---

16 Let

$$\widehat{K} = \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}, \qquad \widehat{M} = \begin{pmatrix} 1/3 & 1/6 \\ 1/6 & 1/3 \end{pmatrix}, \qquad \widehat{D} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}.$$

Show that

$$\frac{1}{6}\widehat{D} \;\leq\; \widehat{M} \qquad \text{and} \qquad \widehat{K} \;\leq\; 2\,\widehat{D}\,.$$

17 Consider the one-dimensional boundary value problem

$$\begin{aligned} -u''(x) &= f(x) \qquad \forall x \in (0,\,1) \\ u(0) &= g_0\,, \qquad u(1) = g_1\,. \end{aligned}$$

Let K_h denote the stiffness matrix obtained by the finite element method using the Courant elements on a subdivision $0 = x_0 < x_1 < \cdots < x_{N_h} = 1$.

Show that

$$\frac{\min_k h_k^2}{6\,c_F^2} D_h \leq K_h \leq 2\,D_h\,,$$

where $D_h = \operatorname{diag}(K_h)$, c_F is the constant arising in Friedrichs' inequality, and $h_k = x_k - x_{k-1}$.

Hint: Use that D_h can be split into element contributions $D_h^{(1)} = K_h^{(1)} = \frac{1}{h_1}$ and $D_h^{(k)} = \operatorname{diag}(K_h^{(k)}) = \frac{1}{h_k}\operatorname{diag}(\widehat{K}) = \frac{1}{h_k}\widehat{D}$.

Programming

18 Define a data type `Mesh` which contains all the information on the mesh $\mathcal{T}_h$ that is necessary for the computation of K_h.

Consider first the case $\Gamma_D = \emptyset$, $\Gamma_R = \{0,\,1\}$, and $\alpha(x) = 0$, which corresponds to homogeneous Neumann boundary conditions (i. e., you needn't worry about any boundary conditions so far).

19 Write a function `AssembleStiffnessMatrix(↓mesh, ↑matrix)` that assembles the global $(n_h+1)\times(n_h+1)$ stiffness matrix `matrix` $= K_h$ for a given subdivision `mesh` $= \mathcal{T}_h$ of Ω.

Hint: Set $K_h = 0$, then start with $K_h^{(1)}$ and loop over all elements T_k to update the matrix K_h. On each element T_k, use the function `ElementStiffnessMatrix` to compute $K_h^{(k)}$ and pay attention to put the entries of $K_h^{(k)}$ at the correct positions in the global matrix K_h.

20 Write a function `AssembleLoadVector(↓(*f)(x), ↓mesh, ↑vector)` that assembles the global (n_h+1)-dimensional load vector `vector` $= \underline{f}_h$ for a given mesh `mesh` $= \mathcal{T}_h$ of Ω.

Hint: Set $\underline{f}_h = 0$, then start with $\underline{f}_h^{(1)}$ and loop over all elements T_k to update the vector $\underline{f}_h$. On each element T_k, use the function `ElementLoadVector` to compute $\underline{f}_h^{(k)}$ and pay attention to add the entries in the right place.

The stiffness matrix and load vector above are now modified, such that we obtain the correct system for other types of boundary conditions, here of Robin/Neumann type.

21 Write a function `ImplementRobinBC(↓i, ↓g, ↓alpha, ↕matrix, ↕vector)` to implement the Robin boundary condition

$$u'(x_i) = \alpha(x_i)\,\big(g_R(x_i) - u(x_i)\big)$$

for given values `g`=$g_R(x_i)$, `alpha`=$\alpha(x_i)$ at the boundary node x_i identified by the index `i`=i. The function `ImplementRobinBC` must update the stiffness matrix `matrix` and the load vector `vector`, previously computed by `AssembleStiffnessMatrix` and `AssembleLoadVector`, respectively, in the case of homogeneous Neumann conditions.

Test the implemented data types and functions using some simple examples, e. g., consider equidistant nodes x_i for different values of n_h, and simple functions $f(x) = 1$, $f(x) = x$, etc.