FINITA ELEMENTMETODEN I 1D – DEL 2
TIDSBEROENDE RANDVÄRDESPROBLEM

I denna datorlaboration kommer vi att studera finita elementmetoden för tidsberoende värmeledningsekvationen i en dimension

$$\begin{cases} \partial_t u - \partial_x(a(x)\partial_x u) = f(t,x), \quad t \in (0,T),\ x \in (0,L), \\ u(t,0) = q_0(t),\ u(t,L) = q_L(t), \quad t \in (0,T), \\ u(0,x) = u_0(x), \quad x \in (0,L). \end{cases} \tag{1}$$

Den okända funktionen $u(t,x)$ beror både på tiden t och rumsvariabeln x. Värmeledningskoefficienten a, randvillkoren q_0, q_L, funktionen f och begynnelsevärdet u_0 är givna funktioner.

1. Reduktion till ett ODE-system

Vi börjar med att skriva en svag formulering till (1) som lyder: Bestäm $u(t,x)$ sådan att $u(t,0) = q_0, u(t,L) = q_L$ och $u(0,x) = u_0(x)$ och som uppfyller

$$\int_0^L \partial_t u\, v\, dx + \int_0^L a(x)\partial_x u\, \partial_x v\, dx = \int_0^L f\, v\, dx, \tag{2}$$

för varje $t \in (0,T)$ och varje $v \in C^1[0,L]$ sådan att $v(0) = v(L) = 0$.

1.1. **Diskretisering av rumsvariabeln.** Låt oss ta ett interval $I = [0,L]$ och införa ett jämnt fördelat beräkningsnät som består av N noder $0 = x_1 < x_2 < \cdots < x_N = L$, med steget $h = x_{i+1} - x_i = \frac{1}{N-1}$. Som för elliptiska problem inför vi ett ändligtdimensionellt delrum V_h:

$$V_h = \{u_h \in C[0,1] : u_h \text{ är affine för } x \in [x_i, x_{i+1}],\ 1 \le i \le N\}. \tag{3}$$

Testfunktioner ska uppfylla homogena Dirichlets randvillkor och tillhöra delrummet

$$V_{0h} = \{v_h \in C[0,1] :\ v_h(0) = v_h(L) = 0 \text{ och } v_h\Big|_{[x_i,x_{i+1}]} \in \mathbb{P}_1,\ 1 \le i \le N\}. \tag{4}$$

Variationsformuleringen till approximationen lyder:
Bestäm u_h så att $u_h(t,0) = q_0(t), u_h(t,L) = q_L(t)$, $u_h(0,x) = u_{0h}(x)$ sådan att

$$\boxed{\int_0^L \partial_t u_h\, v_h\, dx + \int_0^L a(x)\partial_x u_h(x)\partial_x v_h(x)\, dx = \int_0^L f(t,x)v_h(x)\, dx, \quad \text{för alla } v_h \in V_{0h},} \tag{5}$$

och alla t. Här är u_{0h} en lämplig approximation av begynnelsevärdet $u_0(x)$. I V_h väljer vi styckvis affina basfunktioner genom att sätta $\phi_i(x_i) = 1$ och $\phi_i(x_j) = 0$ för $j \neq i$ som illustrerat i Figur 1.

Date: September 18, 2020. Irina Pettersson, MV CTH/GU.

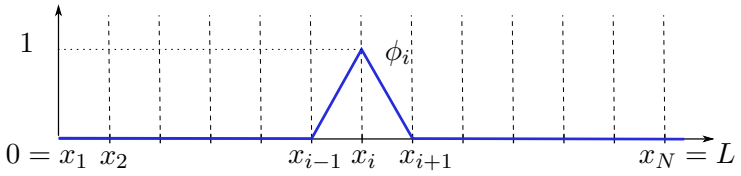

FIGURE 1. Basfunktion $\phi_i(x)$

Vi letar efter en lösning till (5) på formen

$$\boxed{u_h(x) = \sum_{j=1}^{N} U_j(t)\phi_j(x)}.$$

På samma sätt sätter vi $u_0(x) = \sum_{j=1}^{N} u_0(x_j)\phi_j(x)$. Låt oss ta testfunktionen $v_h = \phi_i$, $2 \leq i \leq (N-1)$ och sätta in i (5):

$$\sum_{j=1}^{N} \dot{U}_j(t) \underbrace{\int_0^L \phi_j(x)\,\phi_i(x)\,dx}_{M_{ij}} + \sum_{j=1}^{N} U_j(t) \underbrace{\int_0^L a(x)\phi_j'\phi_i'\,dx}_{A_{ij}} = \underbrace{\int_0^L f\phi_i\,dx}_{b_i}, \quad 2 \leq i \leq (N-1). \tag{6}$$

Här är $\dot{U}$ derivatan av U med avseende på t och ϕ' är derivatan av ϕ med avseende på x. Notera att testfunktionen v_h måste uppfylla homogena Dirichlet-villkor vid ändpunkterna, varför vi inte kan ta ϕ_1 och ϕ_N som testfunktioner i (5). På grund av detta fick vi $(N-2)$ ekvationer i (6) och inte N som i Neumans eller Robins problem. Beteckna

$$\begin{aligned}
M_{ij} &= \int_0^L \phi_j\phi_i\,dx, \quad i = 2,\ldots,(N-1);\ j = 1,\ldots N;\\
A_{ij} &= \int_0^L a(x)\phi_j'\phi_i'\,dx, \quad i = 2,\ldots,(N-1);\ j = 1,\ldots N;\\
b_i &= \int_0^L f\phi_i\,dx, \quad i = 2,\ldots(N-1).
\end{aligned}$$

Vi vill skriva om (6) på matrisform, men innan vi gör detta behöver vi diskretisera med avseende på tiden och ta hänsyn till randvillkoren och begynnelsevärdet.

Elementen i massmatrisen kan vi räkna ut exakt:

$$M_{11} = M_{NN} = \frac{h}{3}, \tag{7}$$

$$M_{ii} = \int_{x_{i-1}}^{x_{i+1}} (\phi_i)^2 \, dx = \frac{2h}{3}, \quad i = 2, \dots (N-1); \tag{8}$$

$$M_{i(i+1)} = M_{(i+1)i} = \int_{x_i}^{x_{i+1}} \phi_i \phi_{i+1} \, dx = \frac{h}{6}, \quad i = 1, \dots (N-1). \tag{9}$$

Som i förra datorlaborationen använder vi trapesformeln för att räkna ut integralerna i matrisen A och lastvektorn b approximativt. Om man ser bort från randvillkoren (vi kommer att ta hand om dem efter att vi har diskretiserat i tiden) så har vi

$$A_{ii} = \int_0^L a(x)\phi_i'\phi_i' \, dx \approx \frac{a(x_{i-1}) + 2a(x_i) + a(x_{i+1})}{2h}, \quad i = 1, \dots N; \tag{10}$$

$$A_{i(i+1)} = A_{(i+1)i} = \int_0^L a(x)\phi_i'\phi_{i+1}' \, dx \approx -\frac{a(x_i) + a(x_{i+1})}{2h}, \quad i = 1, \dots (N-1); \tag{11}$$

$$A_{ij} = 0 \quad j \neq i \pm 1. \tag{12}$$

$$b_i(t) = \int_0^L f(t,x)\phi_i(x) \, dx \approx h f(t, x_i), \quad i = 1, \dots N. \tag{13}$$

På det sättet får vi ett system av ODE på matrisform

$$\begin{cases} M\dot{U} + AU = b(t), \\ U(0) = U^0 \end{cases}$$

med konstanta $N \times N$ mass- och styvhetsmatrisen M, A och N-lastvektorn $b(t)$. Notera att M och A är trediagonala matriser. Den obekanta vektorn är $U(t) = (U_1(t), \dots, U_N(t))$ och begynnelsevärdet är $U^0 = (u_0(x_1), \dots, u_0(x_N))$.

1.2. **Diskretisering i tidsvariabeln.** Vi har diskretiserat problemet med avseende på rumsvariabeln och nu ska vi diskretisera det i tidsvariabeln med hjälp av **finita differenser**.
Tag en partition av $[0, T]$: $0 = t_1 < t_2 < \ldots < t_M = T$ med steget $k = t_{m+1} - t_m$. Vi vill lösa approximativt ett system av ODE på formen

$$\begin{cases} M\dot{U} + AU = b(t), \\ U(0) = U^0 \end{cases}$$

med konstanta $N \times N$ mass- och styvhetsmatrisen M, A. Låt oss diskretisera med hjälp av **θ-scheme**:

$$M\frac{U(t_{m+1}) - U(t_m)}{k} + A(\theta U(t_{m+1}) + (1-\theta)U(t_m)) = \theta b(t_{m+1}) + (1-\theta)b(t_m).$$

Parametern θ kan vara 0 (framåtdifferens), 1 (bakåtdifferens) eller 0.5 (centraldifferens, Crank-Nicolson).

Systemet att lösa för $U(t_{m+1})$ blir

$$\boxed{\tilde{A}U(t_{m+1}) = \tilde{b}},$$

där

$$\tilde{A} = M + \theta k A, \tag{14}$$

$$\tilde{b} = (M - (1-\theta)kA)U(t_m) + \theta\, k\, b(t_{m+1}) + (1-\theta)\, k\, b(t_m),\ m = 1, \ldots, M-1. \tag{15}$$

Vi startar med begynnelsevärdet $\boxed{U(t_1) = (u_0(x_1), u_0(x_2), \ldots, u_0(x_N))}$ och löser sedan ekvationen för $U(t_{m+1})$, $m = 1, \ldots, M-1$. För att ta hand om randvillkoren $u(t,0) = q_0(t)$, $u(t,L) = q_L(t)$ ersätter vi första och sista raderna i $\tilde{A}$ samt första och sista komponenterna i $\tilde{b}$ (för varje m) på samma sätt som vi gjorde för elliptiska problem:

$$\tilde{A}_{11} = 1, \quad \tilde{A}_{1j} = 0,\ j \neq 1;$$

$$\tilde{A}_{NN} = 1, \quad \tilde{A}_{Nj} = 0,\ j \neq N; \tag{16}$$

$$\tilde{b}_1 = q_0(t_{m+1}), \quad \tilde{b}_N = q_L(t_{m+1}). \tag{17}$$

2. Uppgifter

Uppgift 1

Vi vill studera en tidsberoende värmeledningsekvation i en stav gjort av ett kompositmaterial som kommer att bestå av två olika material med olika värmeledningsegenskaper. Låt oss införa en värmeledningskoeffcient. Tag den styckvis konstanta funktionen $a(x)$

$$a(x) = \begin{cases} 1, & 0 \leq x < 0.5, \\ 2, & 0.5 \leq x < 1. \end{cases} \tag{18}$$

Betrakta följande Dirichlet randvärdesproblem i en heterogen endimensionell stav:

$$\begin{cases} \partial_t u - \partial_x\big(a\big(\frac{x}{\varepsilon}\big)\partial_x u(t,x)\big) = 0, & t \in (0,1],\ x \in (0,1), \\ u(t,0) = 0,\ u(t,1) = 0, & t \in (0,1], \\ u(0,x) = \sin(\pi x), & x \in (0,1). \end{cases} \tag{19}$$

(1) Modifiera funktionen `K=StiffnessMatrix(x,ve)` från uppgift 2 i Lab 1 (så att du får med den oscillerande koefficienten) så att den använder `spdiags` eller `sparse` för att definiera den glesa matrisen A enligt (10)–(12).

(2) Modifiera funktionen `b=LoadVector(x)` från första laborationen så att den har två input-parametrar t, x och räknar ut lastvektorn för en funktion $f(t,x)$. Här ska x vara en vektor, t en skalär och output `LoadVector` ska vara en vektor med lika många komponenter som x.

(3) Definiera en partition av rumsintervallet med $N = 10$ noder och steget $h = 1/(N-1)$, och en partition av tidsintervallet $[0, 1]$ med $S = 10$ noder och steget $k = 1/(S-1)$. Definiera också en matris för den approximativa lösningen `U=zeros(N,S)`. Sätt $\varepsilon = 2^{-3}$ och $\theta = 1$. Definiera randvillkoren $q_0(t), q_L(t)$ och begynnelsevärdetvektorn $u_0(x)$ enligt (19). Du kan t ex använda följande kod:

```
% Define a function for the initial condition
u0=@(x)sin(pi*x);
% Define a column vector of initial condition
U0 = u0(x)';
% Write the initial vector in the solution matrix
U(:,1) = U0;
```

(4) Definiera massmatrisen M enligt (7)–(9) (använd `spdiags` eller `sparse`). Räkna ut matrisen $\tilde{A}$ given med (14).

(5) I en loop `for m=1:S-1` (dvs för varje t) räkna ut $U(t_{m+1})$ (en vektor `U(:,m+1)` med N koordinater) som uppfyller

$$\tilde{A}U(t_{m+1}) = \tilde{b},$$

med en $N \times N$ matris $\tilde{A}$ (14)–(16) och en N-vektor $\tilde{b}$ (15) i Kap. 1.2. För matrismultiplikationen i Matlab använder man $*$ och för att lösa ekvationssystemet `AU = b` använder man `U=A\b`. Glöm inte kontrollera att alla vektorer är kolumnvektorer. Du kan t ex använda följande kod för att räkna ut $\tilde{b}$:

```
tildeB=M*U(:,m)-(1-theta)*k*StiffnessMatrix(x,ve)*U(:,m)...
        + theta*k*LoadVector(t(m+1),x)' ...
        + (1-theta)*k*LoadVector(t(m),x)';
tildeB(1) = q0;
tildeB(N) = qL;
```

Om `LoadVector` är en radvektor så den måste transponeras.

Begynnelsevärdet ska uppdateras efter varje iteration: `U(:,m) = U(:,m+1)`. Notera att matrisen $\tilde{A}$ inte beror på tiden och kan räknas ut en gång, medan vektorn $\tilde{b}$ kan bero på t och måste uppdateras vid varje iteration.

Plotta den approximativa lösningen `U(:,m+1)` vid varje iteration.

```
plot(x,U(:,m+1),'r','Linewidth', 2);
lgd=legend(sprintf('Temperature␣at␣t=%.2f',t(m+1)));
lgd.FontSize = 14;
ylim([0,1.2]);
```

(6) Skillnaden mellan den exakta lösningen u och den approximativa u_h vid sluttiden T kan uppskattas i termer av steglängderna h och k:

$$\|u(T) - u_h(T)\| \leq C_1 h^s + C_2 k^r,$$

för några s, r. Denna uppskattning gäller för alla h, k om $\theta \in [0.5, 1]$, och endast för $k << h$ om $\theta = 0$.

Lös (19) för följande värden av parametrar θ, M, N

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

För vilka värden uppstår instabilitet?

Uppgift 2

För små ε kan lösningen till (19) approximeras med lösningen till det effektiva problemet

$$\begin{cases} \partial_t v - \dfrac{4}{3}\partial_{xx} v(t,x) = 0, & t \in (0,1],\ x \in (0,1), \\ v(t,0) = 0,\ v(t,1) = 0, & t \in (0,1], \\ v(0,x) = \sin(\pi x), & x \in (0,1). \end{cases} \tag{20}$$

Man kan räkna ut lösningen exakt (separation av variabler):

$$v(t,x) = e^{-\frac{4\pi^2}{3}t} \sin(\pi\, x). \tag{21}$$

Plotta $v(t,x)$ för varje t_m på samma plot som den approximativa lösningen $U(:,m)$ i Uppgift 1. Ser det ut som att v approximerar U?

Om du vill göra en animation (valfritt) så kan du använda följande kod i en `for`-loop:

```
% Plot the initial profile
plot(x,U0,'r','Linewidth',2);
lgd=legend(sprintf('Temperature␣profile␣at␣t␣=␣%.2f', t(1)));
lgd.FontSize = 14;
ylim([0,1.2]);
% Save the plot for the animation
P(1) = getframe;
for m=1:S-1
        % ...................
        % Compute the approximation U at t=t_m
        U(:,m+1) = tildeA\tildeB;
        % Plot the solution for each t
        plot(x,U(:,m+1),'Linewidth', 2);
        lgd=legend(sprintf('Temperature␣at␣t=%.2f',t(m+1)));
        lgd.FontSize = 14;
        ylim([0,1.5]);
        P(m+1) = getframe;
end
video = VideoWriter('Heat1D.avi', 'Uncompressed␣AVI');
video.FrameRate = 5;
open(video)
writeVideo(video, P);
close(video);
```