

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

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

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

1. Assignment 1 Derive the *weak* formulation.

For this, we multiply Equation (1) by the basis functions  $\phi$  and integrate over the domain.

$$\begin{aligned} \int_0^1 -D \phi \frac{d^2 u}{dx^2} + \lambda \phi u dx &= \int_0^1 \phi f(x) dx \\ &= \int_0^1 -D \left[ \frac{du}{dx} \left( \phi \frac{du}{dx} \right) - \frac{d\phi}{dx} \frac{du}{dx} + \lambda \phi u \right] dx = \\ &= \int_0^1 -D \mathbf{n} \cdot \phi \frac{du}{dx} ds + \int_0^1 D \left[ \frac{d\phi}{dx} \frac{du}{dx} + \lambda \phi u \right] dx \end{aligned} \quad (3)$$

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

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

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

$$S\bar{u} = \bar{f}, \quad (5)$$

all expressed in the basis-functions,  $f(x)$ ,  $\lambda$  and  $D$ .

For the Galerkin formulation we approximate  $u$  with the basis functions  $\phi_j$  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:

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

Then

$$\begin{aligned} S_{ij}^{e_k} &= a_j \int_{e_k} \left[ D \frac{d\phi_i}{dx} \frac{d\phi_j}{dx} + \lambda \phi_i \phi_j \right] dx, & S_{ij} &= \sum_{k=1}^{nel} S_{ij}^{e_k}, \\ f_i^{e_k} &= \int_{e_k} \phi_i f(x) dx, & f_i &= \sum_{k=1}^{nel} f_i^{e_k}. \end{aligned}$$

**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  $x = \text{linspace}(0,1,n)$ .
4. Assignment 4 Write a routine, called GenerateTopology.m, that generates a two dimensional array, called elmat, which contains the indices of the vertices of each element, that is

$$\begin{aligned} \text{elmat}(i, 1) &= i. & \text{for } i \in 1, \dots, n-1. \\ \text{elmat}(i, 2) &= i+1 \end{aligned}$$

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)
for i = 1 : n-1;
    elmat(i, 1) = i;
    elmat(i, 2) = i + 1;
end
end

```

5. Assignment 5 Compute the element matrix,  $S_{elem}$  over a generic line element  $e_k$ .

For each element we have the following matrix:

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

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

$$\begin{aligned} \int_{e_k} \phi_{k-1}\phi_{k-1}dx &= \frac{\|x_k - x_{k-1}\|}{3} \\ \int_{e_k} \phi_k\phi_{k-1}dx &= \frac{\|x_k - x_{k-1}\|}{6} \\ \int_{e_k} \phi'_{k-1}\phi'_{k-1}dx &= \frac{1}{\|x_k - x_{k-1}\|} \\ \int_{e_k} \phi'_{k-1}\phi'_kdx &= -\frac{1}{\|x_k - x_{k-1}\|} \end{aligned}$$

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);
    Selem(1,1) = D/h + l*h/3;
    Selem(1,2) = -D/h + l*h/6;
    Selem(2,1) = -D/h + l*h/6;
    Selem(2,2) = D/h + l*h/3;
end

```

**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
end

```

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

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

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  $i \in \{1, \dots, n-1\}$ . 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.

8. Assignment 8 Compute the element vector over a generic line-element. For this purpose, we proceed as follows

**Algorithm 5**

```

function [felem]=GenerateElementVector(n)
h = 1 / (n-1);
for i=1:n-1
felem(1,i) = (h/2)*fn(x(i));
felem(2,i) = (h/2)*fn(x(i+1));
end
end

```

**Algorithm 6**

```

[f]=AssembleVector(felem,elmat,n)
f=zeros(n,1);
for i=1:n-1
    for j=1:2
        f(elmat(i,j)) = f(elmat(i,j)) + felem(j,i);
    end
end
end

```

## 9. Assignment 9 Implementation of the right-hand vector:

- a Write a matlab routine, called GenerateElementVector.m, that gives the vector felem (column vector of length 2). in which felem (1) and felem (2) respectively provide information about node i and node i + 1, which are the vertices of element  $e_i$ . This is needed for all elements. Use  $f(x) = 1$  here.

According to Newton-Cotes' Theorem, we have:

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

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

Then, for each element we have

$$f_{e_k} = \frac{h}{2} \begin{bmatrix} f(x_{k-1}) \\ f(x_k) \end{bmatrix} = \frac{h}{2} \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

- b Write a matlab routine, called AssembleVector.m, that performs the following summation after setting  $f = \text{zeros}(n, 1)$ :

$$f(\text{elmat}(i, j)) = f(\text{elmat}(i, j)) + f_{\text{elem}}(j), \quad (8)$$

for  $j \in \{1, 2\}$  over all elements  $i \in \{1, \dots, n-1\}$ .

10. Assignment 10 Run the assembly routines to get the matrix  $S$  and vector  $f$  for  $n = 100$ .

**Algorithm 7**

```

clear all
[x] = GenerateMesh(n);
[elmat] = GenerateTopology(n);
[Selem] = GenerateElementMatrix(D,lambda,n);
[S] = AssembleMatrix(elmat,Selem,n);
[felem] = GenerateElementVector(n,fn,x(1:2));
[f] = AssembleVector(felem,elmat,n);
u = S\ f;
plot(x,u)

```

**Algorithm 7b**

```

clear all
n = 100;
D = 1;
lambda = 1;
fn = @(x) 1;
[x] = GenerateMesh(n);
[elmat] = GenerateTopology(n);
[Selem] = GenerateElementMatrix(D,lambda,n);
[S] = AssembleMatrix(elmat,Selem,n);
[felem] = GenerateElementVector(n,fn,x);
[f] = AssembleVector(felem,elmat,n);
u = S\ f;
plot(x,u)

```

11. Assignment 11 Write the main program that gives the finite-element solution. Call the main program femsolve1d.m.
12. Assignment 12 Compute the Finite Element solution  $u$  for  $f(x) = 1$ ,  $D = 1$ ,  $\lambda = 1$  and  $n = 100$  by using  $u = S \backslash f$  in matlab. Plot the solution. Is this what you would expect?
13. Assignment 13 Choose  $f(x) = \sin(20x)$ , do some experiments with several values of  $n$  ( $n = 10, 20, 40, 80, 160$ ). Plot the solutions for the various numbers of gridnodes in one plot. Explain what you see.

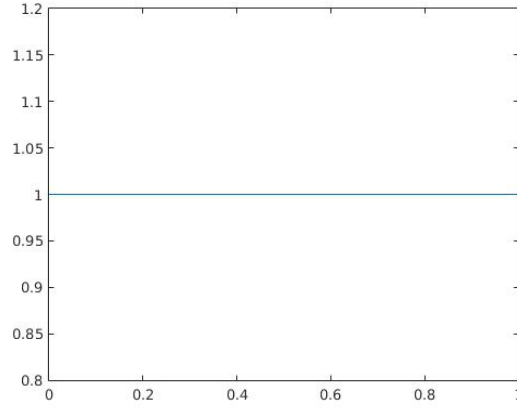


Figure 1: Result  $f(x) = 1$ .

#### Algorithm 7c

```

clear all
close all
t = 0;
for n = [10 20 40 80 160]
    n = 100;
    D = 1;
    lambda = 1;
    fn = @(x) sin(20*x);
    [x] = GenerateMesh(n);
    [elmat] = GenerateTopology(n);
    [Selem] = GenerateElementMatrix(D,lambda,n);
    [S] = AssembleMatrix(elmat,Selem,n);
    [felem] = GenerateElementVector(n,fn,x);
    [f] = AssembleVector(felem,elmat,n);
    u = S \ f;
    plot(x,u)
    hold on
    t = t + 1;
    name{t}=num2str(n);
end
legend(['n=' name{1}],['n=' name{2}],['n=' name{3}],['n=' name{4}],['n=' name{5}])

```

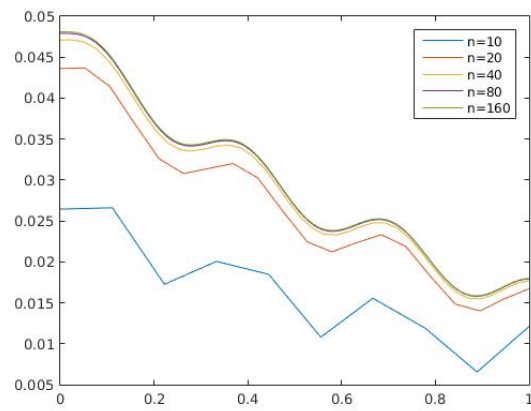


Figure 2: Results  $f(x) = \sin(20x)$ .