Massachusetts Institute of Technology

Project Report

18.0851
Computational Science and Engineering I

Written by
Jered Dominguez-Trujillo

Submitted to:
Prof. Matthew Durey

May 3, 2019

1 Problem Description

One-Dimensional Heat Equation

$$\frac{\partial u}{\partial t} = K\frac{\partial^2 u}{\partial x^2} + q(x,t) \tag{1}$$

Assumptions
1. Uniform Grid Spacing (Δx)
2. Uniform Time Steps (Δt)

Variables
Length of Domain: $L = 2 * \pi$
Left Boundary Condition: $C_1 = 1$
Right Boundary Condition: $C_2 = -0.2$
Thermal Diffusivity: $K = 0.1$
Time Ramp: $t_{ramp} = 1$

Initial Conditions

$$u(x, t = 0) = 0 \tag{2}$$

Boundary Conditions
Dirilecht Condition at the Left Boundary x = 0:

$$u(x = 0, t) = \begin{cases} C_1 * \frac{t}{t_{ramp}} & t \leq t_{ramp} \\ C_1 & t > t_{ramp} \end{cases} \tag{3}$$

Neumann Condition at the Right Boundary x = L:

$$\frac{\partial u}{\partial x}(x = L, t) = C_2 \tag{4}$$

Heating Source Function

$$q(x,t) = -sin(\omega t) * exp\left(\frac{-(x-\frac{L}{2})^2}{\sigma^2}\right) \tag{5}$$

Second Order Finite Difference
At time t, where $u^{(n)} = \frac{\partial^n u}{\partial x^n}$

Taylor Expansions:

$$u(x+\Delta x) = u(x) + \Delta x u^{(1)}(x) + \frac{1}{2}(\Delta x)^2 u^{(2)}(x) + \frac{1}{6}(\Delta x)^3 u^{(3)}(x) + O(h^4) \tag{6}$$

$$u(x-\Delta x) = u(x) - \Delta x u^{(1)}(x) + \frac{1}{2}(\Delta x)^2 u^{(2)}(x) - \frac{1}{6}(\Delta x)^3 u^{(3)}(x) + O(h^4) \tag{7}$$

$$u(x+\Delta x) + u(x-\Delta x) = 2u(x) + (\Delta x)^2 u^{(2)}(x) + O(h^4) \tag{8}$$

$$u^{(2)}(x) = \frac{u(x+\Delta x) - 2u(x) + u(x-\Delta x)}{(\Delta x)^2} + O(h^2) \tag{9}$$

Explicit Euler

Let n denote the time step, and i denote the grid point

$$u_{n+1,i} - u_{n,\imath} = \int_{t_n}^{t_{n+1}} f(t, u(t))dt = \Delta t * f(t_n, u_n) = \Delta t \big[K \frac{\partial^2 u_n}{\partial x^2} + q_n(x)\big]_i \tag{10}$$

$$u_{n+1,i} - u_{n,\imath} = \Delta t \big[K \big(\frac{u_n(x+\Delta x) - 2u_n(x) + u_n(x-\Delta x)}{(\Delta x)^2}\big) + q_n(x)\big]_i \tag{11}$$

Where

$$\begin{aligned} u_n(x+\Delta x) &= u_{n,i+1} \\ u_n(x) &= u_{n,i} \\ u_n(x-\Delta x) &= u_{n,i-1} \end{aligned} \tag{12}$$

$$u_{n+1,i} - u_{n,i} = \Delta t \big[K \big(\frac{u_{n,i+1} - 2u_{n,i} + u_{n,\imath-1}}{(\Delta x)^2}\big) + q_{n,i}(x)\big] \tag{13}$$

$$u_{n+1,i} = u_{n,i} + \Delta t \big[K \big(\frac{u_{n,i+1} - 2u_{n,i} + u_{n,\imath-1}}{(\Delta x)^2}\big) + q_{n,i}(x)\big] \tag{14}$$

We define the CFL Number to be:

$$CFL = K \frac{\Delta t}{(\Delta x)^2} \tag{15}$$

$$\boxed{u_{n+1,i} = CFL * u_{n,i+1} + (1 - 2CFL) * u_{n,i} + CFL * u_{n,i-1} + q_{n,i} \Delta t} \tag{16}$$

Implicit Euler
Let n denote the time step, and i denote the grid point

$$u_{n+1,i} = u_{n,i} + \Delta t * f(t_{n+1}, u_{n+1}) = u_{n,i} + \Delta t\big[K\frac{\partial^2 u_{n+1}}{\partial x^2} + q_n(x)\big]_i \tag{17}$$

$$u_{n+1,i} = u_{n,i} + \Delta t\big[K\big(\frac{u_{n+1}(x+\Delta x) - 2u_{n+1}(x) + u_{n+1}(x-\Delta x)}{\Delta x^2}\big) + q_n(x)\big]_i \tag{18}$$

Where

$$\begin{aligned} u_{n+1}(x+\Delta x) &= u_{n+1,i+1} \\ u_{n+1}(x) &= u_{n+1,i} \\ u_{n+1}(x-\Delta x) &= u_{n+1,i-1} \end{aligned} \tag{19}$$

$$u_{n+1,i} = u_{n,i} + \Delta t\big[K\big(\frac{u_{n+1,i+1} - 2u_{n+1,i} + u_{n+1,i-1}}{\Delta x^2}\big) + q_{n,i}(x)\big] \tag{20}$$

$$u_{n+1,i} - \Delta t\big[K\big(\frac{u_{n+1,i+1} - 2u_{n+1,i} + u_{n+1,i-1}}{\Delta x^2}\big)\big] = u_{n,i} + q_{n,i}(x) \tag{21}$$

$$u_{n+1,i} - \frac{K\Delta t}{(\Delta x)^2}\big[u_{n+1,i+1} - 2u_{n+1,i} + u_{n+1,i-1}\big] = u_{n,i} + q_{n,i}(x) \tag{22}$$

We define the CFL Number to be:

$$CFL = K\frac{\Delta t}{(\Delta x)^2} \tag{23}$$

$$\boxed{-CFL * u_{n+1,i+1} + (2CFL+1) * u_{n+1,i} - CFL * u_{n+1,i-1} = u_{n,i} + q_{n,i}\Delta t} \tag{24}$$

Boundary Conditions

1) Dirilecht Condition at the Left Boundary x = 0:

$$u(x=0,t) = \begin{cases} C_1 * \frac{t}{t_{ramp}} & t \le t_{ramp} \\ C_1 & t > t_{ramp} \end{cases} \tag{25}$$

$$-CFL * f_{BC1} + (2CFL+1) * u_{n+1,2} - CFL * u_{n+1,3} = u_{n,2} + q_{n,2}\Delta t \tag{26}$$

$$\boxed{(2CFL+1) * u_{n+1,2} - CFL * u_{n+1,3} = u_{n,2} + q_{n,2}\Delta t + CFL * f_{BC1}} \tag{27}$$

2) Neumann Condition at the Right Boundary x = L:

$$\frac{\partial u}{\partial x}(x = L, t) = C_2 \tag{28}$$

Forward Difference

$$\frac{u_{n+1,i+1} - u_{n+1,i}}{\Delta x} = C_2 \tag{29}$$

$$u_{n+1,i+1} = C_2 \Delta x + u_{n+1,i} \tag{30}$$

Where $i + 1 = m$ and m represents the last node (ghost node)

$$u_{n+1,m} = C_2 \Delta x + u_{n+1,m-1} \tag{31}$$

Plugging in

$$-CFL * u_{n+1,m-2} + (1 + 2CFL) * u_{n+1,m-1} - CFL * u_{n+1,m} = u_{n,m-1} + q_{n,m-1} \Delta t \tag{32}$$

$$-CFL*u_{n+1,m-2}+(1+2CFL)*u_{n+1,m-1}-CFL*\big(C_2\Delta x+u_{n+1,m-1}\big) = u_{n,m-1}+q_{n,m-1}\Delta t \tag{33}$$

$$-CFL * u_{n+1,m-2} + (1 + CFL) * u_{n+1,m-1} - CFL * C_2 \Delta x = u_{n,m-1} + q_{n,m-1} \Delta t \tag{34}$$

$$\boxed{-CFL * u_{n+1,m-2} + (1 + CFL) * u_{n+1,m-1} = u_{n,m-1} + q_{n,m-1} \Delta t + CFL * C_2 \Delta x} \tag{35}$$

Central Difference

$$\frac{u_{n+1,i+1} - u_{n+1,i-1}}{2\Delta x} = C_2 \tag{36}$$

$$u_{n+1,i+1} = 2C_2 \Delta x + u_{n+1,i-1} \tag{37}$$

Where $i + 1 = m$ and m represents the last node (ghost node)

$$u_{n+1,m} = 2C_2 \Delta x + u_{n+1,m-2} \tag{38}$$

Plugging in

$$-CFL*u_{n+1,m-2}+(1+2CFL)*u_{n+1,m-1}-CFL*(2C_2\Delta x+u_{n+1,m-2}) = u_{n,m-1}+q_{n,m-1}\Delta t \tag{39}$$

$$\boxed{-2CFL * u_{n+1,m-2} + (1 + 2CFL) * u_{n+1,m-1} = u_{n,m-1} + q_{n,m-1} \Delta t + 2CFL * C_2 \Delta x} \tag{40}$$

Crank-Nicolson
Let n denote the time step, and i denote the grid point

$$u_{n+1,i} = u_{n,i} + \Delta t * \Big[\frac{1}{2}f(t_n, u_n) + \frac{1}{2}f(t_{n+1}, u_{n+1}) + q_n\Big]_i \tag{41}$$

$$u_{n+1,i} = u_{n,i} + \Delta t * \Big[\frac{1}{2}\big(K\frac{\partial^2 u_n}{\partial x^2}\big) + \frac{1}{2}\big(K\frac{\partial^2 u_{n+1}}{\partial x^2}\big) + q_n\Big]_i \tag{42}$$

$$\begin{aligned} u_{n+1,i} = u_{n,i} + \Delta t * \Big[\frac{1}{2}\big(K\frac{u_n(x+\Delta x) - 2u_n(x) + u_n(x-\Delta x)}{\Delta x^2}\big) \\ \frac{1}{2}\big(K\frac{u_{n+1}(x+\Delta x) - 2u_{n+1}(x) + u_{n+1}(x-\Delta x)}{\Delta x^2}\big) + q_n\Big]_i \end{aligned} \tag{43}$$

Where

$$\begin{aligned} u_n(x+\Delta x) &= u_{n,i+1} \\ u_n(x) &= u_{n,i} \\ u_n(x-\Delta x) &= u_{n,i-1} \end{aligned} \tag{44}$$

And

$$\begin{aligned} u_{n+1}(x+\Delta x) &= u_{n+1,i+1} \\ u_{n+1}(x) &= u_{n+1,i} \\ u_{n+1}(x-\Delta x) &= u_{n+1,i-1} \end{aligned} \tag{45}$$

$$\begin{aligned} u_{n+1,i} = u_{n,i} + \Delta t * \Big[\frac{1}{2}\big(K\frac{u_{n,i+1} - 2u_{n,i} + u_{n,i-1}}{\Delta x^2}\big) \\ \frac{1}{2}\big(K\frac{u_{n+1,i+1} - 2u_{n+1,i} + u_{n+1,i-1}}{\Delta x^2}\big) + q_{n,i}\Big] \end{aligned} \tag{46}$$

We define the CFL Number to be:

$$CFL = K\frac{\Delta t}{(\Delta x)^2} \tag{47}$$

$$\boxed{\begin{aligned} \frac{-CFL}{2} * u_{n+1,i-1} + (1 + CFL) * u_{n+1,i} - \frac{-CFL}{2} * u_{n+1,i+1} = \qquad (48) \\ \frac{CFL}{2} * u_{n,i-1} + (1 - CFL) * u_{n,i} + \frac{CFL}{2} * u_{n,i+1} + q_{n,1}\Delta t \qquad (49) \end{aligned}}$$

1) Dirilecht Condition at the Left Boundary x = 0:

$$u(x=0,t) = u_{n+1,1} = \begin{cases} C_1 * \frac{t}{t_{ramp}} & t \leq t_{ramp} \\ C_1 & t > t_{ramp} \end{cases} = f_{BC1} \tag{50}$$

$$\begin{gathered} \frac{-CFL}{2} * f_{BC1} + (1+CFL) * u_{n+1,i} - \frac{-CFL}{2} * u_{n+1,i+1} = \\ \frac{CFL}{2} * u_{n,i-1} + (1-CFL) * u_{n,i} + \frac{CFL}{2} * u_{n,i+1} + q_{n,1}\Delta t \end{gathered} \tag{51}$$

$$\boxed{\begin{aligned} (1+CFL) * u_{n+1,i} - \frac{CFL}{2} * u_{n+1,i+1} = \quad (52) \\ \frac{CFL}{2} * u_{n,i-1} + (1-CFL) * u_{n,i} + \frac{CFL}{2} * u_{n,i+1} + q_{n,1}\Delta t + \frac{CFL}{2} * f_{BC1} \quad (53) \end{aligned}}$$

2) Neumann Condition at the Right Boundary x = L:

$$\frac{\partial u}{\partial x}(x=L,t) = C_2 \tag{54}$$

Forward Difference

$$\frac{u_{n+1,i+1} - u_{n+1,i}}{\Delta x} = C_2 \tag{55}$$

$$u_{n+1,i+1} = C_2\Delta x + u_{n+1,i} \tag{56}$$

Where $i+1 = m$ and m represents the last node (ghost node)

$$u_{n+1,m} = C_2\Delta x + u_{n+1,m-1} \tag{57}$$

Plugging in

$$\begin{gathered} \frac{-CFL}{2} * u_{n+1,i-1} + (1+CFL) * u_{n+1,i} - \frac{CFL}{2} * \big(C_2\Delta x + u_{n+1,i}\big) = \\ \frac{CFL}{2} * u_{n,i-1} + (1-CFL) * u_{n,i} + \frac{CFL}{2} * u_{n,i+1} + q_{n,1}\Delta t \end{gathered} \tag{58}$$

$$\boxed{\begin{aligned} \frac{-CFL}{2} * u_{n+1,i-1} + (1+\frac{CFL}{2}) * u_{n+1,i} = \quad (59) \\ \frac{CFL}{2} * u_{n,i-1} + (1-CFL) * u_{n,i} + \frac{CFL}{2} * u_{n,i+1} + q_{n,1}\Delta t + \frac{CFL}{2} * \big(C_2\Delta x\big) \quad (60) \end{aligned}}$$

Central Difference

$$\frac{u_{n+1,i+1} - u_{n+1,i-1}}{2\Delta x} = C_2 \tag{61}$$

$$u_{n+1,i+1} = 2C_2\Delta x + u_{n+1,i-1} \tag{62}$$

Where $i + 1 = m$ and m represents the last node (ghost node)

$$u_{n+1,m} = 2C_2\Delta x + u_{n+1,m-2} \tag{63}$$

Plugging in

$$\begin{split}\frac{-CFL}{2} * u_{n+1,i-1} + (1 + CFL) * u_{n+1,i} - \frac{CFL}{2} * \big(2C_2\Delta x + u_{n+1,i-1}\big) = \\ \frac{CFL}{2} * u_{n,i-1} + (1 - CFL) * u_{n,i} + \frac{CFL}{2} * u_{n,i+1} + q_{n,1}\Delta t\end{split} \tag{64}$$

$$\boxed{\begin{aligned} -CFL * u_{n+1,i-1} + (1 + CFL) * u_{n+1,i} = \quad (65) \\ \frac{CFL}{2} * u_{n,i-1} + (1 - CFL) * u_{n,i} + \frac{CFL}{2} * u_{n,i+1} + q_{n,1}\Delta t + \frac{CFL}{2} * \big(2C_2\Delta x\big) \quad (66)\end{aligned}}$$

2 Appendix

MATLAB Code for Numerical Solution of Heat Equation

Listing 1: Numerical Heat Equation

```matlab
%% 18.0851 Project
% Author      : Jered Dominguez-Trujillo
% Date        : May 2, 2019
% Description : Numerical Solution to Heat Equation

% SCHEME = 0 -> EXPLICIT
% SCHEME = 1 -> IMPLICIT
% SCHEME = 2 -> CRANK_NICOLSON

function NumHT(SCHEME, BC1, BC2, KT, L, NX, TM, NT, TR, SOURCE_FLAG, W, SIGMA, QMAX)
    close all; clc;

    % Boundary Conditions
    LENGTH = L;                         % Length of Domain
    TIME_RAMP = TR;                     % Time Ramp at X = 0 for BC to Go from 0 to C1
    C1 = BC1;                           % Dirilecht Condition U(x) = C1 at X = 0
    C2 = BC2;                           % Neumann Condition at dU/dX = C2 at X = L
    K = KT;                             % Thermal Diffusivity

    % Default Arguments
    if nargin <= 12
        QMAX = 1;                       % Default Maximum Value of Heat Source Function

        if nargin <= 11
            SIGMA = 0.2 * LENGTH;       % Default Standard Deviation of Heat Source Function
        end

        if nargin <= 10
            W = 1;                      % Default Frequency of Heat Source Function (rad/s)
        end
    end

    % Spatial Domain
    NODES = NX;                         % Nodes
    DX = LENGTH / NODES;                % DX Calculation
    X = linspace(0, LENGTH + DX, NODES + 2);% X Vector with Ghost Node

    % Time Domain
    TMAX = TM;                          % End Time of Simulation
    DT = TMAX / NT;                     % DT Calculation
    TIMESTEPS = TMAX / DT + 1;          % Number of Time Steps
    T = linspace(0, TMAX, TIMESTEPS);   % Time Vector

    % Calculate CFL Number
    CFL = (DT .* K) ./ (DX .* DX);      % Multiplication Factor K (DT / DX^2)

    % Print Out Simulation Info
    if SCHEME == 0
        fprintf('Explicit Method:\n');
    elseif SCHEME == 1
        fprintf('Implicit Method:\n');
    elseif SCHEME == 2
        fprintf('Crank-Nicolson Method:\n');
    end

    fprintf('BCs:\n(1) U(x) = %.2f At X = 0\n(2) dU/dX = %.2f at X = L = %.2f\n\n', C1, C2, L);
```

```
fprintf('Length: %.2f\t\tDX: %.2f\t\tNodes: %.0f\n', LENGTH, DX, NODES);
fprintf('Max Time: %.2f\t\tDT: %.2f\t\t\n\n', TMAX, DT);
if SOURCE_FLAG == 1
    fprintf('Q(x) = - QMAX * SIN(OMEGA * T) * EXP(-(X - L/2)^2 / SIGMA ^ 2)\n');
    fprintf('QMAX: %.2f\t\tOMEGA: %.2f rad/s\t\tSIGMA: %.2f\t\t L: %.2f\n\n', QMAX, W, SIGMA,
         LENGTH);
end
fprintf('CFL Number: %.2f\n', CFL);

% Initialize Matrices
U = zeros(TIMESTEPS, NODES + 2);
Q = zeros(TIMESTEPS, NODES + 2);

% If Heat Source Flag is True
if SOURCE_FLAG == 1
    for ii = 1:TIMESTEPS
        Q(ii, :) = - QMAX .* (sin((W .* T(ii)))) .* exp(-((X - (LENGTH ./ 2)) .^ 2) ./ (SIGMA .^
             2));
    end
end

% Initialize
CURRENT_T = 0;
TIMESTEP = 0;

% Plot Initial Conditions
fTemperature = figure('Name', 'Temperature History', 'NumberTitle', 'off');
figure(fTemperature);

% Temperature Profile
subplot(211);
plot(X, U(1, :), 'o-');
xlabel('X', 'FontSize', 18); ylabel('T', 'Fontsize', 18);
axis([0 LENGTH -2 2]);

% Heat Source Profile
subplot(212);
plot(X, Q(1, :), 'o-');
xlabel('X', 'FontSize', 18); ylabel('Q', 'Fontsize', 18);
axis([0 LENGTH -QMAX QMAX]);

suptitle(['Time = ', num2str(CURRENT_T), ' s']);
pause();

% Explicit Scheme
if SCHEME == 0

    % Iterate Until TMAX
    while CURRENT_T < TMAX
        CURRENT_T = CURRENT_T + DT;
        TIMESTEP = TIMESTEP + 1;

        n = TIMESTEP;

        % Explicit Euler Method
        for ii = 2:NODES + 1
            U(n + 1, ii) = CFL .* U(n, ii + 1) + (1 - 2 .* CFL) .* U(n, ii) + CFL .* U(n, ii -
                 1) + Q(n, ii) .* DT;
        end

        % Boundary Conditions
```

```
        U(n + 1, 1) = C1 * (CURRENT_T / TIME_RAMP) * (CURRENT_T <= TIME_RAMP) + C1 * (CURRENT_T
             > TIME_RAMP);
        U(n + 1, NODES + 2) = U(n + 1, NODES) + 2 * DX * C2;

        % Plot New Temperature Profile
        figure(fTemperature);

        % Temperature Profile
        subplot(211);
        plot(X, U(n + 1, :), 'o-');
        xlabel('X', 'FontSize', 18); ylabel('T', 'Fontsize', 18);
        axis([0 L -2 2]);

        % Heat Source Profile
        subplot(212);
        plot(X, Q(n + 1, :), 'o-');
        xlabel('X', 'FontSize', 18); ylabel('Q', 'Fontsize', 18);
        axis([0 LENGTH -QMAX QMAX]);

        suptitle(['Time = ', num2str(CURRENT_T), ' s']);

        pause(0.01);
    end

% Implicit Scheme
elseif SCHEME == 1
    LOWER = zeros(1, NODES);
    DIAG  = zeros(1, NODES + 1);
    UPPER = zeros(1, NODES);

    LOWERI = 2:NODES + 1; LOWERJ = 1:NODES;
    DIAGI = 1:NODES + 1; DIAGJ = 1:NODES + 1;
    UPPERI = 1:NODES; UPPERJ = 2:NODES + 1;

    DIAG(1) = 1; UPPER(1) = 0;

    for ii = 2:NODES
        LOWER(ii) = -CFL;
        DIAG(ii) = 2 * CFL + 1;
        UPPER(ii) = -CFL;
    end

    DIAG(NODES + 1) = 2 * CFL + 1;

    % Iterate Until TMAX
    while CURRENT_T < TMAX
        CURRENT_T = CURRENT_T + DT;
        TIMESTEP = TIMESTEP + 1;

        n = TIMESTEP;

        RHS = zeros(NODES + 1, 1);

        % Implicit Euler Method
        for ii = 2:NODES + 1
            RHS(ii) = U(n, ii) + Q(n, ii) .* DT;
        end

        % Boundary Conditions
        fBC1 = C1 * (CURRENT_T / TIME_RAMP) * (CURRENT_T <= TIME_RAMP) + C1 * (CURRENT_T >
             TIME_RAMP);
        LOWER(1) = 0;
```

```
        RHS(1) = fBC1;
        RHS(2) = RHS(2) + CFL .* fBC1;

        % Forward Difference Scheme
        % DIAG(NODES + 1) = CFL + 1;
        % RHS(NODES + 1) = RHS(NODES + 1) + C2 .* CFL .* DX;

        % Central Difference Scheme
        L(NODES) = -2 * CFL;
        RHS(NODES + 1) = RHS(NODES + 1) + 2 .* C2 .* CFL .* DX;

        MA = sparse([LOWERI, DIAGI, UPPERI], [LOWERJ, DIAGJ, UPPERJ], [LOWER, DIAG, UPPER],
             NODES + 1, NODES + 1);

        U(n + 1, 1:end-1) = MA \ RHS;

        % Forward Difference Scheme
        % U(n + 1, end) = U(n + 1, end - 1) + C2 .* DX;

        % Central Difference Scheme
        U(n + 1, end) = U(n + 1, end - 2) + 2 .* C2 .* DX;

        % Plot New Temperature Profile
        figure(fTemperature);

        % Temperature Profile
        subplot(211);
        plot(X, U(n + 1, :), 'o-');
        xlabel('X', 'FontSize', 18); ylabel('T', 'Fontsize', 18);
        axis([0 LENGTH -0.5 1.5]);

        % Heat Source Profile
        subplot(212);
        plot(X, Q(n + 1, :), 'o-');
        xlabel('X', 'FontSize', 18); ylabel('Q', 'Fontsize', 18);
        axis([0 LENGTH -QMAX QMAX]);

        suptitle(['Time = ', num2str(CURRENT_T), ' s']);

        pause(0.01);
    end

% Crank-Nicolson Scheme
elseif SCHEME == 2
    LOWER = zeros(1, NODES);
    DIAG  = zeros(1, NODES + 1);
    UPPER = zeros(1, NODES);

    LOWERI = 2:NODES + 1; LOWERJ = 1:NODES;
    DIAGI = 1:NODES + 1; DIAGJ = 1:NODES + 1;
    UPPERI = 1:NODES; UPPERJ = 2:NODES + 1;

    DIAG(1) = 1; UPPER(1) = 0;

    for ii = 2:NODES
        LOWER(ii) = -CFL ./ 2;
        DIAG(ii) = CFL + 1;
        UPPER(ii) = -CFL ./ 2;
    end

    DIAG(NODES + 1) = CFL + 1;
```

```
        % Iterate Until TMAX
        while CURRENT_T < TMAX
            CURRENT_T = CURRENT_T + DT;
            TIMESTEP = TIMESTEP + 1;

            n = TIMESTEP;

            RHS = zeros(NODES + 1, 1);

            % Crank-Nicolson Method
            for ii = 2:NODES + 1
                RHS(ii) = CFL .* U(n, ii - 1) ./ 2 + (1 - CFL) .* U(n, ii) + CFL .* U(n, ii + 1) ./
                     2 + Q(n, ii) .* DT;
            end

            % Boundary Conditions
            fBC1 = C1 * (CURRENT_T / TIME_RAMP) * (CURRENT_T <= TIME_RAMP) + C1 * (CURRENT_T >
                 TIME_RAMP);
            LOWER(1) = 0;
            RHS(1) = fBC1;
            RHS(2) = RHS(2) + CFL .* fBC1 ./ 2;

            % Forward Difference Scheme
            % DIAG(NODES + 1) = CFL ./ 2 + 1;
            % RHS(NODES + 1) = RHS(NODES + 1) + C2 .* CFL .* DX ./ 2;

            % Central Difference Scheme
            L(NODES) = -CFL;
            RHS(NODES + 1) = RHS(NODES + 1) + C2 .* CFL .* DX;

            MA = sparse([LOWERI, DIAGI, UPPERI], [LOWERJ, DIAGJ, UPPERJ], [LOWER, DIAG, UPPER],
                 NODES + 1, NODES + 1);

            U(n + 1, 1:end-1) = MA \ RHS;

            % Forward Difference Scheme
            % U(n + 1, end) = U(n + 1, end - 1) + C2 .* DX;

            % Central Difference Scheme
            U(n + 1, end) = U(n + 1, end - 2) + 2 .* C2 .* DX;

            % Plot New Temperature Profile
            figure(fTemperature);

            % Temperature Profile
            subplot(211);
            plot(X, U(n + 1, :), 'o-');
            xlabel('X', 'FontSize', 18); ylabel('T', 'Fontsize', 18);
            axis([0 LENGTH -0.5 1.5]);

            % Heat Source Profile
            subplot(212);
            plot(X, Q(n + 1, :), 'o-');
            xlabel('X', 'FontSize', 18); ylabel('Q', 'Fontsize', 18);
            axis([0 LENGTH -QMAX QMAX]);

            suptitle(['Time = ', num2str(CURRENT_T), ' s']);

            pause(0.01);
        end
    end
end
```