On the interval (0,1) we consider a steady-state convection-diffusion-reaction equation, with homogeneous Neumann boundary conditions:

$$-D\frac{d^2u}{dx^2} + \lambda u = f(x), \tag{1}$$

$$-D\frac{du}{dx}(0) = 0, \qquad -D\frac{du}{dx}(1) = 0. \tag{2}$$

1. Assignment 1 Derive the *weak* formulation.
   For this, we multiply Equation (1) by the basis functions ϕ and integrate over the domain.

$$\int_0^1 -D\phi\frac{d^2u}{dx^2} + \lambda\phi u dx = \int_0^1 \phi f(x)dx \tag{3}$$

$$= \int_0^1 -D\left[\frac{du}{dx}\left(\phi\frac{du}{dx}\right) - \frac{d\phi}{dx}\frac{du}{dx} + \lambda\phi u\right]dx =$$
$$\int_0^1 -D\mathbf{n}\cdot\phi\frac{du}{dx}ds + \int_0^1 D\left[\frac{d\phi}{dx}\frac{du}{dx} + \lambda\phi u\right]dx$$

   But using Equation (1) (bc) the first term above is zero, then we have:

$$\int_0^1 \left[D\frac{d\phi}{dx}\frac{du}{dx} + \lambda\phi u\right]dx = \int_0^1 \phi f(x)dx \tag{4}$$

2. Assignment 2 Write the Galerkin formulation of the weak form as derived in the previous assignment for a general number of elements given by n (hence $x_n = 1$). Give the Galerkin equations, that is, the linear system in terms of

$$S\bar{u} = \bar{f}, \tag{5}$$

   all expressed in the basis-functions, $f(x)$, λ and D.
   For the Galerkin formulation we approximate u with the basis functions ϕ_j as:

$$u(x) \sim \sum_{j=1}^{n} a_j\phi_j(x).$$

   Approximating u and substituting $\phi = \phi_i$ in Equation (4) we have:

$$\sum_{j=1}^{n} a_j \int_0^1 \left[D\frac{d\phi_i}{dx}\frac{d\phi_j}{dx} + \lambda\phi_i\phi_j\right]dx = \int_0^1 \phi_i f(x)dx \tag{6}$$

   Then

$$S_{ij}^{e_k} = a_j\int_{e_k}\left[D\frac{d\phi_i}{dx}\frac{d\phi_j}{dx} + \lambda\phi_i\phi_j\right]dx, \qquad S_{ij} = \sum_{k=1}^{nel} S_{ij},$$

$$f_i^{e_k} = \int_{e_k}\phi_i f(x)dx, \qquad f_i = \sum_{k=1}^{nel} f_i^{e_k}.$$

Algorithm 1

```
function [x]=GenerateMesh(n)
    x = linspace(0,1,n);
end
```

3. Assignment 3 Write a matlab routine, called GenerateMesh.m that generates an equidistant distribution of meshpoints over the interval [0, 1], where $x_1 = 0$ and $x_n = 1$ and $h = \frac{1}{n-1}$. You may use x = linspace(0,1,n).

4. Assignment 4 Write a routine, called GenerateTopology.m, that generates a two dimensional array, called elmat, which contains the indices of the ver- tices of each element, that is

$$elmat(i,1) = i. \qquad fori \in 1, ..., n-1.$$
$$elmat(i,2) = i+1$$

Next we compute the element matrix S_{elem}. In this case, the matrix is the same for each element, that is, if we consider element e_i .

Algorithm 1

```
function [elmat] = GenerateTopology(n)
    elmat(i, 1) = i;
    elmat(i, 2) = i + 1;
end
```

5. Assignment 5 Compute the element matrix, S_{elem} over a generic line element e_k.
For each element we have the following marix:

$$S^{e_k} = \begin{bmatrix} \int_{e_k} D\phi'_{k-1}\phi'_{k-1} + \lambda\phi_{k-1}\phi_{k-1}dx & \int_{e_k} D\phi'_{k-1}\phi'_k + \lambda\phi_{k-1}\phi_k dx \\ \int_{e_k} D\phi'_k\phi'_{k-1} + \lambda\phi_k\phi_{k-1}dx & \int_{e_k} D\phi'_k\phi'_k + \lambda\phi_k\phi_k dx \end{bmatrix}$$

According to Holland and Bell's Theorem, we have:

$$\int_{e_k} \phi_{k-1}\phi_{k-1}dx = \frac{||x_k - x_{k-1}||}{3}$$
$$\int_{e_k} \phi_k\phi_{k-1}dx = \frac{||x_k - x_{k-1}||}{6}$$
$$\int_{e_k} \phi'_{k-1}\phi'_{k-1}dx = \frac{1}{||x_k - x_{k-1}||}$$
$$\int_{e_k} \phi'_{k-1}\phi'_k dx = -\frac{1}{||x_k - x_{k-1}||}$$

Then we have

$$S^{e_k} = \begin{bmatrix} D\frac{1}{||x_k - x_{k-1}||} + \lambda\frac{||x_k - x_{k-1}||}{3} & -D\frac{1}{||x_k - x_{k-1}||} + \lambda\frac{||x_k - x_{k-1}||}{6} \\ -D\frac{1}{||x_k - x_{k-1}||} + \lambda\frac{||x_k - x_{k-1}||}{6} & D\frac{1}{||x_k - x_{k-1}||} + \lambda\frac{||x_k - x_{k-1}||}{3} \end{bmatrix}$$

Algorithm 3

```
function [Selem]=GenerateElementMatrix(D,l,n)
    h = 1/(n-1);
    S(1,1) = D/h + l*h/3;
    S(1,2) = -D/h + l*h/6;
    S(2,1) = -D/h + l*h/6;
    S(2,2) = D/h + l*h/3;
end
```

But $||x_k - x_{k-1}|| = h = \frac{1}{n-1}$ then

$$S^{e_k} = \begin{bmatrix} D\frac{1}{h} + \lambda\frac{h}{3} & -D\frac{1}{h} + \lambda\frac{h}{6} \\ -D\frac{1}{h} + \lambda\frac{h}{6} & D\frac{1}{h} + \lambda\frac{h}{3} \end{bmatrix}$$

According to Newton-Cotes' Theorem, we have:

$$\int_{e_k} \phi_{k-1} f(x)dx = \frac{x_k - x_{k-1}}{2} f(x_k)$$

$$fe_k = \begin{bmatrix} \int_{e_k} \phi_{k-1} f(x)dx \\ \int_{e_k} \phi_k f(x)dx \end{bmatrix}$$

6. Assignment 6 Write a matlab routine, called GenerateElementMatrix.m, in which S_{elem} (2 x 2-matrix) is generated. Subsequently, we are going to sum the connections of the vertices in each element matrix, over all the elements. The result is an n-by-n matrix, called S.

7. Assignment 7 Write a matlab routine, called AssembleMatrix.m, that performs this summation, such that S is first initialized as a zero n-by-n matrix and subsequently:

$$S(elmat(i,j), elmat(i,k)) = S(elmat(i,j), elmat(i,k)) + Selem(j,k), \quad (7)$$

for $j, k \in \{1, 2\}$ over all elements $i \in \{1, ..., n-1\}$. Note that GenerateElementMatrix.m needs to be called for each element. Now, you developed a routine for the assembly of the large matrix S from the element matrices Selem for each element. This procedure is common for the construction of the large discretization matrices needed in Finite Element methods. The procedure, using the array elmat looks a bit overdone and complicated. However, this approach facilitates the application to multi dimensional problems. The next step is to generate a large right-hand side vector using the same procedure. First, we need the element vector.

Algorithm 4

```
function [S]=AssembleMatrix(elmat,Selem,n)
S = zeros(n,n);
for i = 1 : n-1
    for j = 1 : 2
        for k = 1 : 2
            S(elmat(i, j), elmat(i, k)) = S(elmat(i, j), elmat(i, k))+Selem(j, k);
        end
    end
end
end
```