$, given that $N_3(x) = 0$ for $x \in \Omega_1$ and hence the value of the integral is zero. Only elements 2 and 3 contribute non-zero values to K_{33}. The contribution of each element, K_{22}^2 and K_{11}^3, are entries of the element stiffness matrices for elements 2 and 3. Over each element we can replace the global basis functions (N_3) by the local ones (N_2^2 and N_1^3), or shape functions. The value of K_{33} is obtained by *accumulating* the contributions to its value by all elements in the mesh. This is what is called *assembling* K_{33}.

For problems with two and three-dimensional domains, computing in this way simplifies the integration problem enormously, since it is only necessary to learn how to compute integrals over each element, rather than over arbitrarily-shaped domains.

Element Stiffness Matrix and Element Load Vector. The element stiffness matrix K^e and element load vector F^e are inspired and emerge from the decomposition of the integrals involved in the definition of the bilinear form and linear functional as sums of integrals over elements. Symbolically, we can write

$$
\int_{\Omega} (\cdot) = \sum_{e=1}^{n_{\mathrm{el}}} \int_{\Omega_e} (\cdot), \tag{1.124}
$$

where Ω is the domain over which an integral is performed. Many commonly found bilinear forms and linear functionals can be written as

$$
\begin{aligned}
a(u, v) &= \sum_{e=1}^{n_{\mathrm{el}}} a^e(u, v) \quad \text{where} \quad a^e(u, v) &= \int_{\Omega_e} \dots \, dx \\
\ell(v) &= \sum_{e=1}^{n_{\mathrm{el}}} \ell^e(v) \quad \text{where} \quad \ell^e(v) &= \int_{\Omega_e} \dots \, dx,
\end{aligned}
\tag{1.125}
$$

For model Problem 1.3 with $b(x) = 0$ and $d_L = 0$ (c.f. (1.122)) this is

$$\begin{aligned} a(u,v) &= \sum_{e=1}^{n_{\text{el}}} \underbrace{\int_{\Omega_e} \left[k(x)u'(x)v'(x) + c(x)u(x)v(x)\right] dx}_{=a^e(u,v)} = \sum_{e=1}^{n_{\text{el}}} a^e(u,v), \\ \ell(v) &= \sum_{e=1}^{n_{\text{el}}} \underbrace{\int_{\Omega_e} f(x)v(x)\, dx}_{\ell^e(v)} = \sum_{e=1}^{n_{\text{el}}} \ell^e(v). \end{aligned} \tag{1.126}$$

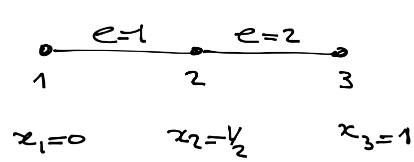
We are now in position to define K^e and F^e as

$K^e_{ab} = a^e(N^e_b, N^e_a)$	Element Stiffness Matrix	(1.127a)
$F^e_a = \ell^e(N^e_a)$	Element Load Vector	(1.127b)

for any $a, b = 1, \dots, k$. Before we discuss how these are used to construct K and F, let's look at an example.

The definition of the element stiffness matrix and element load vector is also notable for what it does not define: the values of $a^e(N^{e_1}_b, N^{e_2}_a)$ and $\ell^e(N^{e_1}_a)$, for elements e, e_1 and e_2 with $e \neq e_1$ and $e \neq e_2$. The reason for this is that if either $e_1 \neq e$ or $e_2 \neq e$, then both values are identically zero, given that either $N^{e_1}_b$ and/or $N^{e_2}_a$ are zero in Ω_e. Since the values of any basis function N_A in Ω_e are exclusively defined by linear combinations of shape functions in an element, these are the only potentially non-zero contributions to the stiffness matrix or load vector, and hence the only ones included in the element stiffness matrix and element load vector. Therefore, out of the $m \times m$ combinations of basis functions (generally a large number), the only non-zero contributions of $a^e(N_A, N_B)$ are accounted for by the $k \times k$ elemental stiffness matrix (generally a much smaller number).

Example 1.71 We compute the element stiffness matrix K^e and element load vector F^e defined by (1.126) for every element in a simple case. Consider a mesh with two P_1 elements over the interval $[0,1]$, and let $f(x) = 10$, $k(x) = 1$ and $c(x) = 3x$ for $x \in (0,1)$.



We do not yet specify the local-to-global map LG because it is not needed to compute K^e and F^e; it will be needed later to build K and F.

Since we have two elements, we need to compute K^1, K^2, F^1 and F^2. From (1.126) and the values for k, c and f,

$$a^e(u,v) = \int_{\Omega_e} \left[u'(x)v'(x) + 3xu(x)v(x)\right] dx, \qquad \ell^e(v) = \int_e 10v\, dx. \tag{1.128}$$

The shape functions over each element of this finite element mesh are

$$N_1^1(x) = \frac{1/2 - x}{1/2}$$
$$N_2^1(x) = \frac{x}{1/2}$$
$$N_1^2(x) = \frac{1 - x}{1/2}$$
$$N_2^2(x) = \frac{x - 1/2}{1/2}.$$

Notice here the superindex N_a^e with $e = 1,2$ is the element index, and not exponentiation. To simplify the notation, we will also use $N_{,x}$ to indicate the derivative of N, instead of N'.

The element stiffness matrices follow as:

$$K_{ab}^1 = \int_0^{1/2} N_{a,x}^1 N_{b,x}^1 + 3x N_a^1 N_b^1 \, dx, \qquad K_{ab}^2 = \int_{1/2}^1 N_{a,x}^2 N_{b,x}^2 + 3x N_a^2 N_b^2 \, dx.$$

This results in

$$K^1 = \begin{bmatrix} \int_0^{1/2} \left(-\frac{1}{1/2}\right)\left(-\frac{1}{1/2}\right) + 3x \frac{1/2-x}{1/2}\frac{1/2-x}{1/2} \, dx & \int_0^{1/2} \left(-\frac{1}{1/2}\right)\left(\frac{1}{1/2}\right) + 3x \frac{1/2-x}{1/2}\frac{x}{1/2} \, dx \\ \int_0^{1/2} \left(\frac{1}{1/2}\right)\left(-\frac{1}{1/2}\right) + 3x \frac{x}{1/2}\frac{1/2-x}{1/2} \, dx & \int_0^{1/2} \left(\frac{1}{1/2}\right)\left(\frac{1}{1/2}\right) + 3x \frac{x}{1/2}\frac{x}{1/2} \, dx \end{bmatrix}$$
$$= \begin{bmatrix} \frac{33}{16} & -\frac{31}{16} \\ -\frac{31}{16} & \frac{35}{16} \end{bmatrix},$$
$$K^2 = \begin{bmatrix} \int_{1/2}^1 \left(-\frac{1}{1/2}\right)\left(-\frac{1}{1/2}\right) + 3x \frac{1-x}{1/2}\frac{1-x}{1/2} \, dx & \int_{1/2}^1 \left(-\frac{1}{1/2}\right)\left(\frac{1}{1/2}\right) + 3x \frac{1-x}{1/2}\frac{x-1/2}{1/2} \, dx \\ \int_{1/2}^1 \left(\frac{1}{1/2}\right)\left(-\frac{1}{1/2}\right) + 3x \frac{x-1/2}{1/2}\frac{1-x}{1/2} \, dx & \int_{1/2}^1 \left(\frac{1}{1/2}\right)\left(\frac{1}{1/2}\right) + 3x \frac{x-1/2}{1/2}\frac{x-1/2}{1/2} \, dx \end{bmatrix}$$
$$= \begin{bmatrix} \frac{37}{16} & -\frac{29}{16} \\ -\frac{29}{16} & \frac{39}{16} \end{bmatrix}.$$

The element load vectors are:

$$\begin{aligned} F_a^1 &= \int_0^{1/2} 10 N_a^1 \, dx \quad \Longrightarrow F^1 = \begin{bmatrix} \int_0^{1/2} 10 \frac{1/2-x}{1/2} \, dx \\ \int_0^{1/2} 10 \frac{x}{1/2} \, dx \end{bmatrix} = \begin{bmatrix} 5/2 \\ 5/2 \end{bmatrix}, \\ F_a^2 &= \int_{1/2}^1 10 N_a^2 \, dx \quad \Longrightarrow F^2 = \begin{bmatrix} \int_{1/2}^1 10 \frac{1-x}{1/2} \, dx \\ \int_{1/2}^1 10 \frac{x-1/2}{1/2} \, dx \end{bmatrix} = \begin{bmatrix} 5/2 \\ 5/2 \end{bmatrix}. \end{aligned} \tag{1.129}$$

To encompass the most common types of variational equations, we further need to consider load vectors that have contributions from evaluating test functions on the boundary of the domain. An example of such contribution is found in (1.122b), in the term

$$k(L) d_L v(L).$$

This term cannot be written as an integral over an element, and it involves the value of the test function v at the end of the interval, $x = L$. When terms of this type are present, the linear form of the problem is written as

$$\ell(v) = \sum_{e=1}^{n_{\mathrm{el}}} \ell^e(v) + h_0 v(0) + h_L v(L), \tag{1.130}$$

where h_0 and h_L are two real numbers defined by the bilinear form. For our example in (1.122b), $h_0 = 0$ and $h_L = k(L)\,d_L$.

For simplicity, we will incorporate the potential contributions of the two boundary terms into the element load vector of the first and last elements of the mesh. Specifically, we modify the definition (1.127a) of the element load vector for $e = 1$ and $e = n_{\text{el}}$ to become

$$\begin{aligned} F_a^1 &= \ell^1(N_a^1) + h_0 N_a^1(0) && \text{Element load vector for } e = 1 \\ F_a^e &= \ell^e(N_a^e) && \text{Element load vector for } e \neq 1, n_{\text{el}} \\ F_a^{n_{\text{el}}} &= \ell^1(N_a^{n_{\text{el}}}) + h_L N_a^{n_{\text{el}}}(L) && \text{Element load vector for } e = n_{\text{el}}, \end{aligned} \tag{1.131}$$

for $a = 1, \dots, k$.

Assembly. The construction of the stiffness matrix K and load vector F in terms of the ones from the elements is called the finite element **assembly** operation. It is also called the **Direct Stiffness Method**. It is a result of the way global basis functions are constructed (c.f. (1.113a)). Recall that basis functions in $\mathscr{W}_h$ can be written in terms of the shape functions, or local basis functions, as

$$N_A(x) = \mathop{\sum\nolimits^{\circ}}_{\{(a,e)\,|\,\mathsf{LG}(a,e)=A\}} N_a^e(x). \tag{1.132}$$

for $A \in \{1, \dots, m\}$. Based on this relation, we show below that

$$\boxed{F_A = \sum_{e=1}^{n_{\text{el}}} \sum_{\{a|\mathsf{LG}(a,e)=A\}} F_a^e, \qquad A \in \eta_a} \tag{1.133a}$$

and

$$\boxed{K_{AB} = \sum_{e=1}^{n_{\text{el}}} \sum_{\substack{\{a|\mathsf{LG}(a,e)=A\} \\ \{b|\mathsf{LG}(b,e)=B\}}} K_{ab}^e, \qquad A \in \eta_a, B \in \eta} \tag{1.133b}$$

These identities directly connect entries of the stiffness matrix and load vector in active indices' rows to entries in the corresponding contributions from the elements. Specifically, each such entry in K and F is obtained by accumulating the contributions of some elements in the mesh.

Example 1.72 We show next that (1.123) is a result of (1.133b). The local-to-global map for the mesh in Fig. 1.18 is (1.120), namely,

$$\mathsf{LG} = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 3 & 4 \end{bmatrix}.$$

To compute K_{33} from (1.133b), we need to identify the set

$$\{a \mid \mathsf{LG}(a,e) = 3\},$$

for each $e \in \{1,2,3\}$, since $A = B = 3$. By inspection of LG, it follows that

$$\{a \mid \mathsf{LG}(a,1) = 3\} = \emptyset,$$
$$\{a \mid \mathsf{LG}(a,2) = 3\} = \{2\},$$
$$\{a \mid \mathsf{LG}(a,3) = 3\} = \{1\}.$$

Replacing with these indices in (1.133b), we obtain

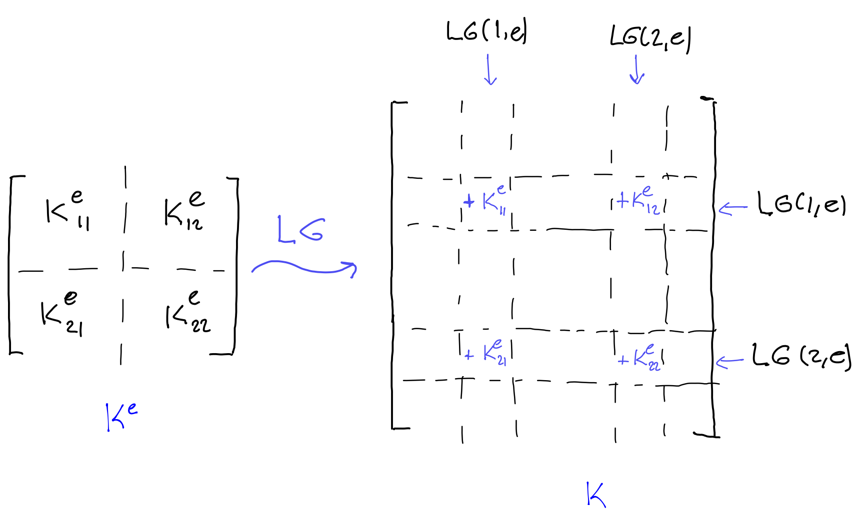
$$K_{33} = K^2_{22} + K^3_{11},$$

and we recover (1.123).

This example illustrates that computing a value of K_{AB} for a single pair of indices AB or the value of F_A for a single index A requires searching for those elements that contain indices of local degrees of freedom that are mapped to A and/or B by LG. However, since we want to compute K_{AB} and F_A for *all* active indices A and *all* indices B, it is more efficient to proceed in a different way:

1. Initially set $K = 0$ and $F = 0$.
2. Visit every element e in the mesh to compute K^e and F^e and
3. Add K^e_{ab} to $K_{\mathsf{LG}(a,e)\mathsf{LG}(b,e)}$ for all $a, b\{1,\ldots,k\}$ if $\mathsf{LG}(a,e) \in \eta_a$, and
4. Add F^e_a to $F_{\mathsf{LG}(a,e)}$ for all $a \in \{1,\ldots,k\}$ if $\mathsf{LG}(a,e) \in \eta_a$.

In this way, there is no need to search for which elements contribute to an entry. This is the distinguishing aspect of the assembly. A sketch of the way the local-to-global map defines where to add the element stiffness matrix is shown next:



Example 1.73 Let's revisit Example 1.71 to assemble a stiffness matrix and a load vector.

The mesh contains two elements, and to build the finite element space we need to specify the local-to-global map LG. For a space of continuous functions, this map is

$$\mathsf{LG} = \begin{bmatrix} 1 & 2 \\ 2 & 3 \end{bmatrix},$$

and the dimension of $\mathscr{W}_h$ is $m = 3$. Therefore, K is a 3×3 matrix, and F is a 3×1 matrix. Furthermore, we will assume that all indices are active [11], namely, $\eta_a = \eta = \{1, 2, 3\}$.

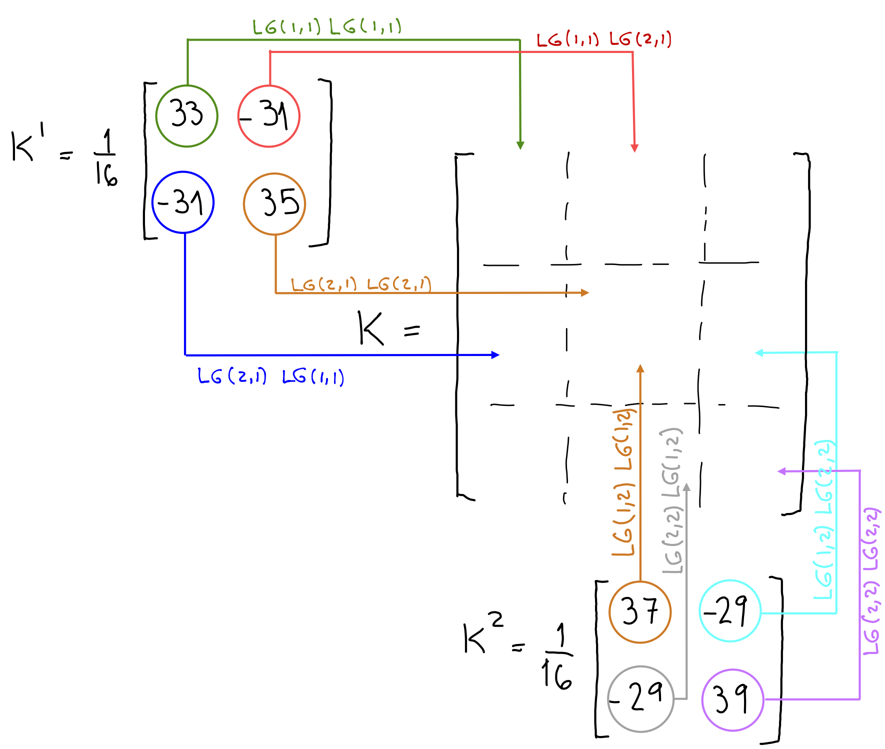
The element stiffness matrices are, from Example 1.71,

$$K^1 = \frac{1}{16} \begin{bmatrix} 33 & -31 \\ -31 & 35 \end{bmatrix} \qquad K^2 = \frac{1}{16} \begin{bmatrix} 37 & -29 \\ -29 & 39 \end{bmatrix},$$

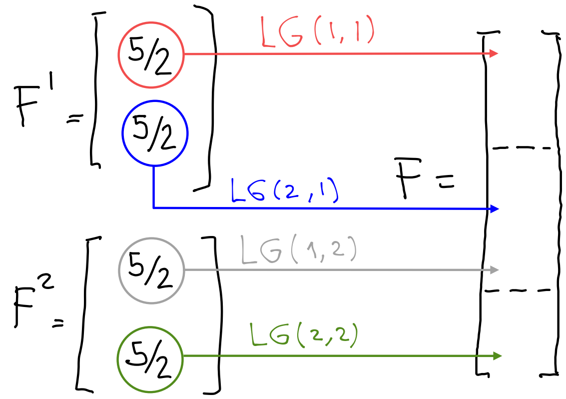
and the element load vectors are

$$F^1 = F^2 = \begin{bmatrix} 5/2 \\ 5/2 \end{bmatrix}.$$

The assembly process is sketched in the following figures



[11] this is the case when the boundary conditions are natural and *homogeneous*, i.e., equal to zero



The results of the assembly are

$$K = \frac{1}{16}\begin{bmatrix} 33 & -31 & 0 \\ -31 & 35+37 & -29 \\ 0 & -29 & 39 \end{bmatrix} = \frac{1}{16}\begin{bmatrix} 33 & -31 & 0 \\ -31 & 72 & -29 \\ 0 & -29 & 39 \end{bmatrix},$$

and

$$F = \begin{bmatrix} 5/2 \\ 5/2+5/2 \\ 5/2 \end{bmatrix} = \begin{bmatrix} 5/2 \\ 5 \\ 5/2 \end{bmatrix}. \tag{1.134}$$

Insofar we have discussed how to assemble the rows of K and F whose indices are active. Rows with constrained indices are still defined by (1.78c), i.e., for $A \in \eta_g$ and $B \in \eta$,

$$K_{AB} = \delta_{AB}, \qquad F_A = \overline{u}_A.$$

Taking this into account, the pseudocode for the assembly procedure is

```
K = 0,   F = 0
FOR e ∈ {1, ..., n_el}
    FOR a ∈ {1, ..., k}
        IF LG(a,e) ∈ η_a
            FOR b ∈ {1, ..., k}
                K(LG(a,e), LG(b,e)) += K^e_ab
            END FOR
            F(LG(a,e)) += F^e_a
        END IF
    END FOR
END FOR
FOR A ∈ η_g
    K(A,A) = 1
    F(A)   = ū_A
END FOR
```

Example 1.74 Let's modify Example 1.73 and assemble K and F by assuming that $\eta_a = \{2, 3\}$ and that $\overline{u}_1 = 4$, instead of the original assumption that all indices are active.

The results of the assembly are

$$K = \begin{bmatrix} 1 & 0 & 0 \\ -31/16 & 35/16 + 37/16 & -29/16 \\ 0 & -29/16 & 39/16 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ -31/16 & 72/16 & -29/16 \\ 0 & -29/16 & 39/16 \end{bmatrix},$$

and

$$F = \begin{bmatrix} 4 \\ 5/2 + 5/2 \\ 5/2 \end{bmatrix} = \begin{bmatrix} 4 \\ 5 \\ 5/2 \end{bmatrix}. \tag{1.135}$$

Example 1.75 As a final twist, let's examine the assembly for meshes with a different element type and with an increasing number of elements, so as to observe the pattern of non-zero entries that emerges. For simplicity, we consider meshes with elements of equal length, and the bilinear form and linear functional in 1.126 with $f(x) = k(x) = c(x) = 1$. In this way, all element stiffness matrices and element load vectors are the same, since none of the three functions depends on x.

Since $\Omega = (0,1)$, the vertices for this mesh are at $x_i = (i-1)h$ for $i = 1,\dots,n_{\text{el}}+1$ and $h = 1/n_{\text{el}}$. We will consider a finite element space made of continuous piecewise quadratic functions (P_2-elements), and as customary, we add a node at the midpoint of each elements to indicate the third local degree of freedom. Thus, we set $x_{n_{\text{el}}+1+i} = (x_i + x_{i+1})/2$ for $i = 1,\dots,n_{\text{el}}$. Figure 1.16 shows the mesh for $n_{\text{el}} = 3$.

The element stiffness matrix is a 3×3 matrix, the load vector has length 3, and are computed as

$$\begin{aligned}
K^e_{ab} &= \int_{x_e}^{x_{e+1}} \left[N^e_{a,x}(x)N^e_{b,x}(x) + N_a(x)N_b(x)\right]\,dx,\\
F^e_a &= \int_{x_e}^{x_{e+1}} N_a(x)\,dx,
\end{aligned}$$

where the shape functions are those in (1.108). We explicitly show the computation for two entries of K^e, and leave the rest for the reader to verify. From (1.108), we will use that $x_1^e - x_2^e = h$, $x_3^e - x_1^e = x_2^e - x_3^e = h/2$, and that

$$\begin{aligned}
N_1^e\left(x_1^e + \xi h\right) &= (\xi-1)(2\xi-1), & N^e_{1,x}(x_1^e + h\xi) &= \frac{4\xi-3}{h}\\
N_3^e\left(x_1^e + \xi h\right) &= 4(1-\xi)\xi, & N^e_{3,x}(x_1^e + h\xi) &= \frac{4-8\xi}{h}
\end{aligned} \tag{1.136}$$

for $\xi \in [0,1]$. Then,

$$\begin{aligned}
K^e_{11} &= \int_{x_e}^{x_{e+1}} \left[N^e_{1,x}(x)N^e_{1,x}(x) + N_1(x)N_1(x)\right]\,dx\\
&= \int_0^1 \left[N^e_{1,x}(x_e+\xi h)N^e_{1,x}(x_e+\xi h) + N_1(x_e+\xi h)N_1(x_e+\xi h)\right] h\,d\xi && \text{Change of variables } \xi = \tfrac{x-x_e}{h}\\
&= \int_0^1 \left[\frac{(4\xi-3)^2}{h^2} + (\xi-1)^2(2\xi-1)^2\right] h\,d\xi && \text{From (1.136)}\\
&= \frac{7}{3h} + \frac{2h}{15}.\\
K^e_{13} &= \int_{x_e}^{x_{e+1}} \left[N^e_{1,x}(x)N^e_{3,x}(x) + N_1(x)N_3(x)\right]\,dx\\
&= \int_0^1 \left[N^e_{1,x}(x_e+\xi h)N^e_{3,x}(x_e+\xi h) + N_1(x_e+\xi h)N_3(x_e+\xi h)\right] h\,d\xi && \text{Change of variables } \xi = \tfrac{x-x_e}{h}\\
&= \int_0^1 \left[\frac{(4\xi-3)(4-8\xi)}{h^2} - 4(\xi-1)^2(2\xi-1)\xi\right] h\,d\xi && \text{From (1.136)}\\
&= -\frac{8}{3h} + \frac{h}{15}.
\end{aligned}$$

Proceeding with the computation, the element stiffness matrix is then

$$K^e = \begin{bmatrix} \frac{7}{3h} + \frac{2h}{15} & \frac{1}{3h} - \frac{h}{30} & -\frac{8}{3h} + \frac{h}{15} \\ \frac{1}{3h} - \frac{h}{30} & \frac{7}{3h} + \frac{2h}{15} & -\frac{8}{3h} + \frac{h}{15} \\ -\frac{8}{3h} + \frac{h}{15} & -\frac{8}{3h} + \frac{h}{15} & \frac{16}{3h} + \frac{8h}{15} \end{bmatrix}, \tag{1.137}$$

and the element load vector is

$$F^e = \begin{bmatrix} \frac{h}{6} \\ \frac{h}{6} \\ \frac{2h}{3} \end{bmatrix}. \tag{1.138}$$

Next, we assemble the stiffness matrix and load vector for a mesh with 3, 6, and 9 elements of the same length, with $\eta_c = \{1\}$ and $\overline{u}_1 = -1$. As always, we need to decide how to index the global degrees of freedom. In this case, it is customary to adopt the number of the node as the index of the global degree of freedom, so the local-to-global maps for each case are

$$\begin{aligned} n_{\text{el}} = 3, \quad \mathsf{LG} &= \begin{bmatrix} 1 & 2 & 3 \\ 2 & 3 & 4 \\ 5 & 6 & 7 \end{bmatrix} \\ n_{\text{el}} = 6, \quad \mathsf{LG} &= \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 2 & 3 & 4 & 5 & 6 & 7 \\ 8 & 9 & 10 & 11 & 12 & 13 \end{bmatrix} \\ n_{\text{el}} = 9, \quad \mathsf{LG} &= \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \\ 11 & 12 & 13 & 14 & 15 & 16 & 17 & 18 & 19 \end{bmatrix}. \end{aligned} \tag{1.139}$$

When $n_{\text{el}} = 3$, $h = 1/3$, the stiffness matrix is

$$K = \frac{1}{90} \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 89 & 1268 & 89 & 0 & -718 & -718 & 0 \\ 0 & 89 & 1268 & 89 & 0 & -718 & -718 \\ 0 & 0 & 89 & 634 & 0 & 0 & -718 \\ -718 & -718 & 0 & 0 & 1456 & 0 & 0 \\ 0 & -718 & -718 & 0 & 0 & 1456 & 0 \\ 0 & 0 & -718 & -718 & 0 & 0 & 1456 \end{bmatrix},$$

and the (transpose of the) force vector is

$$F^{\mathsf{T}} = \frac{1}{18} \begin{bmatrix} -18 & 2 & 2 & 1 & 4 & 4 & 4 \end{bmatrix}.$$

When $n_{\text{el}} = 6$, $h = 1/6$, the stiffness matrix is

$$K = \frac{1}{180} \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 359 & 5048 & 359 & 0 & 0 & 0 & 0 & -2878 & -2878 & 0 & 0 & 0 & 0 \\ 0 & 359 & 5048 & 359 & 0 & 0 & 0 & 0 & -2878 & -2878 & 0 & 0 & 0 \\ 0 & 0 & 359 & 5048 & 359 & 0 & 0 & 0 & 0 & -2878 & -2878 & 0 & 0 \\ 0 & 0 & 0 & 359 & 5048 & 359 & 0 & 0 & 0 & 0 & -2878 & -2878 & 0 \\ 0 & 0 & 0 & 0 & 359 & 5048 & 359 & 0 & 0 & 0 & 0 & -2878 & -2878 \\ 0 & 0 & 0 & 0 & 0 & 359 & 2524 & 0 & 0 & 0 & 0 & 0 & -2878 \\ -2878 & -2878 & 0 & 0 & 0 & 0 & 0 & 5776 & 0 & 0 & 0 & 0 & 0 \\ 0 & -2878 & -2878 & 0 & 0 & 0 & 0 & 0 & 5776 & 0 & 0 & 0 & 0 \\ 0 & 0 & -2878 & -2878 & 0 & 0 & 0 & 0 & 0 & 5776 & 0 & 0 & 0 \\ 0 & 0 & 0 & -2878 & -2878 & 0 & 0 & 0 & 0 & 0 & 5776 & 0 & 0 \\ 0 & 0 & 0 & 0 & -2878 & -2878 & 0 & 0 & 0 & 0 & 0 & 5776 & 0 \\ 0 & 0 & 0 & 0 & 0 & -2878 & -2878 & 0 & 0 & 0 & 0 & 0 & 5776 \end{bmatrix},$$

$$
K = \frac{1}{270}\begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
809 & 11348 & 809 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -6478 & -6478 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 809 & 11348 & 809 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -6478 & -6478 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 809 & 11348 & 809 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -6478 & -6478 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 809 & 11348 & 809 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -6478 & -6478 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 809 & 11348 & 809 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -6478 & -6478 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 809 & 11348 & 809 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -6478 & -6478 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 809 & 11348 & 809 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -6478 & -6478 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 809 & 11348 & 809 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -6478 & -6478 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 809 & 5674 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -6478 \\
-6478 & -6478 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 12976 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -6478 & -6478 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 12976 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -6478 & -6478 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 12976 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -6478 & -6478 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 12976 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -6478 & -6478 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 12976 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -6478 & -6478 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 12976 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -6478 & -6478 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 12976 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -6478 & -6478 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 12976 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -6478 & -6478 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 12976
\end{bmatrix},
$$

Figure 1.19 Stiffness matrix for $n_{\text{el}} = 9$ in Example 1.75.

and the (transpose of the) force vector is

$$
F^{\mathsf{T}} = \frac{1}{36}\begin{bmatrix} -36 & 2 & 2 & 2 & 2 & 2 & 1 & 4 & 4 & 4 & 4 & 4 & 4 \end{bmatrix}.
$$

Finally, when $n_{\text{el}} = 9$, $h = 1/9$, the stiffness matrix is shown in Fig. 1.19 and the (transpose of the) force vector is

$$
F^{\mathsf{T}} = \frac{1}{54}\begin{bmatrix} -54 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 1 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 \end{bmatrix}.
$$

The reader may want to follow this example closely to understand how the assembly works.

Derivation of Assembly Formulas (1.133)**.**

In addition to the way in which global basis functions are constructed, in the following, we will use the facts that: (a) shape functions are zero outside the elements over which they are defined, and (b) the integrals over each element in the definition of ℓ^e and a^e will therefore be zero when computed for a shape function of another element. More precisely, for $a \in \{1, \ldots, m\}$ and $e, e', e'' \in \{1, \ldots, n_{\text{el}}\}$,

$$
N_a^e(x) = 0 \text{ if } x \notin \Omega_e, \text{ from where, } \begin{cases} \ell^e\left(N_a^{e'}\right) = 0 & \text{if } e \neq e', \\ a^e(N_a^{e''}, N_b^{e'}) = 0 & \text{if } e' \neq e \text{ or } e'' \neq e. \end{cases} \tag{1.140}
$$

We examine how we arrive to (1.133a) first. For simplicity, we proceed by assuming that $h_0 = 0$, since it reduces the bookkeeping needed in the derivation. For $A \in \eta_a$,

$$
\begin{aligned}
F_A &= \ell(N_A) && \text{from (1.78}c\text{)}\\
&= \sum_{e=1}^{n_{\text{el}}} \ell^e(N_A) + h_L N_A(L) && \text{from (1.130)}\\
&= \sum_{e=1}^{n_{\text{el}}} \ell^e\left(\mathop{\sum\nolimits^{\circ}}_{\{(a,e')|\mathsf{LG}(a,e')=A\}} N_a^{e'}\right) + h_L \mathop{\sum\nolimits^{\circ}}_{\{(a,e)|\mathsf{LG}(a,e)=A\}} N_a^e(L) && \text{from (1.132)}\\
&= \sum_{e=1}^{n_{\text{el}}} \sum_{\{(a,e')|\mathsf{LG}(a,e')=A\}} \ell^e\left(N_a^{e'}\right) + \mathop{\sum\nolimits^{\circ}}_{\{(a,e)|\mathsf{LG}(a,e)=A\}} h_L N_a^e(L) && \text{linearity}\\
&= \sum_{e=1}^{n_{\text{el}}} \sum_{\{a|\mathsf{LG}(a,e)=A\}} \ell^e(N_a^e) + \sum_{\{a|\mathsf{LG}(a,n_{\text{el}})=A\}} h_L N_a^{n_{\text{el}}}(L), && \text{from (1.140)}\\
&= \sum_{e=1}^{n_{\text{el}}-1} \sum_{\{a|\mathsf{LG}(a,e)=A\}} \ell^e(N_a^e) + \sum_{\{a|\mathsf{LG}(a,n_{\text{el}})=A\}} \left[\ell^{n_{\text{el}}}(N_a^{n_{\text{el}}}) + h_L N_a^{n_{\text{el}}}(L)\right]\\
&= \sum_{e=1}^{n_{\text{el}}} \sum_{\{a|\mathsf{LG}(a,e)=A\}} F_a^e. && \text{from (1.131).}
\end{aligned}
$$

It should be evident from this derivation that the equality in the last line does not change if $h_0 \neq 0$. This proves (1.133a).

Similarly, to obtain (1.133b), for $A \in \eta_a, B \in \eta$,

$$
\begin{aligned}
K_{AB} &= a(N_B, N_A) && \text{from (1.78}c\text{)}\\
&= \sum_{e=1}^{n_{\text{el}}} a^e(N_B, N_A) && \text{from (1.125)}\\
&= \sum_{e=1}^{n_{\text{el}}} a^e\left(\mathop{\sum\nolimits^{\circ}}_{\{(b,e'')|\mathsf{LG}(b,e'')=B\}} N_b^{e''}, \mathop{\sum\nolimits^{\circ}}_{\{(a,e')|\mathsf{LG}(a,e')=A\}} N_a^{e'}\right) && \text{from (1.132)}\\
&= \sum_{e=1}^{n_{\text{el}}} \sum_{\{(a,e')|\mathsf{LG}(a,e')=A\}} \sum_{\{(b,e'')|\mathsf{LG}(b,e'')=B\}} a^e(N_b^{e''}, N_a^{e'}) && \text{bilinearity}\\
&= \sum_{e=1}^{n_{\text{el}}} \sum_{\substack{\{a|\mathsf{LG}(a,e)=A\}\\ \{b|\mathsf{LG}(b,e)=B\}}} a^e(N_b^e, N_a^e) && \text{from (1.140)}\\
&= \sum_{e=1}^{n_{\text{el}}} \sum_{\substack{\{a|\mathsf{LG}(a,e)=A\}\\ \{b|\mathsf{LG}(b,e)=B\}}} K_{ab}^e && \text{from (1.127}a\text{).}
\end{aligned}
$$

1.4.4.1 Symmetrization of the Stiffness Matrix

When the bilinear from is symmetric, it is possible to manipulate the stiffness matrix to obtain a symmetric matrix. Symmetric matrices are needed when some iterative solvers for the linear system are adopted, such as Conjugate Gradients. Additionally, symmetric matrices can be stored with less memory, or more efficiently. Problems in structural mechanics, such as a elasticity, have a symmetric bilinear form and can benefit from a symmetric stiffness matrix.

Given a stiffness matrix K such that (a) $K_{AB} = K_{BA}$ for $A \in \eta_a, B \in \eta_a$ and, (b) $K_{AB} = \delta_{AB}$ if $A \notin \eta_a$, we can construct a symmetric matrix K^S and load vector F^S such that U is the solution of both

$$KU = F \quad \text{and} \quad K^S U = F^S. \tag{1.141}$$

It is then possible to solve $K^S U = F^S$ to find U, instead of $KU = F$.

Any stiffness matrix that emerges from a variational method of the form in Problem 1.2 satisfies condition (a) if the bilinear form is symmetric, and satisfies condition (b) automatically, c.f. (1.78c).

The symmetric stiffness matrix and associated load vector follow as

$$K^S_{AB} = \begin{cases} K_{AB} & \text{if } A \in \eta_a, B \in \eta_a, \\ \delta_{AB} & \text{otherwise.} \end{cases} \tag{1.142a}$$

$$F^S_A = \begin{cases} F_A - \sum_{B \in \eta_g} K_{AB} F_B & \text{if } A \in \eta_a, \\ F_A & \text{otherwise} \end{cases} \tag{1.142b}$$

Example 1.76 Let's symmetrize the matrix in Example 1.74. Notice that the matrix in the earlier example, Example 1.73, is already symmetric. The stiffness matrix and load vector from Example 1.74 are

$$K = \begin{bmatrix} 1 & 0 & 0 \\ -31/16 & 72/16 & -29/16 \\ 0 & -29/16 & 39/16 \end{bmatrix}, \qquad F = \begin{bmatrix} 4 \\ 5 \\ 5/2 \end{bmatrix},$$

with $\eta_a = \{2, 3\}$.

Before we proceed with the direct construction of the matrix, we look at why it works. The linear system defined by the stiffness matrix and load vector is

$$\begin{array}{llllll} 1 \cdot u_1 & +0 \cdot u_2 & +0 \cdot u_3 & = & 4, \\ -31/16 \cdot u_1 & +72/16 \cdot u_2 & -29/16 \cdot u_3 & = & 5, \\ 0 \cdot u_1 & -29/16 \cdot u_2 & +39/16 \cdot u_3 & = & 5/2. \end{array}$$

Because the first line is a constrained index, it defines directly the value of $u_1 = 4$. We can then replace this in the first column of lines 2 and 3, and move them to the right hand side. The linear system can then be written as

$$\begin{array}{llllll} 1 \cdot u_1 & +0 \cdot u_2 & +0 \cdot u_3 & = & 4, \\ 0 \cdot u_1 & +72/16 \cdot u_2 & -29/16 \cdot u_3 & = & 5 + 31/16 \cdot 4, \\ 0 \cdot u_1 & -29/16 \cdot u_2 & +39/16 \cdot u_3 & = & 5/2. \end{array}$$

The matrix associated to this linear system is then symmetric. This is what (1.142) is doing.

Notice that the conditions for symmetrization are satisfied. Condition (a) is satisfied because the submatrix formed by the entries (2,2), (2,3), (3,2), and (3,3) is symmetric. Condition (b) is satisfied because the first row is identically zero, except for the diagonal, in which it is equal to 1. This is of course expected, since this matrix emerged from a variational numerical method and the symmetric bilinear form in Example 1.71.

The symmetrized matrix is

$$K^S = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 72/16 & -29/16 \\ 0 & -29/16 & 39/16 \end{bmatrix},$$

The associated load vector F^S follows as

$$F^S = \begin{bmatrix} 4 \\ 5 \\ 5/2 \end{bmatrix} - 4 \begin{bmatrix} 0 \\ -31/16 \\ 0 \end{bmatrix} = \begin{bmatrix} 4 \\ 51/4 \\ 5/2 \end{bmatrix}.$$

To conclude, you can check that both systems lead to the same solution

$$U = \begin{bmatrix} 4 \\ 9116/1967 \\ 8796/1967 \end{bmatrix}.$$

Derivation of Symmetrization Formulas (1.142)

To see that $KU = F$ if and only if $K^S U = F^S$, we consider first the equations for $A \notin \eta_a$. In this case,

$$F_A = K_{AB} u_B = \delta_{AB} u_B = K^S_{AB} u_B = F^S_A. \tag{1.143}$$

Next, we consider the equations for $A \in \eta_a$. We will need to use that from 1.143,

$$F_B = u_B \text{ for } B \notin \eta_a, \tag{1.144}$$

and from (1.142a),

$$K^S_{AB} = 0 \text{ if } A \in \eta_a, B \notin \eta_a. \tag{1.145}$$

Then,

$$\begin{aligned}
0 &= \sum_{B\in\eta} K_{AB} u_B - F_A \\
&= \sum_{B\in\eta_a} K_{AB} u_B + \sum_{B\notin\eta_a} K_{AB} F_B - F_A && \text{from } (1.142a) \\
&= \sum_{B\in\eta_a} K^S_{AB} u_B - \left(F_A - \sum_{B\in\eta_g} K_{AB} F_B\right) && \text{from } (1.144) \\
&= \sum_{B\in\eta_a} K^S_{AB} u_B - F^S_A && \text{from } (1.142b) \\
&= \sum_{B\in\eta_a} K^S_{AB} u_B + \sum_{B\notin\eta_a} K^S_{AB} u_B - F^S_A && \text{from } (1.145) \\
&= \sum_{B\in\eta} K^S_{AB} u_B - F^S_A.
\end{aligned}$$

Together with (1.143), this proves that if U satisfies one set of equations, it satisfies the others.

1.4.5 Finite Element Bases and Sparse Stiffness Matrices

A glance at the stiffness matrix computed with P_1-elements in (1.101) reveals that the matrix has non-zero entries only along three of its diagonals, so it is called a **tri-diagonal** matrix. Similarly, an examination of the stiffness matrices in Example 1.75 shows that the only non-zero entries lie along 7 different diagonal directions. This means that as the number of elements grows, the majority of the entries in the matrix are equal to zero. In fact, the number of non-zero entries grows linearly with the number of rows $n_{\rm el}+1$, since it is proportional to the length of the diagonals. In contrast, the number of zeros grows quadratically with the number of rows, since it corresponds to the rest of the $(n_{\rm el}+1)^2$ entries. Matrices in which the fraction of non-zero entries is small are called **sparse**.

The sparsity of the matrix has consequences in the computational efficiency of a method. If the matrix is going to be stored in memory[12], it is convenient to store the non-zero entries only. This drastically reduces the memory requirements as the number of elements grows, since the amount of memory needed is $\mathcal{O}(n_{\rm el})$ instead of $\mathcal{O}({n_{\rm el}}^2)$ as $n_{\rm el} \to \infty$. The same scaling law applies to matrix-vector products, which is the main operation used in the solution of linear systems of equations through iterative methods .

For the forthcoming discussion, it is convenient to introduce the definition of support of a function. Given a real-valued function f with domain Ω, the set

$$\operatorname{supp}(f) = \overline{\{x \in \Omega \mid f(x) \neq 0\}} \tag{1.146}$$

is called the **support** of f. In (1.146), the line over set indicates its closure: it says that the support of f is formed by all the points in the set, but also by those that may not be in the set but that can be reached as limits of sequences of points in the set. For the forthcoming discussion, it is enough to think about the support of f as essentially the set of all points in Ω at which f is not equal to zero.

Back to the main discussion then, when are stiffness matrices in the finite element method sparse? For active indices, entries in the stiffness matrix of a variational method are computed as $K_{AB} = a(N_B, N_A)$. The most commonly found scenario is illustrated by the bilinear form in Example 1.55

$$a(u,v) = \int_0^1 u_{,x}(x)\, v_{,x}(x)\, dx. \tag{1.147}$$

In this case, $a(N_B, N_A) = 0$ if $N_{B,x}$ and $N_{A,x}$ are different than zero over non-intersecting regions of the domain, or more precisely, when the intersection of their supports has zero length. For a finite element space of continuous functions

[12] The so-called matrix-free methods never build the matrix, see e.g. CITE

over P_1-elements, $N_{B,x}N_{A,x}$ is a non-zero function only when $|A-B| \leq 1$, that is, when A and B are indices of the same node or of neighboring nodes. This is the reason for the appearance of the tri-diagonal matrix in (1.101), since for each row A, only the entries $K_{A(A-1)}$, K_{AA} and $K_{A(A+1)}$ are non-zero. There are always only at most three non-zero entries per row regardless of the number of elements n_{el}, so the matrix becomes increasingly sparse as n_{el} grows.

A similar scenario is found with the bilinear form of Example 1.75,

$$a(u, v) = \int_0^1 u_{,x} v_{,x} + uv \, dx. \tag{1.148}$$

In this case, if A is the index of a node between elements in the mesh, row A has at most 5 non-zero entries: $K_{AB} \neq 0$ only if $|A-B| \leq 1$ (indices of the neighboring nodes or the same node), $B = A + n_{\text{el}}$, or $B = A + n_{\text{el}} + 1$ (indices of the neighboring nodes in the middle of an element); see Fig. 1.19. Alternatively, if A is the index of a node in the middle of an element, then $K_{AB} \neq 0$ only if $B = A - n_{\text{el}}$ (the index of the node on its right) or $B = A - n_{\text{el}} - 1$ (the index node of the node on its left). As in the previous example, the number of non-zero entries per row is the same for all values of n_{el}, so the matrix becomes increasingly sparse as n_{el} grows.

These two examples should be contrasted with one of a variational method with non-finite element bases. For example, it is possible to select $\mathscr{W}_h = \mathbb{P}_{m-1}(\Omega)$ for $m \geq 1$, with basis functions $N_A(x) = x^{A-1}$ for $A = 1, \ldots, m$. In this case, $\operatorname{supp}(N_A) = \Omega$ for all A, so the support of each basis function is the entire domain. For any of the two bilinear forms (1.147) or (1.148), choosing this basis leads to non-sparse matrices, as we can expect from the non-empty intersection of the support of any pair of basis functions. Specifically, assuming all indices are active, we have

$$K_{AB} = \int_0^1 (A-1)(B-1)x^{A+B-4} \, dx = \frac{(A-1)(B-1)}{A+B-3}, \qquad \text{for (1.147)}$$

$$K_{AB} = \int_0^1 (A-1)(B-1)x^{A+B-4} + x^{A+B-2} \, dx = \frac{(A-1)(B-1)}{A+B-3} + \frac{1}{A+B-1}. \qquad \text{for (1.148)}$$

These are all non-zero entries (except for the $A = 1$ row or $B = 1$ column for (1.147)). Matrices in which most of the entries are non-zero are called **dense**.

Sufficient Conditions for Stiffness Matrices in the Finite Element Method to be Sparse

The following are sufficient conditions on a finite element basis and the bilinear form to generate a sparse stiffness matrix.

If

(a) $\operatorname{supp}(f) \cap \operatorname{supp}(g) = \emptyset \Longrightarrow a(f, g) = 0$, and

(b) There exists $n_{\text{width}} \in \mathbb{N}$ such that for all $A = 1, \ldots, m$ the number of elements of the set $s_A = \{e \mid \mathsf{LG}(a, e) = A\}$ is less or equal than n_{width}, i.e., $\#s_A \leq n_{\text{width}}$, then the number of non-zero entries in each row of K is less or equal than $n_{\text{width}} \times k$.

To see this, notice first that $\text{supp}(N_A) = \cup_{e \in s_A} \Omega_e$, since these are all of the elements in which a shape function is added to form N_A. Therefore, due to the first condition, $K_{AB} = 0$ if $s_A \cap s_B = \emptyset$. For each element $e \in s_A$, there are at most k other basis functions whose support includes Ω_e, given that each shape function is added to one and only one global basis function, c.f., 1.4.3. Thus, due to the second condition, the set $\{B \in \{1, \dots, m\} \mid s_A \cap s_B \neq \emptyset\}$ has at most $n_{\text{width}} \times k$ elements, and hence each row of K has at most that number of non-zero elements.

Let's briefly discuss the conditions and the implications of this result. The first condition is satisfied by essentially all commonly found bilinear forms. The second condition examines the local-to-global map to request global basis functions to be formed by adding at most n_{width} shape functions, regardless of the number of elements in the mesh. Because the number of non-zero entries is less or equal than $n_{\text{width}} \times k$, a quantity that does not change as n_{el} grows, the associated stiffness matrix grows increasingly sparse as $n_{\text{el}} \to \infty$.

We conclude this section with a couple of final remarks:

- The use of methods that generate non-sparse matrices can be convenient when they lead to numerical solutions of similar accuracy with fewer global degrees of freedom than a finite element method, such as in some boundary element methods or spectral methods.

- The property of the finite element basis functions to be non-zero only in a few elements is typically referred to in the literature by stating that *finite element basis functions have compact support.* It is worth reflecting a bit on the meaning of this statement. Mathematically, a function f is said to have **compact support** if $\text{supp}(f)$ is a bounded set. Hence, any function whose support is included in the domain $\Omega = (0, L)$ of our example has compact support. It is clear then that a literal interpretation of this statement does not capture that the support of a finite element basis function is included in the union of at most a fixed number of elements, regardless of the number of elements in the mesh.

1.5 Elliptic Fourth-Order Problems

The goal of this section is to show how to build variational methods for fourth-order problems. For this purpose, we will reenact the same sequence of the previous sections. We will first formulate a class of general fourth-order problems and obtain a variational equation. Then we will deduce and implement a variational method and build an adequate finite element space.

A key insight that we will gain from this example will be the convenience of having additional smoothness of the finite element functions, so that a variational method for the problem is trivially consistent.