

TMA373 Assignment 5

Edoardo Mangia
edoardom@student.chalmers.se

Bibiana Farinha
bibiana@student.chalmers.se

February 2025

Contents

1 Task 1 1

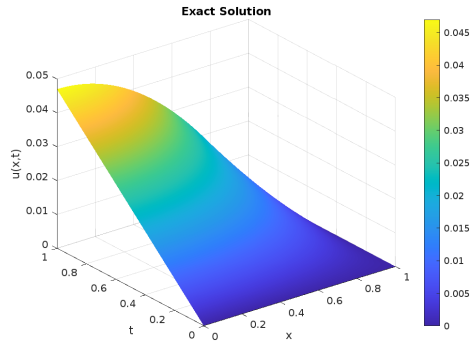
1 Task 1

In this assignment, we will implement and study the FE approximation of the heat equation.

$$\begin{cases} u_t(x, t) + u_{xx}(x, t) = f(x, t), & 0 < x < 1, \ 0 < t < T, \\ u_x(0, t) = 0, \quad u(1, t) = 0, & 0 < t < T, \\ u(x, 0) = u_0(x), & 0 < x < 1. \end{cases} \quad (1)$$

To be able to test our implementation, we will need to first calculate $f(x, t)$ and $u_0(x)$. Assume that $u(x, t) = -\frac{\text{epsSTUD}}{2}(tx^2 - t)$, with $\text{epsSTUD} = 47/10^3$, from (1), we can determine the functions:

$$f(x, t) = \frac{\text{epsSTUD}}{2}(1 - x^2 - 2t), \quad u_0(x) = 0.$$



From the lessons, we know the values of the mass matrix M and the stiffness matrix S . In the implementation below we calculate an approximation of the load vector F by using the trapezoidal rule for each of the elements of the vector. The only task missing is solving the linear equation (12) for each time step.

Implementation

```

1 % Template for linear heat equation on (0,1)
2 % Neumann BC at x=0, Dirichlet BC at x=1
3
4 % number of nodes and mesh size and grid points
5 m = 50;
6 h = 1/(m+1);
7 x = 0:h:1;
8
9 % time interval, nbr of steps, stepsize
10 T = 1; n = 100;
11 k = T/n;
12 t = linspace(0, T, n+1);
13
14 % for the solution matrix with all discrete space and time points
    (inc. BC)
15 U = zeros(m+2, n+1);
16
17 % IC and BC for time t=0
18 epsSTUD = 47/10^3;
19 U(:,1) = 0;
20 U(end,1) = 0;
21
22 % construction of mass and stiffness matrices
23 M = zeros(m+1, m+1);
24 S = zeros(m+1, m+1);
25 M_local = (h/6)*[2 1; 1 2];
26 S_local = (1/h)*[1 -1; -1 1];
27 for elem = 1:m+1
28     if elem <= m+1 && elem+1 <= m+1
29         M(elem:elem+1, elem:elem+1) = M(elem:elem+1, elem:elem+1)
            + M_local;
30         S(elem:elem+1, elem:elem+1) = S(elem:elem+1, elem:elem+1)
            + S_local;
31     elseif elem <= m+1
32         M(elem, elem) = M(elem, elem) + M_local(1,1);
33         S(elem, elem) = S(elem, elem) + S_local(1,1);
34     end
35 end
36
37 A = M + k * S;
38
39 % time discretisation
40 for l = 1:n
41     t1 = t(l+1);
42     f_fun = @(x) -epsSTUD*(x.^2 - 1) + 2*epsSTUD*t1;
43     F_vec = zeros(m+1,1);
44     for i = 1:(m+1)
45         if i == 1
46             xi_local = x(1:2);

```

```

47     phi = @(xi) (xi - x(1))/h;
48     F_vec(i) = trapz(xi_local,
49         f_fun(xi_local).*phi(xi_local));
50     elseif i == m+1
51         xi_local = x(m+1:end);
52         phi = @(xi) (x(end) - xi)/h;
53         F_vec(i) = trapz(xi_local,
54             f_fun(xi_local).*phi(xi_local));
55     else
56         xi_local1 = x(i:i+1);
57         phi1 = @(xi) (xi - x(i))/h;
58         I1 = trapz(xi_local1,
59             f_fun(xi_local1).*phi1(xi_local1));
60         xi_local2 = x(i+1:i+2);
61         phi2 = @(xi) (x(i+2) - xi)/h;
62         I2 = trapz(xi_local2,
63             f_fun(xi_local2).*phi2(xi_local2));
64         F_vec(i) = I1 + I2;
65     end
66 end
67 xi_old = U(1:m+1, 1);
68 xi_new = A \ (M*xi_old + k * F_vec);
69 U(1:m+1, 1+1) = xi_new;
70 U(end, 1+1) = 0;
71 end
72 % ... plotting

```

Visualization

