

ATHENS Course Introduction to Finite Elements

Computer Assignment Day 3

Galerkin Finite Element Solution of a Heat Equation in One Dimension

In this assignment, we solve a heat equation in one spatial dimension. A new issue is the integration of the partial differential equation in time. We consider the following problem for $u(t, x)$

$$(P_1) \begin{cases} \frac{\partial u}{\partial t} = -\frac{\partial^2 u}{\partial x^2}, & \text{for } (t, x) \in (0, T] \times (0, 1), \\ \frac{\partial u}{\partial x}(t, 0) = \frac{\partial u}{\partial x}(t, 1) = 0, & \text{for } t \in (0, T], \\ u(0, x) = \cos(\pi x) + 1, & \text{for } x \in (0, 1). \end{cases} \quad (1)$$

Assignment 1 Show that the analytic solution of the above heat problem (P_1) , including the initial and boundary conditions, is given by

$$u(t, x) = e^{-\pi^2 t} \cos(\pi x) + 1.$$

Plot this solution as a function of x at several times t .

To obtain a finite element solution, we divide the spatial interval $(0, 1)$ into n elements (where n is a given positive integer), such that $e_i = [x_i, x_{i+1}]$, for $i \in \{1, \dots, n\}$. So element e_i has vertices x_i and x_{i+1} , where we require $x_1 = 0$ and $x_{n+1} = 1$.

Assignment 2 Write a matlab routine, called `GenerateMesh.m`, that returns as output a vector of equidistant meshpoints x_i , $i \in \{1, \dots, n+1\}$ (You may use the file that you created yesterday).

Further, we need to know which vertices belong to a certain element i . This is called the *topology* of the mesh.

Assignment 3 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

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

(You may use the file that you created yesterday).

To solve problem (P_1) by the use of the Galerkin finite element method, we have to formulate (P_1) in a weak sense.

Assignment 4 Give a weak formulation of problem (P_1) , in which the order of the spatial derivatives is minimized. Note that both boundary conditions are essential.

Subsequently, the solution is expressed as a linear combination of linear basis functions $\phi_j(x)$, for $j \in \{1, \dots, n+1\}$, that is $u(t, x) = \sum_{j=1}^{n+1} u_j(t) \phi_j(x)$.

Assignment 5 Give the Galerkin equations, and distinguish between the various matrices that appear. What is the size of the system of ordinary differential equations with respect to time, that we will have to solve?

For each element, we generate the element matrices. For this time dependent, we need two matrices to be constituted M and S , for which we have

$$M\mathbf{u}' = S\mathbf{u},$$

where M and S represent the *mass* matrix and *stiffness* matrix respectively. These matrices are formed by assembly of the element matrices.

Assignment 6 Show that for linear basis functions, the element mass and element stiffness matrices, on element i , are given by

$$M_{e_i} = \frac{x_{i+1} - x_i}{2} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad S_{e_i} = \frac{1}{x_{i+1} - x_i} \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}. \quad (3)$$

For each element, the element stiffness and mass matrices S_e and M_e are generated. Subsequently, we are going to create the large stiffness and mass matrices that are necessary to obtain our finite element solution. For instance, in the case of an equidistant grid, where $x_{i+1} - x_i = h$, we will have $S(1, 1) = S(n+1, n+1) = 1/h$, $S(i, i) = 2/h$ for $i \in \{2, \dots, n\}$, and $S(i, i+1) = S(i, i-1) = -1/h$. The remaining entries are zero. To obtain the large matrices S and M , we need to do an assembly procedure.

Assignment 7 Write a matlab routine, called `AssemblyStiffnessMatrix.m`, that performs this operation, where S is initialized as a (sparse) zero $n+1$ -by- $n+1$ matrix, and subsequently:

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

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

Write a similar routine, called `AssemblyMassMatrix.m`, to get the large mass matrix.

Now, we are at the stage of solving the problem

$$M\mathbf{u}' = S\mathbf{u}, \quad \text{with our initial condition.}$$

We will consider several time integration methods.

Assignment 8 Write a routine for the (explicit) forward Euler time integration of this problem. Integrate from $t = 0$ to $t = 1$, with an element size of $h = 0.1$ and $\Delta t = 0.001$. Repeat the simulations with $\Delta t = 0.1$ and $h = 0.01$. What do you observe? Explain your results.

Assignment 9 Change the routine in which the (implicit) backward Euler time integration. Repeat the simulations. Show plots of your solution and compare your result with the exact solution.

Assignment 10 Change the routine in which the trapezoidal (Crank-Nicholson) method is used. Repeat the simulations. Compare your solution with the exact solution.