HW1

20250925

-by Yimei Li

HW1_1

matlab code

```
clear; clc;

%% Problem statement parameters setting
k = 0.49;
C = 0.2174;
rho = 2.7;
kappa = k / (rho * C);
dx = 2;
dt = 0.1;
L = 10;
t_total = 0.2;
Nx = L / dx + 1
Nt = t_total / dt;

%% Temperature field initialization
T = zeros(Nx, Nt + 1);
x = linspace(0, L, Nx);
T(:, 1) = 0;
T(1, :) = 100;

%% Crank-Nicolson method parameters calculation
c = kappa * dt / (dx^2);
alpha = c / 2;
beta = 1 + c;
main_diag = beta  ones(Nx, 1);
lower_diag = -alpha  ones(Nx-1, 1);
upper_diag = -alpha * ones(Nx-1, 1);
main_diag(1) = 1;
upper_diag(1) = 0;
lower_diag(end) = -alpha;
main_diag(end) = 1 + alpha;
upper_diag(end) = -alpha;
A = diag(main_diag) + diag(lower_diag, -1) + diag(upper_diag, 1);

%% Time-iterative solution
for n = 1:Nt
    b = zeros(Nx, 1);
    b(1) = T(1, n);
```

```
    for i = 2:Nx-1
        b(i) = alpha  T(i-1, n) + (1 - c)  T(i, n) + alpha  T(i+1, n);
    end
    b(end) = alpha  T(end-1, n) + (1 - c)  T(end, n) + alpha  T(end-1, n);
    T(:, n+1) = A </span> b;
end

%% Result output and visualization
fprintf('Temperature of each node at t = 0.1s（x/cm → T/°C）: \n');
for i = 1:Nx
    fprintf('x=%d → T=%.4f\n', x(i), T(i, end));
end
figure;
plot(x, T(:, end), 'o-', 'LineWidth', 1.5, 'MarkerSize', 8);
xlabel('Position x (cm)', 'FontSize', 11);
ylabel('Temprature T (°C)', 'FontSize', 11);
title('Temperature distribution at t = 0.1s', 'FontSize', 12);
grid on;
```

Result & Output

Temperature of each node at t = 0.1s（x/cm → T/°C）:

x=0 → T=100.0000

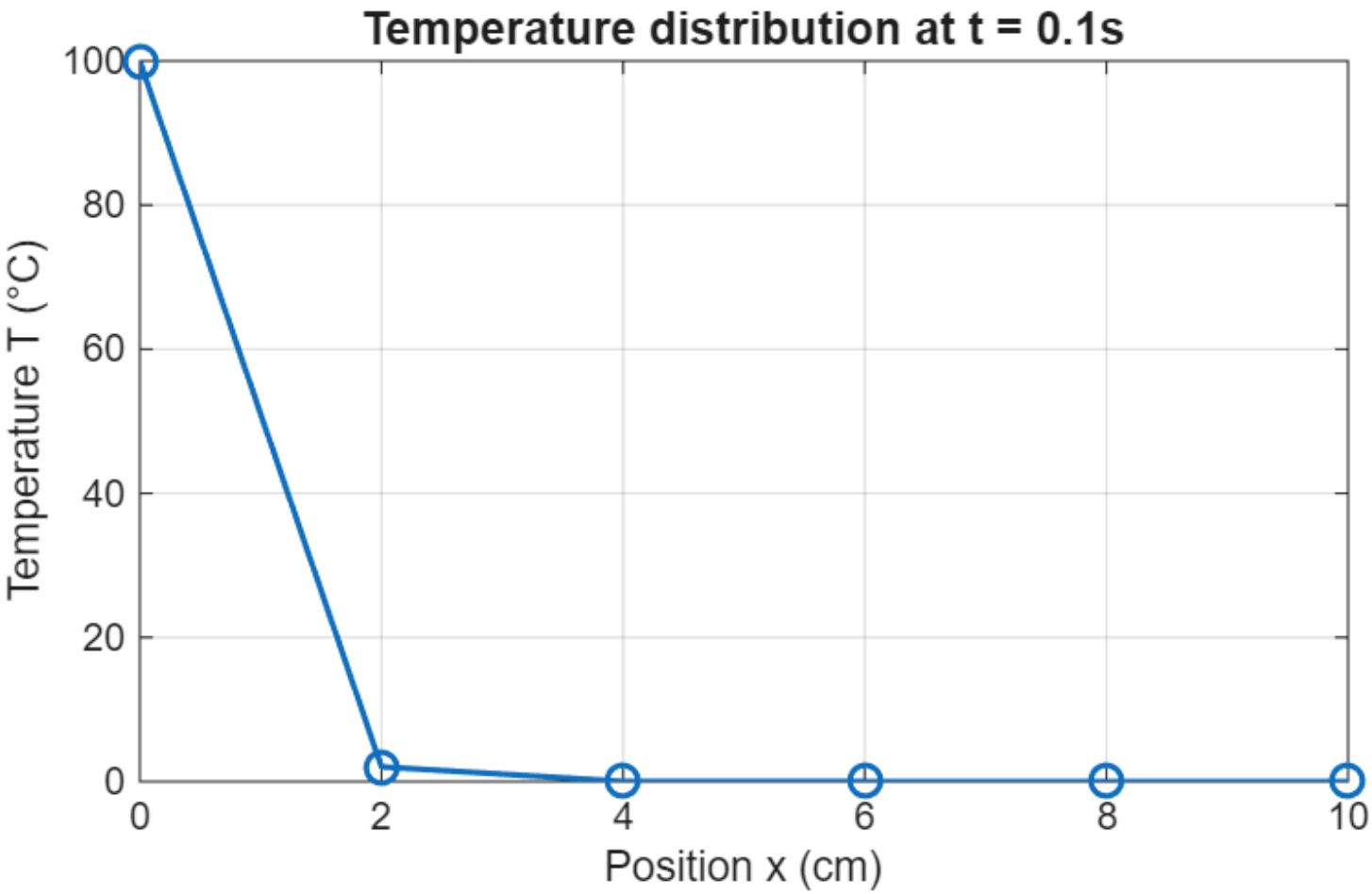
x=2 → T=2.0445

x=4 → T=0.0209

x=6 → T=0.0002

x=8 → T=0.0000

x=10 → T=0.0000


Temperature distribution at t = 0.1s
Temperature T (°C)
Position x (cm)
100
80
60
40
20
0
0
2
4
6
8
10

hw1_2

MATLAB code

```
clear; clc;

%% Problem statement parameters setting
L = 0.5;
H = 0.3;
Nx = 50;
Ny = 50;
dx = L / (Nx - 1);
dy = H / (Ny - 1);
T_initial = 30;
T_S1 = 100;
T_S2 = 200;
max_iter = 100000;
tol = 1e-6;
err = 1;
iter = 0;

%% Initialize temperature field
T = T_initial * ones(Ny, Nx);
T(:, 1) = T_S1;
T(1, :) = T_S2;

%% Iterative solution
while iter < max_iter && err > tol
    err = 0;
    for i = 2:Ny-1
        for j = 2:Nx-1
            T_old = T(i, j);
            if abs(dx - dy) < 1e-10
                beta = 1;
                T(i, j) = 0.25  (T(i+1, j) + T(i-1, j) + T(i, j+1) + T(i, j-1));
            else
                dx2 = dx^2; dy2 = dy^2;
                T(i, j) = ( (T(i+1, j) + T(i-1, j)) / dy2 + (T(i, j+1) + T(i, j-1)) / dx2 ) / (
            end
            err = max(err, abs(T(i,j) - T_old));
        end
    end
```

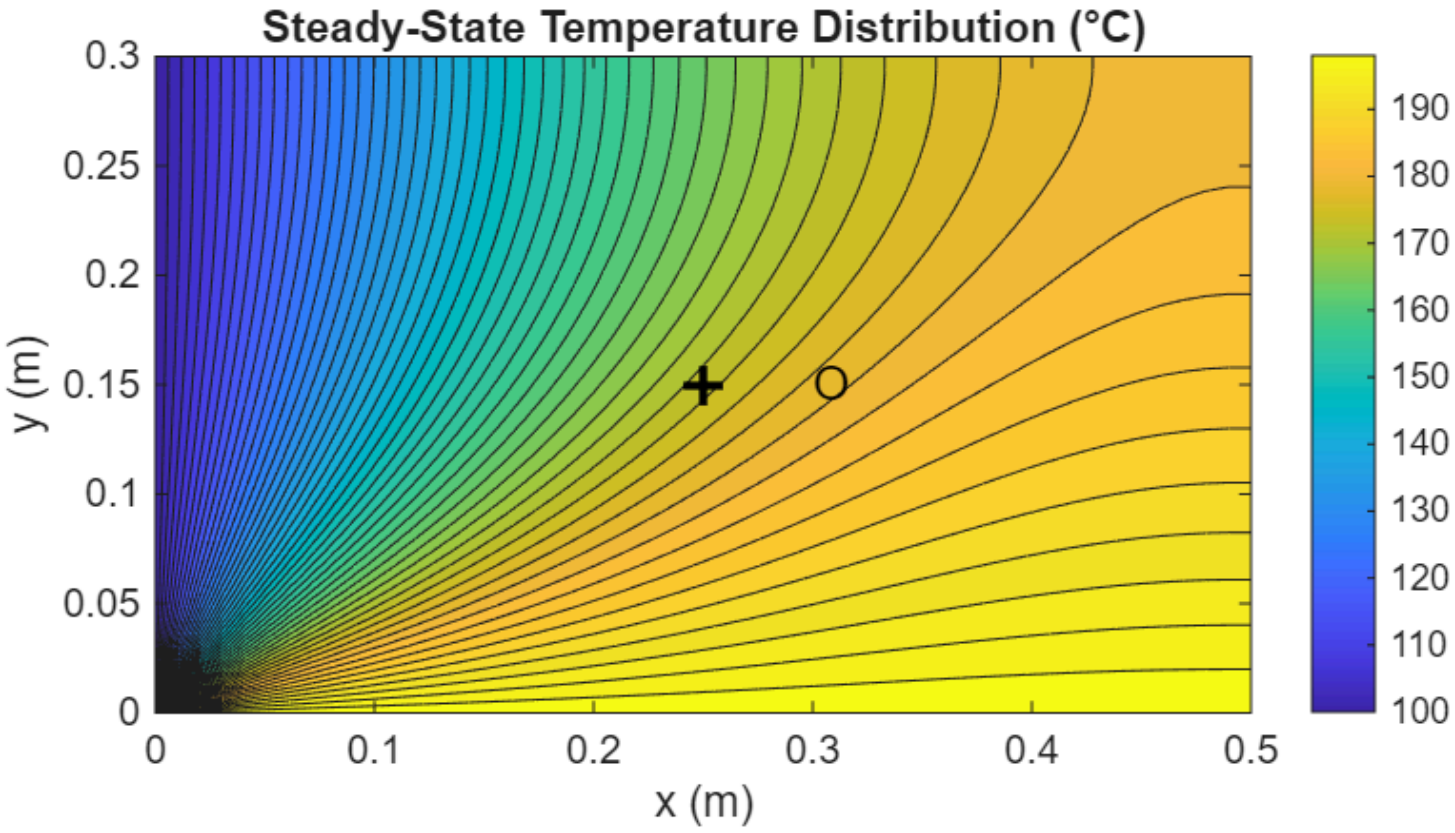
```
    iter = iter + 1;

    T(end, :) = T(end - 1, :);
    T(:, end) = T(:, end - 1);
    T(:, 1) = T_S1;
    T(1, :) = T_S2;
end
fprintf('Converged in %d iterations. Max error = %.2e\n', iter, err);

%% Plot temperature distribution
figure;
contourf(linspace(0,L,Nx), linspace(0,H,Ny), T, 50);
hold on;
colorbar;
xlabel('x (m)');
ylabel('y (m)');
title('Steady-State Temperature Distribution (°C)');
axis equal tight;

%% Compute heat flux at center point O
ix = round(Nx/2);
iy = round(Ny/2);
dTdx = (T(iy, ix+1) - T(iy, ix-1)) / (2dx);
dTdy = (T(iy+1, ix) - T(iy-1, ix)) / (2dy);
qx = -dTdx;
qy = -dTdy;
q_mag = sqrt(qx^2 + qy^2);
fprintf('\nAt center point O (%.3f, %.3f):\n', (ix-1)dx, (iy-1)dy);
fprintf(' dT/dx = %.4f °C/m, dT/dy = %.4f °C/m\n', dTdx, dTdy);
fprintf(' qx = %.4f W/m², qy = %.4f W/m²\n', qx, qy);
fprintf(' |q| = %.4f W/m²\n', q_mag);
```

Result & Output



At center point O:

dT/dx = 128.8489 °C/m, dT/dy = -129.7361 °C/m

qx = -128.8489 W/m², qy = 129.7361 W/m²

|q| = 182.8483 W/m²