Faculty of Electrical Eng, Mathematics and Computer Science

Lab Assignment for 1-Dimensional Code 2015–2016 WI4243FEM

Finite-element analysis for Applied Physics

Fred Vermolen

Delft Institute of Applied Mathematics

Delft University of Technology

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

$$\begin{aligned} -D\frac{d^2u}{dx^2} + \lambda u &= f(x), \\ -D\frac{du}{dx}(0) = 0, \qquad D\frac{du}{dx}(1) &= 0. \end{aligned} \tag{1}$$

Here D, and λ are positive real constants. Further, $f(x)$ is a given function.

The interval $[0,1]$ is divided into $n-1$ elements (where n is a given positive integer), such that $e_i = [x_i, x_{i+1}]$, for $i \in \{1,\ldots,n-1\}$. So element e_i has end points (also called vertices) x_i and x_{i+1}, where we require $x_1 = 0$ and $x_n = 1$ and $h = 1/(n-1)$. Hence there are n gridpoints. In this lab assignment, the participant develops a finite-element code for 1D in Matlab from scratch. The treatment is formal in terms of topology, element matrices and vectors such that the student gets the idea of how finite-element packages are constructed. Once the mesh and topology have been adapted to multi-dimensional problems, then it is relatively straightforward to adjust the code to higher dimensional problems.

Assignment 1 *Derive a weak form of the above problem (see equation (1)), where the order of the spatial derivative is minimised. Take care of the boundary conditions.* ◊

We are going to solve this differential equation by the use of Galerkin's Finite Element method.

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*

$$S\underline{u} = \underline{f}, \tag{2}$$

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

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).* ◊

Further, we need to know which vertices belong to a certain element i.

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*

$$\begin{matrix} elmat(i,1) = i \\ elmat(i,2) = i+1 \end{matrix}, \quad for\ i \in \{1,\ldots,n-1\}. \tag{3}$$

◊

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.

Assignment 5 *Compute the element matrix, S_{elem} over a generic line element e_i.* $\Diamond$

Assignment 6 *Write a matlab routine, called* GenerateElementMatrix.m*, in which S_{elem} (2×2-matrix) is generated.* $\Diamond$

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.

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:*

$$S(elmat(i,j), elmat(i,k)) = S(elmat(i,j), elmat(i,k)) + S_{elem}(j,k), \quad (4)$$

for $j, k \in \{1, 2\}$ over all elements $i \in \{1, \ldots, n-1\}$. Note that GenerateElementMatrix.m needs to be called for each element. $\Diamond$

Now, you developed a routine for the assembly of the large matrix S from the element matrices S_{elem} 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.

Assignment 8 *Compute the element vector over a generic line-element.* $\Diamond$

For this purpose, we proceed as follows

Assignment 9 *Implementation of the right-hand vector:*

a *Write a matlab routine, called* GenerateElementVector.m*, that gives the vector f_{elem} (column vector of length 2). in which $f_{elem}(1)$ and $f_{elem}(2)$ 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.*

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

$$f(elmat(i,j)) = f(elmat(i,j)) + f_{elem}(j), \quad (5)$$

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

$\Diamond$

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

Assignment 11 *Write the main program that gives the finite-element solution. Call the main program* femsolve1d.m. *The program femsolve1d.m should consist of*

clear all
GenerateMesh
GenerateTopology
AssembleMatrix
AssembleVector
$u = S\backslash f$
plot(x,u) $\diamondsuit$

Now, you wrote the backbone of a simple Finite Element program for a one dimensional model problem. The discretisation matrix and right-hand side vector have been constructed.

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?* $\diamondsuit$

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.* $\diamondsuit$

You just wrote a simple finite-element code in such a way that an extension to two- and three dimensional Finite Element programs is rather straightforward. All you need to know is, which mesh points are vertices of each element. The latter distribution is commonly called the topology of the elements.