LAB4: FEM FOR 1D INITIAL-BOUNDARY VALUE PROBLEMS DUE: 2ND OF JUNE, 12:00

In one of the previous lab we learned how to solve boundary value problems without any time dependence. Such problems are called stationary or steady, and they represent physical processes that have reached equilibrium.

In this lab we extend the methods we have learned to solve time-dependent (also called unsteady) problems. Specifically, we will consider the heat equation and the wave equation. Although we only consider problems in one spatial dimension here, the methods apply just as well to problems in higher dimensions.

1. The heat equation

The first problem we consider is the heat equation, which could be used to describe a diffusion process or the evolution of the temperature in a semi-infinite plate or a thin rod. For this problem the unknown function $u = u(x, t)$ depends on both the spatial coordinate x and the time t. We seek a solution to the following problem,

$$u_t - (cu_x)_x = f \quad \text{in} \quad (a, b) \times (0, T], \tag{1a}$$

where subscripts u_t and u_x denotes partial derivatives with respect to t and x. As before, c and f are given functions (depending now on both x and t), and we have a set of boundary conditions at the endpoints of the interval,

$$\left.\begin{aligned} u(a, t) &= u_a(t) \\ c(b, t)u_x(b, t) &= q_b(t) \end{aligned}\right\} \quad \text{for} \quad t \in (0, T]. \tag{1b}$$

Here the boundary conditions may also depend on the time t. For the heat equation we also need an *initial condition*,

$$u(x, 0) = g(x) \quad \text{for} \quad x \in (a, b). \tag{1c}$$

Again, the first step in solving this problem is to derive a weak or variational formulation. We multiply the equation in (1a) with a smooth test function v with $v(a) = 0$ (because there is a Dirichlet boundary condition at the left boundary) and integrate over the spatial domain (a, b), and we get

$$\int_a^b \big(u_t - (cu_x)_x\big)v \, dx = \int_a^b fv \, dx.$$

Now we do integration by parts on the second term on left hand side. This is exactly the same as in the previous time-independent case, and we get the variational problem: Find u such that

$$\int_a^b u_t(x, t)v(x) + cu_x(x, t)v_x(x) \, dx = \int_a^b f(x, t)v(x) \, dx + q_b(t)v(b) \qquad \forall v, \text{ with } v(a) = 0. \tag{2}$$

Here we have integrated away the dependence on the spatial coordinate x, but not on the time t, so (2) must hold for each $0 < t \le T$.

2. A finite element method for the heat equation

The variational formulation (2) is the starting point for deriving a finite element formulation. In the previous lab, we discretized the variational form by partitioning the interval and introducing a subordinate basis of "hat functions" to represent the solution. In effect, the continuous coordinate x was replaced with a discrete set $x_1, x_2, \ldots, x_n$. For the current time-dependent problem, we will also have to similarly discretize the time coordinate t.

2.1. Discretizing in space.

First we derive a space discretization based on FEM without taking into account the Dirichlet boundary condition. We will modify the linear algebra system we obtain later to enforce the Dirichlet boundary condition. As for the stationary problem in the previous lab, we introduce a partitioning of the interval (a, b), with $a = x_1 < x_2 < \ldots < x_n = b$, and we define a finite-dimensional space V_h, consisting of the continuous and piecewise linear functions with the same basis $\{\varphi_j\}_{j=1}^n$ as before,

$$\varphi_j(x_i) = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j. \end{cases}$$

Note that the functions in V_h vary in space, but not in time. We introduce the time dependence in the expansion coefficients with respect to the basis and look for u_h in the form

$$u_h(x,t) = \sum_{j=1}^{n} \xi_j(t)\varphi_j(x).$$

Now, in (2), we replace u with u_h and v with a basis function φ_i. We then have

$$\sum_{j=1}^{n} \int_a^b \dot{\xi}_j(t)\varphi_j(x)\varphi_i(x)\,dx + \sum_{j=1}^{n} \int_a^b c(t,x)\xi_j(t)\varphi_j'(x)\varphi_i'(x)\,dx$$
$$= \int_a^b f(x,t)\varphi_i(x)\,dx + q_b(t)\varphi_i(b) \qquad i = 1, 2, \ldots, n,$$

that is

$$\sum_{j=1}^{n} \dot{\xi}_j(t) \underbrace{\int_a^b \varphi_j(x)\varphi_i(x)\,dx}_{m_{ij}} + \sum_{j=1}^{n} \xi_j(t) \underbrace{\int_a^b c(t,x)\varphi_j'(x)\varphi_i'(x)\,dx}_{a_{ij}(t)}$$
$$= \underbrace{\int_a^b f(x,t)\varphi_i(x)\,dx + q_b(t)\varphi_i(b)}_{b_i(t)} \qquad i = 1, 2, \ldots, n,$$

This is a system of first order ordinary differential equations for $\boldsymbol{\xi} = (\xi_1(t), \ldots, \xi_n(t))$. We rewrite it in a (simpler) matrix form

$$\boldsymbol{M}\dot{\boldsymbol{\xi}}(t) + \boldsymbol{A}(t)\boldsymbol{\xi}(t) = \boldsymbol{b}(t) \tag{3}$$

where $\boldsymbol{M}$ is the *mass matrix*, and $\boldsymbol{A}$ and $\boldsymbol{b}$ are the stiffness matrix and loadvector as before, but now being time-dependent. The entries in $\boldsymbol{M}$, $\boldsymbol{A}$ and $\boldsymbol{b}$ are

$$\begin{aligned} m_{i,j} &= \int_a^b \varphi_j(x)\varphi_i(x)\,dx \\ a_{i,j}(t) &= \int_a^b c(t,x)\varphi_j'(x)\varphi_i'(x)\,dx \\ b_i(t) &= \int_a^b f(x,t)\varphi_i(x)\,dx + q_b(t)\varphi_i(b). \end{aligned}$$

Note that here we have not taken into account the effects of Dirichlet boundary conditions on the matrix $\boldsymbol{A}$ and the vector $\boldsymbol{b}$. The stiffness matrix $\boldsymbol{A}$ and the load vector $\boldsymbol{b}$ are computed as for the stationary problem, but are now time-dependent.

2.2. **Using a diagonal mass matrix.** We will replace the exact mass matrix with an approximation, again making use of the trapezoidal rule to compute the integral. Recall that the basis functions have the property

$$\phi_i(x_j) = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j. \end{cases}$$

The only nonzero entries in the *approximate mass matrix* $\boldsymbol{M}$ are then the diagonal entries

$$m_{i,i} \approx \begin{cases} h/2 & \text{if } i = 1 \text{ or } i = n \\ h & \text{if } 2 \leq i \leq n-1. \end{cases}$$

For example, for $i = 2, \dots, n-1$,

$$\begin{aligned} m_{i,i} = \int_a^b \varphi_i\varphi_i\,dx &= \int_{x_{i-1}}^{x_{i+1}} (\varphi_i)^2\,dx = \int_{x_{i-1}}^{x_i} (\varphi_i)^2\,dx + \int_{x_i}^{x_{i+1}} (\varphi_i)^2\,dx \\ &\approx \frac{h}{2}\Big[\varphi_i(x_{i-1})^2 + \varphi_i(x_i)^2\Big] + \frac{h}{2}\Big[\varphi_i(x_i)^2 + \varphi_i(x_{i+1})^2\Big] = \frac{h}{2} + \frac{h}{2} = h \end{aligned}$$

Note that for the original mass matrix either $m_{ij} \approx 0$ or $m_{ij} \approx 0$ for $i \neq j$ and therefore we set them equal to 0 in the approximate mass matrix.

2.3. **Discretizing in time.** The semi-discrete system (3) have to be discretized in time. Suppose that $0 = t_1 < t_2 < \cdots < t_m = T$ is a partitioning of the time domain, and let $\boldsymbol{\xi}^l = \boldsymbol{\xi}(t_l)$, $l = 1, \dots, m$. Note that $\boldsymbol{\xi}^l = \boldsymbol{\xi}(t_l) = (\xi_1^l, \xi_2^l, \dots, \xi_n^l)^\top$ is the unknown vector we are looking for. For simplicity we will use the θ-scheme with uniform time-step, denoted by $k > 0$. We then get

$$\boldsymbol{M}\left(\frac{\boldsymbol{\xi}_{l+1} - \boldsymbol{\xi}_l}{k}\right) = \theta\boldsymbol{b}(t_l) + (1-\theta)\boldsymbol{b}(t_{l+1}) - \theta\boldsymbol{A}(t_l)\boldsymbol{\xi}(t_l) - (1-\theta)\boldsymbol{A}(t_{l+1})\boldsymbol{\xi}(t_{l+1}).$$

After rearranging the terms, we get an equation we can solve for $\boldsymbol{\xi}_{l+1}$:

$$\big(\boldsymbol{M} + (1-\theta)k\boldsymbol{A}(t_{l+1})\big)\boldsymbol{\xi}^{l+1} = \big(\boldsymbol{M} - \theta k\boldsymbol{A}(t_l)\big)\boldsymbol{\xi}^l + k(\theta\boldsymbol{b}(t_l) + (1-\theta)\boldsymbol{b}(t_{l+1})), \quad (4)$$

$l = 1, \dots, m-1$. That is, for each $l = 1, \dots, m-1$, we solve a linear system

$$\tilde{\boldsymbol{A}}\boldsymbol{\xi}^{l+1} = \tilde{\boldsymbol{b}},$$

where

$$\begin{aligned}\tilde{\boldsymbol{A}} &= \boldsymbol{M} + (1-\theta)k\boldsymbol{A}(t_{l+1})\\ \tilde{\boldsymbol{b}} &= \big(\boldsymbol{M} - \theta k\boldsymbol{A}(t_l)\big)\boldsymbol{\xi}^l + k(\theta\boldsymbol{b}(t_l) + (1-\theta)\boldsymbol{b}(t_{l+1})).\end{aligned}$$

Taking $\theta = 1$ results in a forward (or explicit) Euler scheme, $\theta = 0$ corresponds to the backward (or implicit) Euler scheme, while $\theta = 1/2$ corresponds to Crank-Nicolson scheme. We can solve (4) sequentially for each $\boldsymbol{\xi}^{l+1}$, starting from $\boldsymbol{\xi}^1 = (\xi_1^1, \xi_2^1, \ldots, \xi_n^1)^\top$ determined by the initial condition $u(x, 0) = g(x)$. As we indicated in the lecture notes, there are several ways of choosing the initial vector. Here, we simply choose $\boldsymbol{\xi}^1 = (g(x_1), g(x_2), \ldots, g(x_n))^\top$, which amounts to using a linear spline interpolation of the initial function.

2.4. Handling Dirichlet boundary conditions.

In (4) the Neumann boundary conditions are accounted for in the vector $\boldsymbol{b}$, but we still need to impose the Dirichlet boundary condition. Recall that the full system that we have to solve in each time step reads as

$$\tilde{\boldsymbol{A}}\boldsymbol{\xi}^{l+1} = \tilde{\boldsymbol{b}}, \tag{5}$$

where

$$\begin{aligned}\tilde{\boldsymbol{A}} &= \boldsymbol{M} + (1-\theta)k\boldsymbol{A}(t_{l+1})\\ \tilde{\boldsymbol{b}} &= \big(\boldsymbol{M} - \theta k\boldsymbol{A}(t_l)\big)\boldsymbol{\xi}^l + k(\theta\boldsymbol{b}(t_l) + (1-\theta)\boldsymbol{b}(t_{l+1})).\end{aligned}$$

We impose the Dirichlet boundary condition the same as we did for the stationary problem in the previous lab. That is, we set the first row in $\tilde{\boldsymbol{A}}$ to (1,0,0,. . . , 0), and set the value of the first entry of $\tilde{\boldsymbol{b}}$ to $u_a(t_{l+1})$.

Note that this means that the Dirichlet and Neumann boundary conditions are handled at different times in the procedure. The Neumann boundary condition contributes early to the load vector $\boldsymbol{b}$, but the Dirichlet boundary condition is imposed on the full system (5) later.

2.5. Error analysis for the time-stepping scheme.

The error at the final time T can be estimated according to

$$\|u(T) - u_h(T)\|_2 \leq C_1 h^2 + C_2 k^r, \tag{6}$$

where $r = 1$ for forward and backward Euler, and $r = 2$ for Crank-Nicolson. The two terms on the right (6) comes from spatial and temporal discretisation errors, respectively. This estimate holds in general for $\theta \in [0, 1/2]$ but for $\theta \in (1/2, 1]$ it is only valid when $k \ll h$. This relates to the fact that the θ-method is A-stable if and only if $\theta \in [0, 1/2]$ and (3) is stiff.

When we want to test our implementation of a time-stepping scheme with the method of manufactured solutions, it is often convenient to choose a problem where there is no spatial discretion error. This happens when the exact solution can be exactly represented in the finite element basis. For example, if the exact solution is a line, then we make no error when approximating it with a piecewise linear function (see, Exercise 2).

3. The wave equation

The next problem we will look at is a simple wave equation:

$$u_{tt} - cu_{xx} = f \quad \text{in} \quad (a,b) \times (0,T], \tag{7a}$$

where subscripts u_t and u_x denotes partial derivatives with respect to t and x. Here, $c > 0$ is a constant, f is a function (depending on both x and t), and we have a set of Neumann boundary conditions at the endpoints of the interval given by

$$\left.\begin{aligned} u_x(a,t) &= q_a(t) \\ u_x(b,t) &= q_b(t) \end{aligned}\right\} \quad \text{for} \quad t \in (0,T]. \tag{7b}$$

For the wave equation we also need *two* initial conditions:

$$\left.\begin{aligned} u(x,0) &= g(x) \\ u_t(x,0) &= h(x) \end{aligned}\right\} \quad \text{for} \quad x \in (a,b). \tag{7c}$$

3.1. **Solving the wave equation.** We can derive the variational form of the wave equation in exactly the same way as we did for the heat equation, and we can also discretize it the same way. In the end we get a system of second order ODEs that reads as

$$\boldsymbol{M}\ddot{\boldsymbol{\xi}}(t) + \boldsymbol{A}\boldsymbol{\xi}(t) = \boldsymbol{b}(t), \tag{8}$$

where the only difference between (3) and (8) is the order of the time derivative and that $\boldsymbol{A}$ does not depend on t in this case.

One way to solve the system (8) is to reformulate is a first order system by introducing an auxiliary variable $\boldsymbol{\chi} = \dot{\boldsymbol{\xi}}$. The resulting first order system can be written as

$$\begin{aligned} \dot{\boldsymbol{\xi}}(t) &= \boldsymbol{\chi}(t) \\ \boldsymbol{M}\dot{\boldsymbol{\chi}}(t) &= \boldsymbol{b}(t) - \boldsymbol{A}(t)\boldsymbol{\xi}(t). \end{aligned}$$

In matrix notation, we have

$$\begin{bmatrix} \boldsymbol{I} & 0 \\ 0 & \boldsymbol{M} \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{\xi}} \\ \dot{\boldsymbol{\chi}} \end{bmatrix} = \begin{bmatrix} 0 & \boldsymbol{I} \\ -\boldsymbol{A} & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{\xi} \\ \boldsymbol{\chi} \end{bmatrix} + \begin{bmatrix} 0 \\ \boldsymbol{b} \end{bmatrix}. \tag{9}$$

It turns out that this is not a stiff problem and one can also use explicit time stepping schemes to solve it.

4. Exercises

Exercise 1. We consider the following heat equation.

$$\begin{cases} u_t - u_{xx} = 0 & x \in (0,1), \; t \in (0,1], \\ u(0,t) = u(1,t) = 0 & t \in (0,1], \\ u(x,0) = \sin(\pi x) & x \in (0,1). \end{cases} \tag{10}$$

The exact solution is given by $u(x,t) = e^{-\pi^2 t}\sin(\pi x)$.

(a) Write a function `M = MassMatrix(N)` that computes the mass matrix for a uniform partitioning of $(0,1)$ into N subintervals ($h = \frac{1}{N}$) using the approximate mass matrix from Section 2.2. Note that $n = N+1$ is the number of space mesh points.

(b) Partition the time interval $[0,1]$ into M subintervals ($k = \frac{1}{M}$), with $m = M+1$ time mesh points. Write a function `[y]=ApplyIC(g,x)`, where $u(x,0) = g(x)$, $x \in (0,1)$ is the initial value and $x = 0 : h : 1$ is the vector of spatial nodes. Note that `y` is a $n \times 1$ column vector, i.e., $y_i = g(x_i)$, $i = 1,\dots,n$. Define a matrix U as an $n \times m$ matrix for the approximate solution. Write a main function `FEM_HeatEq_1D()` to find the approximate solution U (in a `for` loop, apply (5)).
Note that you need to:
- initiate the `for` loop by `ApplyIC` function,
- update $\tilde{\boldsymbol{A}}$ and $\tilde{\boldsymbol{b}}$ and `ApplyBC` on $\tilde{\boldsymbol{A}}$ and $\tilde{\boldsymbol{b}}$) in each iteration. Observe that here both boundary conditions are Dirichlet!

(c) Solve (10) using the following values for the parameters θ, M and N. Plot the exact and approximate solutions at the final time $t = 1$ (you can use `subplot` to plot them in a 2 by 1 figure).

θ	1			0.5			0		
M	10	10	200	10	10	200	10	10	200
N	10	200	10	10	200	10	10	200	10

For which values of θ, M and N the approximate solution is not correct? Explain why.

Exercise 2. We consider the following heat equation.

$$\begin{cases} u_t - u_{xx} = 3x\exp(-3t) & x \in (0,1),\ t \in (0,1], \\ u(0,t) = 0 & t \in (0,1], \\ u_x(1,t) = 1 - \exp(-3t) & t \in (0,1], \\ u(x,0) = 0 & x \in (0,1). \end{cases} \tag{11}$$

The exact solution is given by $u(x,t) = x(1 - \exp(-3t))$.

(a) Solve (11), using `FEM_Heat_1D` (from parts (a) and (b) in Exercise 1) with the parameters in part (c) of Exercise 1.
(b) Determine experimentally the convergence rate r in (6) for the backward Euler and the Crank-Nicolson schemes. Note that as the exact solution is a line only the time discretisation contribute to the error in (6). Since the error does not depend on the spatial discretisation, you can use a small value of N when computing errors and rates, for example N = 4.

Exercise 3. Consider the wave equation

$$u_{tt} - u_{xx} = 0 \quad \text{in} \quad (0,1) \times (0,1], \tag{12a}$$

with boundary conditions

$$\left.\begin{aligned} u_x(0,t) &= \pi\cos(\pi t) \\ u_x(1,t) &= \pi\cos(\pi - \pi t) \end{aligned}\right\} \quad \text{for} \quad t \in (0,1]. \tag{12b}$$

and initial conditions

$$\left.\begin{aligned} u(x,0) &= \sin(\pi x) \\ u_t(x,0) &= -\pi\cos(\pi x) \end{aligned}\right\} \quad \text{for} \quad x \in (0,1). \tag{12c}$$

The exact solution is given by

$$u(x,t) = \sin(\pi x - \pi t).$$

Solve (12) with the finite element method in space and a second order explicit Runge-Kutta method in time. Construct the two initial vectors the same way as for the heat equation. Plot and compare the solution and the approximate solution at the final time $t = 1$.