World Journal of Advanced Engineering Technology and Sciences
eISSN: 2582-8266
Cross Ref DOI: 10.30574/wjaets
Journal homepage: https://wjaets.com/

(REVIEW ARTICLE)

Numerical solution of the heat equation

Stanley A. Omenai *

Department of Mechanical Engineering, Georgia Institute of Technology, Atlanta, Georgia, USA.

World Journal of Advanced Engineering Technology and Sciences, 2024, 12(02), 468–478

Publication history: Received on 15 June 2024; revised on 27 July 2024; accepted on 30 July 2024

Article DOI: https://doi.org/10.30574/wjaets.2024.12.2.0322

Abstract

Numerical solution of the heat equation using finite difference approximation was developed to determine the temperature distribution profile for cooling of a rectangular steel bar.

Temperature distribution plots were made at characteristic time steps describing the heat distribution along the steel bar and ultimately depicting the time it takes for the steel bar to achieve isothermal equilibrium with the cooling medium.

Keywords: Finite Difference; Time Step; Diffusion Number; Stability; Node

1. Introduction

1.1. Problem description

In the steel industry, it is typical to rapidly quench steel bars coming directly from a blast furnace in a reservoir of cold water.

Consider a very long rectangular steel bar initially heated to a temperature (T_0 = 1000 K) from the furnace and placed in a reservoir of cold water (T_∞ = 300 K).

Assume the following properties:

Length of slab, L = 10 cm

Width of slab, W = 10 cm

Convective Heat Transfer coefficient of Water, h = 100 W/m^2-K

Thermal conductivity of Steel, k = 50 W/m-K

Density of Steel, ρ = 8000 kg/m^3

Specific Heat Capacity of Steel, c_p = 500 J/kg-K

Our task is to determine the temperature distribution profile for cooling of the steel bar by solving the heat equation using numerical techniques.

* Corresponding author: Stanley A. Omenai

2. Methodology

This problem shall be solved numerically using finite difference approximation. The finite difference method shall be explicit and forward-time, centred-space (FTCS).

MATLAB codes shall be developed to depict the temperature distribution in the slab at characteristic time steps.

The theory, assumptions, governing equations, and boundary conditions shall be provided with the solution steps.

A summary of the results (plots) shall be presented as part of the report.

3. Numerical solution

3.1. Assumptions

The steel bar is treated as very long such that there are no variations in the longitudinal axis (z), hence it is treated as a 2D slab (x,y)

The thermal conductivity of the material does not vary with space or time, i.e. material is isotropic

3.2. Methodology

- Discretize the domain into an m x n grid for analysis
- Consider a number of time steps p
- Use the forward-time and centred-space Finite Difference approach to obtain equations to describe the temperature profile of the slab at every location (x,y) and time t
- Vary grid spacings and time steps to obtain a stable solution using the stability criteria
- Plot the temperature profile of the ingot at five characteristic time steps

3.3. Solution

Governing Equation:

$$\frac{d^2T}{dx^2} + \frac{d^2T}{dy^2} = \frac{1}{\alpha}\frac{dT}{dt}$$

3.3.1. Boundary Conditions

- At x = 0, $\frac{dT}{dx} = -\frac{h}{k}(T - T_\infty)$
- At x = L, $\frac{dT}{dx} = -\frac{h}{k}(T - T_\infty)$
- At y = 0, $\frac{dT}{dy} = -\frac{h}{k}(T - T_\infty)$
- At y = W, $\frac{dT}{dy} = -\frac{h}{k}(T - T_\infty)$

3.3.2. Initial Condition:

$$\text{At } t = 0,\ T = T_o$$

To obtain T, we need to solve the governing equation using Finite Difference method in the following steps:

- Replace the partial derivatives with finite difference approximations

- Replace the time derivative with first order forward difference approximations
- Replace the space derivatives with second order centred-difference approximations

The governing equation becomes;

$$\left(\frac{T_{i,j}^{p+1} - T_{i,j}^{p}}{\Delta t}\right) = \alpha * \left[\left(\frac{T_{i-1,j}^{p} - 2 * T_{i,j}^{p} + T_{i+1,j}^{p}}{\Delta x^2}\right) + \left(\frac{T_{i,j-1}^{p} - 2 * T_{i,j}^{p} + T_{i,j+1}^{p}}{\Delta y^2}\right)\right]$$

Where *i* represents the node location along the *x* direction, *j* represents the node location along the *y* direction and *p* represents the time step.

Considering equal grid spacing in the *x* and *y* directions, $\Delta x = \Delta y$, thus;

$$T_{i,j}^{p+1} = T_{i,j}^{p} + \left(\frac{\alpha * \Delta t}{\Delta x^2}\right) * \left[T_{i-1,j}^{p} + T_{i+1,j}^{p} - 4 * T_{i,j}^{p} + T_{i,j-1}^{p} + T_{i,j+1}^{p}\right]$$

Let d = $\left(\frac{\alpha * \Delta t}{\Delta x^2}\right)$; where d – diffusion number,

The finite difference approximation of the equation becomes;

$$T_{i,j}^{p+1} = T_{i,j}^{p} + \mathrm{d} * \left[T_{i-1,j}^{p} + T_{i+1,j}^{p} - 4 * T_{i,j}^{p} + T_{i,j-1}^{p} + T_{i,j+1}^{p}\right]$$

3.4. Notes

- The above method is an explicit method where temperatures $T_{i,j}$'s at future times (p+1) are directly obtained based on $T_{i,j}$'s at present times as shown in the final equation above
- Explicit methods are conditionally stable. The stability criteria is given as $d <= 0.25$ (for 2D problems)
- Time step (Δt) needs to be small for more accuracy
- The error is of the order, $O(\Delta t) + O(\Delta x^2) + O(\Delta y^2)$

Next, we express the convective boundary condition equations as finite difference approximations.

Consider the 2D slab

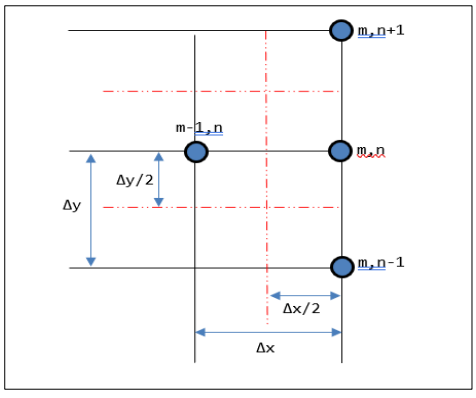

Figure 1 2D Slab for evaluation of Internal and Side Nodes

Applying energy balance;

$$\Delta y * \left(\frac{T_{m-1,n}^{p} - T_{m,n}^{p}}{\Delta x}\right) + \left(\frac{\Delta x}{2}\right) * \left(\frac{T_{m,n+1}^{p} - T_{m,n}^{p}}{\Delta y}\right) + \left(\frac{\Delta x}{2}\right) * \left(\frac{T_{m,n-1}^{p} - T_{m,n}^{p}}{\Delta y}\right) + h * \Delta y * (T_{\infty} - T_{m,n}^{p}) = \frac{1}{\propto} * \left(\frac{\Delta x}{2}\right) * \Delta y * \left(\frac{T_{m,n}^{p+1} - T_{m,n}^{p}}{\Delta t}\right)$$

Substituting $\Delta x = \Delta y$ and simplifying;

$$2 * \left(T_{m-1,n}^{p} - T_{m,n}^{p}\right) + \left(T_{m,n+1}^{p} - T_{m,n}^{p}\right) + \left(T_{m,n-1}^{p} - T_{m,n}^{p}\right) + \left(\frac{2 * h * \Delta x}{k}\right) * (\mathrm{T}_{\infty} - T_{m,n}^{p}) = \frac{1}{\propto}\left(\frac{\Delta x^2}{k}\right) * \left(\frac{T_{m,n}^{p+1} - T_{m,n}^{p}}{\Delta t}\right)$$

Simplifying further and substituting the following relations,

- Bi = $\left(\frac{h*\Delta x}{k}\right)$, where Bi = Biot number
- Fo = $\left(\frac{\alpha*\Delta t}{\Delta x^2}\right)$, where Fo = Fourier number

The final expression for the convective boundary condition at the side node becomes;

$$T_{m,n}^{p+1} = (Fo) * \left\{ (2 * Bi) * T_{inf+2} * T_{m-1,n}^{p} + T_{m,n+1}^{p} + T_{m,n-1}^{p} + \left[\left(\frac{1}{Fo}\right) - (2 * Bi) - 4\right] * T_{m,n}^{p} \right\}$$

For stability of side nodes,

$$scs = Fo*(2+Bi) < 0.5$$

For the external nodes, we also obtain by similar analogy as follows;

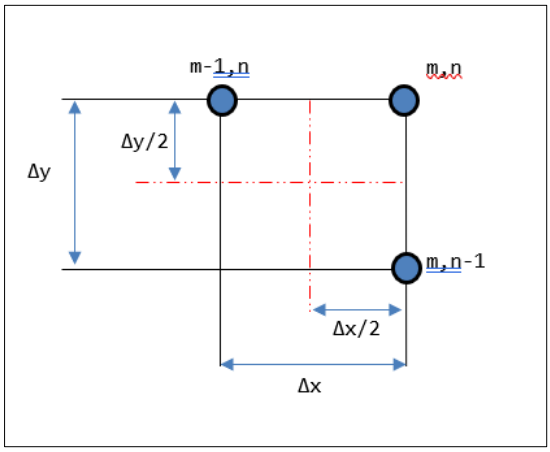

Figure 2 2D Slab for evaluation of External and Corner Nodes

Applying energy balance;

$$\left(\frac{\Delta y}{2}\right) * \left(\frac{T_{m-1,n}^{p} - T_{m,n}^{p}}{\Delta x}\right) + \left(\frac{\Delta x}{2}\right) * \left(\frac{T_{m,n-1}^{p} - T_{m,n}^{p}}{\Delta y}\right) + \frac{h}{k} * \left(\frac{\Delta y}{2}\right) * (T_{\infty} - T_{m,n}^{p}) + \frac{h}{k} * \left(\frac{\Delta x}{2}\right) * (T_{\infty} - T_{m,n}^{p}) = \frac{1}{\propto} * \left(\frac{\Delta x}{2}\right) * \left(\frac{\Delta y}{2}\right) * \left(\frac{T_{m,n}^{p+1} - T_{m,n}^{p}}{\Delta t}\right)$$

Substituting Δx = Δy and simplifying;

$$2*\left(T_{m-1,n}^{p} - T_{m,n}^{p}\right) + 2 * \left(T_{m,n-1}^{p} - T_{m,n}^{p}\right) + \left(\frac{4*h*\Delta x}{k}\right)*(T_{\infty} - T_{m,n}^{p}) = \left(\frac{\Delta x^2}{\alpha*\Delta t}\right)*\left(T_{m,n}^{p+1} - T_{m,n}^{p}\right)$$

Simplifying further and substituting as before, we obtain the final expression for the convective boundary conditions at the edge nodes as follows;

$$T_{m,n}^{p+1} = 2*(Fo) * \left\{T_{m-1,n}^{p} + T_{m,n-1}^{p} + (2 * Bi1 * T_{\infty}) + \left(-2 * Bi + \left(\frac{1}{2*Fo}\right) - 2\right) T_{m,n}^{p}\right\}$$

For stability of corner nodes,

$$scc = Fo*(1+Bi) < 0.25$$

Thus, we have a set of nine (9) equations; one (1) governing equation which characterizes the internal nodes and eight (8) equations corresponding to the four sides and four edges of the discretized 2D domain as follows:

Governing Equation

$$T_{i,j}^{p+1} = T_{i,j}^{p} + \mathrm{d} * \left[T_{i-1,j}^{p} + T_{i+1,j}^{p} - 4 * T_{i,j}^{p} + T_{i,j-1}^{p} + T_{i,j+1}^{p}\right]$$

Boundary Condition BC1 (nodes along the right side)

$$T_{m,n}^{p+1} = (Fo) * \left\{ (2 * Bi) * \mathrm{T}_{\infty} + 2 * T_{m-1,n}^{p} + T_{m,n+1}^{p} + T_{m,n-1}^{p} + \left[\left(\frac{1}{Fo}\right) - (2 * Bi) - 4\right] * T_{m,n}^{p} \right\}$$

Boundary Condition BC3 (nodes along the left side)

$$T_{m,n}^{p+1} = (Fo) * \left\{ (2 * Bi) * \mathrm{T}_{\infty} + 2 * T_{m+1,n}^{p} + T_{m,n+1}^{p} + T_{m,n-1}^{p} + \left[\left(\frac{1}{Fo}\right) - (2 * Bi) - 4\right] * T_{m,n}^{p} \right\}$$

Boundary Condition BC2 (nodes along the bottom side)

$$T_{m,n}^{p+1} = (Fo) * \left\{ (2 * Bi) * \mathrm{T}_{\infty} + 2 * T_{m,n-1}^{p} + T_{m+1,n}^{p} + T_{m-1,n}^{p} + \left[\left(\frac{1}{Fo}\right) - (2 * Bi) - 4\right] * T_{m,n}^{p} \right\}$$

Boundary Condition BC4 (nodes along the top side)

$$T_{m,n}^{p+1} = (Fo) * \left\{ (2 * Bi) * \mathrm{T}_{\infty} + 2 * T_{m,n+1}^{p} + T_{m+1,n}^{p} + T_{m-1,n}^{p} + \left[\left(\frac{1}{Fo}\right) - (2 * Bi) - 4\right] * T_{m,n}^{p} \right\}$$

Boundary Condition at the top right corner

$$T_{m,n}^{p+1} = 2 * (Fo) * \left\{ T_{m-1,n}^{p} + T_{m,n+1}^{p} + (2 * Bi * T_{\infty}) + \left(-2 * Bi + \left(\frac{1}{2 * Fo}\right) - 2\right) T_{m,n}^{p} \right\}$$

Boundary Condition at the bottom right corner

$$T_{m,n}^{p+1} = 2 * (Fo) * \left\{ T_{m-1,n}^{p} + T_{m,n-1}^{p} + (2 * Bi * T_{\infty}) + \left(-2 * Bi + \left(\frac{1}{2 * Fo}\right) - 2\right) T_{m,n}^{p} \right\}$$

Boundary Condition at the bottom left corner

$$T_{m,n}^{p+1} = 2 * (Fo) * \left\{ T_{m+1,n}^{p} + T_{m,n-1}^{p} + (2 * Bi * T_{\infty}) + \left(-2 * Bi + \left(\frac{1}{2 * Fo}\right) - 2\right) T_{m,n}^{p} \right\}$$

Boundary Condition at the top left corner

$$T_{m,n}^{p+1} = 2 * (Fo) * \left\{ T_{m+1,n}^{p} + T_{m,n+1}^{p} + (2 * Bi * T_{\infty}) + \left(-2 * Bi + \left(\frac{1}{2 * Fo}\right) - 2\right) T_{m,n}^{p} \right\}$$

Next, we select and vary grid spacings and time steps to obtain a stable solution considering the stability criteria;

- Total time, t = 10,000s
- Number of time steps, nt = 25,000
- Number of grid spacings, nx = 20
- Slab length L= 0.1m
- Slab Width W= 0.1m
- $\Delta t = t/nt$ = 0.4s
- $\Delta x = L/nx$ = 0.005m
- Number of interior points, r = 441
- Number of points in a row, m =21
- Number of points in a column, n =21
- Time steps, $p = (t/\Delta t) + 1$=25,001
- Initial Temperature, T_o = 1000 K
- Ambient Temperature, T_∞ = 300 K
- Density, ρ = 8000 kg/m3
- Specific Heat Capacity, c_p = 500 J/kg-K
- Convection coefficient, h = 100W/m^2-K

- Thermal conductivity, κ = 50W/m-K
- Thermal diffusivity, $\alpha = \frac{k}{\rho c_p} = 1.25 \times 10^{-5}$ m²/s
- Fourier number, $Fo = (\frac{\alpha \times \Delta t}{\Delta x^2}) = \left(\frac{1.25 \text{ x } 10^{-5} \text{ x } 0.36}{0.005^2}\right) = 0.2$
- Biot number, $Bi = \left(\frac{h \times \Delta x}{k}\right) = \left(\frac{100 \text{ x } 0.005}{50}\right) = 0.01$

3.5. Stability check

Diffusion number, $d = \alpha \times \Delta t/\Delta x^2 = 0.2 < 0.25$ [solution stable]

Stability of sides, $scs = Fo \times (2+Bi) = 0.2 \times (2+0.01) = 0.402 < 0.5$ [solution stable]

Stability of corners, $scc = Fo \times (1+Bi) = 0.2 \times (1+0.01) = 0.202 < 0.25$ [solution stable]

Finally, we develop a numerical code of the set of equations Using MATLAB to obtain an overall temperature profile for the slab at any time (t) and location (x, y) *(code is attached as Appendix)*.

4. Results and discussions

Using the numerical solution developed, we plot the temperature profile as a function of location (x, y) of the slab at the initial condition and at five different characteristic times (t = *0.4s, 60s, 360s, 900s & 10,000s*).

The results are presented in the following colormaps.

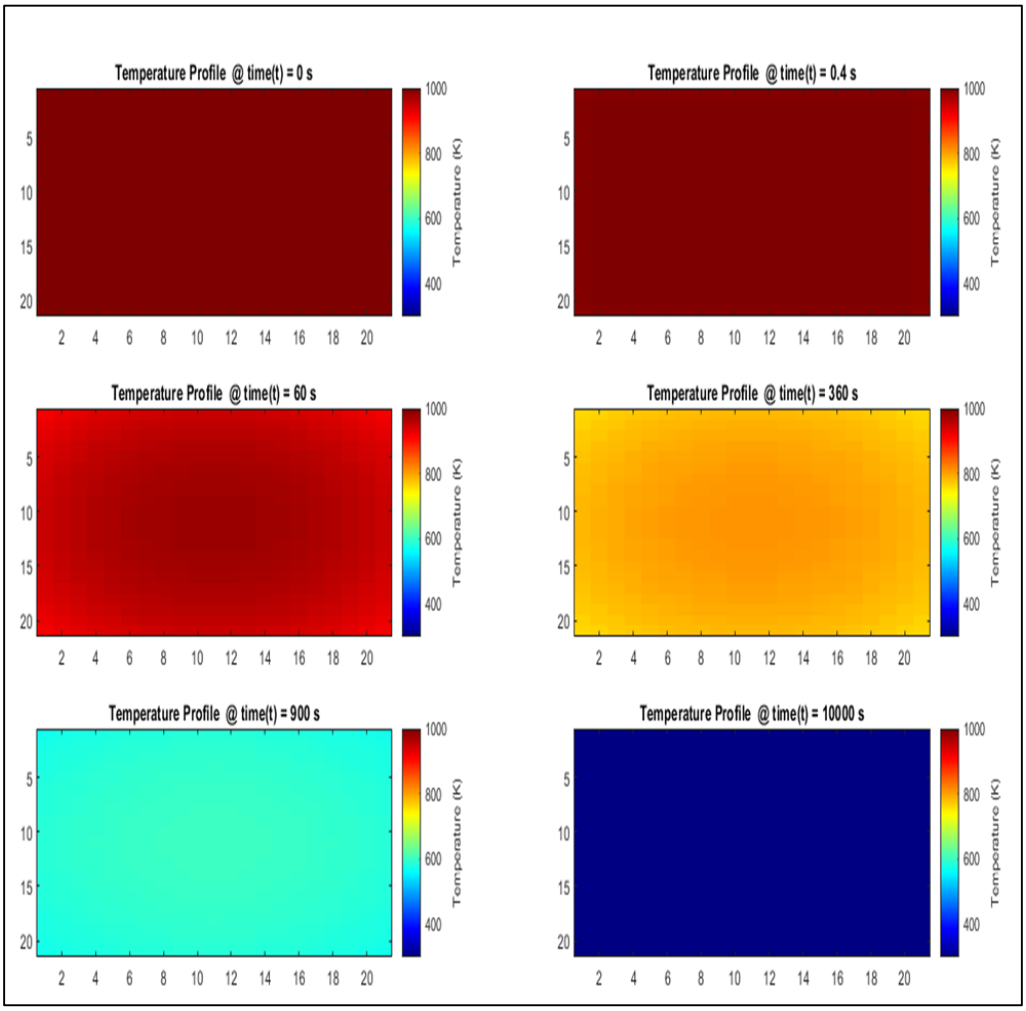

Figure 3 2D Temperature Plot at characteristic times (t = 0.4s, 60s, 360s, 900s & 10,000s)

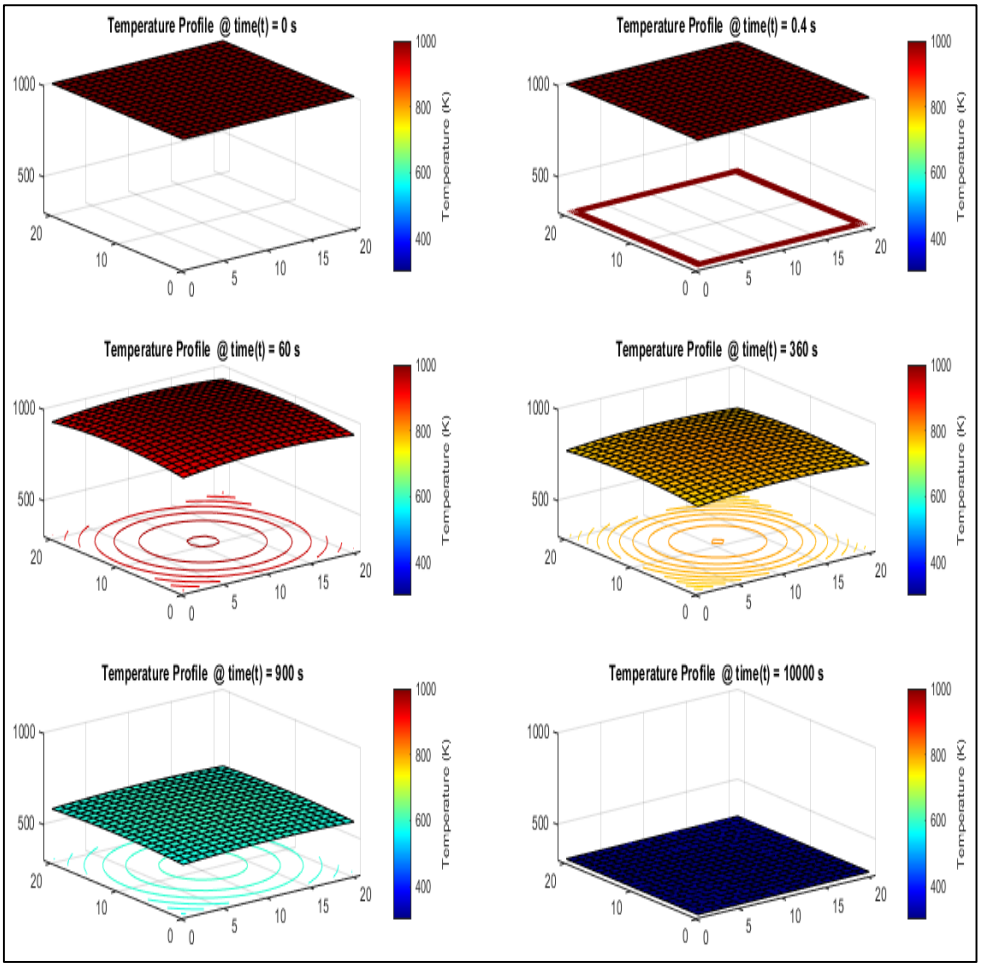

Figure 4 Temperature Profiles as a function of location (x, y) at characteristic times (t = 0.4s, 60s, 360s, 900s & 10,000s)

Table 1 Temperature Range at Different Characteristic Times

S/n	Time (s)	Maximum temperature (k)	Minimum temperature (k)
1	0	1000.0000	1000.0000
2	0.4	1000.0000	994.4000
3	60	980.7217	919.4122
4	360	810.1785	762.3811
5	900	602.5205	574.1781
6	10,000	300.0453	300.0410

Note: Maximum temperature occurs at the centre of the slab, while minimum temperature occurs at the corner edges

Notes

- The numerical solution gives an accurate description of the state of the slab at the initial condition (T = 1000K everywhere).
- The solution depicts Newton's cooling with the edges generally at lower temperatures than the centre of the slab.
- The solution shows isothermal condition of the slab at around t =10,000s.

- The magnitude of the error in the numerical solution is of the order, $O(\Delta t) + O(\Delta x^2) + O(\Delta y^2)$, i.e. $[0.4+0.005^2+0.005^2] = 0.40005$.
- This error magnitude can be reduced by increasing the number of time steps, *nt*, thus reducing Δt.
- Increasing the number of time steps significantly increases the time it takes for MATLAB to compute the solution. For the case of 100,000 time steps, it takes over one hour with slight improvement in accuracy of results.
- An unstable explicit numerical solution gives results that are not predictable, i.e. do not vary with time and space in a predictable manner, thus, explicit numerical solutions require stability check.

5. Conclusion

In this study, we have successfully developed a numerical solution for the heat equation using the explicit forward-time, centred-space (FTCS) finite difference method. The solution accurately captures the temperature distribution profile for cooling of a typical rectangular steel bar.

The numerical solution presented in this study offers a valuable tool for understanding and predicting the temperature distribution for steel structures during heating or cooling. Future work can extend this approach to more complex geometry and boundary conditions, further enhancing its applicability in engineering and scientific research.

References

[1] David L. Powers. Boundary Value Problems and Partial Differential Equations, 5th Edition. Elsevier Academic Press.

[2] David W. Hahn, M. Necati Özisik. Heat Conduction, 3rd Edition. John Wiley & Sons, Inc.

[3] Bergman, Theodore L.; Lavine, Adrienne S.; Incropera, Frank P.; Dewitt, David P. (2011). Fundamentals of heat and mass transfer (7th ed.). Hoboken, NJ: Wiley.

[4] https://www.youtube.com/watch?v=Ip47nsJOQqs. Solve 2D Transient Heat Conduction Problem with Convection Boundary Conditions using FTCS Finite Difference Method.

Appendix – MATLAB code

```
clc;
clear all;
close all;

% Inputs

L = 0.1; %m
W = 0.1; %m
T_0 = 1000; %K
T_inf = 300; %K
h = 100; %W/m^2-K
kappa = 50; %W/m-K
rho = 8000; %kg/m^3
C_p = 500; %J/kg-K
alpha = kappa/(rho*C_p); %m^2/s
t = 10000; % s
nt = 25000;
delta_t = t/nt; % s
nx = 20;
delta_x = L/nx; % m
delta_y = delta_x; % m
Bi = (h*delta_x)/kappa;
Fo = alpha*delta_t/delta_x^2;

% Solution

m = (L/delta_x) + 1; % no. of points in a row
n = (W/delta_x) + 1; % no. of points in a column
r = n*m; % no. of interior points
p = (t/delta_t) + 1; % no. of time steps
```

```
d = alpha*delta_t/delta_x^2; % diffusion number
scs=Fo*(2+Bi);
scc=Fo*(1+Bi);

% Stability Criteria

if d <= 0.25
  fprintf('\n')
  fprintf('solution stable at the interior nodes\nd = %8.4f', d)
else
  fprintf('\n')
  fprintf('solution unstable at the interior nodes\nd = %8.4f', d)
end
if scs <= 0.5
  fprintf('\n')
  fprintf('solution stable at the sides\nscs = %8.4f', scs)
else
  fprintf('\n')
  fprintf('solution unstable at the sides\nscs = %8.4f', scs)
end
if scc <= 0.25
  fprintf('\n')
  fprintf('solution stable at the corners\nscc = %8.4f', scc)
else
  fprintf('\n')
  fprintf('solution unstable at the corners\nscc = %8.4f', scc)
end

% Creating initial and boundary conditions

T = zeros(m,n,p);

% Creating initial conditions

for k = 1:1
  for j = 1:n
   for i = 1:m
     T(i,j,k) = T_0;
      end
  end
end
T;

% Creating boundary conditions

for k = 1:p-1
  for i = 2:m-1
   for j = 1:1
        T(i,j,k+1) = Fo*( 2*Bi*T_inf + 2*T(i,j+1,k) + T(i+1,j,k) + T(i-1,j,k) + ((1/Fo) - 2*Bi - 4)*T(i,j,k) );
   end
  end
    for i = m:m
   for j = 2:n-1
        T(i,j,k+1) = Fo*( 2*Bi*T_inf + 2*T(i-1,j,k) + T(i,j+1,k) + T(i,j-1,k) + ((1/Fo) - 2*Bi - 4)*T(i,j,k) );
   end
  end
    for i = 2:m-1
   for j = n:n
        T(i,j,k+1) = Fo*( 2*Bi*T_inf + 2*T(i,j-1,k) + T(i+1,j,k) + T(i-1,j,k) + ((1/Fo) - 2*Bi - 4)*T(i,j,k) );
   end
  end
    for i = 1:1
   for j = 2:n-1
        T(i,j,k+1) = Fo*( 2*Bi*T_inf + 2*T(i+1,j,k) + T(i,j+1,k) + T(i,j-1,k) + ( (1/Fo) - 2*Bi - 4) *T(i,j,k) );
   end
  end
        T(1,1,k+1) = 2*Fo*( T(2,1,k) + T(1,2,k) + 2*Bi*T_inf + (- 2*Bi + (1/(2*Fo)) - 2 )*T(1,1,k));
       T(m,1,k+1) = 2*Fo*( T(m-1,1,k) + T(m,2,k) + 2*Bi*T_inf + (- 2*Bi + (1/(2*Fo)) - 2 )*T(m,1,k));

       T(m,n,k+1) = 2*Fo*( T(m-1,n,k) + T(m,n-1,k) + 2*Bi*T_inf + (- 2*Bi + (1/(2*Fo)) - 2 )*T(m,n,k));

       T(1,n,k+1) = 2*Fo*( T(2,n,k) + T(1,n-1,k) + 2*Bi*T_inf + (- 2*Bi + (1/(2*Fo)) - 2 )*T(1,n,k));
```

```
 for i = 2:m-1
  for j = 2:n-1
      T(i,j,k+1) = T(i,j,k) + d*( T(i+1,j,k) + T(i-1,j,k) + T(i,j+1,k) + T(i,j-1,k) - 4*T(i,j,k) );
  end
 end
end
T(:,:,1);
T(:,:,2);
T(:,:,3);
T(:,:,p);
T;

% Temperature Profile

T11 = T;

%T11 = rot90(T11);

% Initial Temperature Profile

figure(1)
subplot(3,2,1)
T12 = imagesc(T11(:,:,1));
colorbar;
colormap(jet);
title(['Temperature Profile ', '@ time(t) = ', num2str(0), ' s'])
caxis([300 1000]);
set(get(colorbar,'label'),'string','Temperature (K)');

% Intermediate Temperature Profiles

subplot(3,2,2)
T16 = imagesc(T11(:,:,2));
colorbar;
colormap(jet);
title(['Temperature Profile ', '@ time(t) = ', num2str(0.4), ' s'])
caxis([300 1000]);
set(get(colorbar,'label'),'string','Temperature (K)');
subplot(3,2,3)
T13 = imagesc(T11(:,:,151));
colorbar;
colormap(jet);
title(['Temperature Profile ', '@ time(t) = ', num2str(60), ' s'])
caxis([300 1000]);
set(get(colorbar,'label'),'string','Temperature (K)');
subplot(3,2,4)
T14 = imagesc(T11(:,:,901));
colorbar;
colormap(jet);
title(['Temperature Profile ', '@ time(t) = ', num2str(360), ' s'])
caxis([300 1000]);
set(get(colorbar,'label'),'string','Temperature (K)');
subplot(3,2,5)
T15 = imagesc(T11(:,:,2251));
colorbar;
colormap(jet);
title(['Temperature Profile ', '@ time(t) = ', num2str(900), ' s'])
caxis([300 1000]);
set(get(colorbar,'label'),'string','Temperature (K)');

% Final Temperature Profile

subplot(3,2,6)
T17 = imagesc(T11(:,:,p));
colorbar;
colormap(jet);
title(['Temperature Profile ', '@ time(t) = ', num2str(t), ' s'])
caxis([300 1000]);
set(get(colorbar,'label'),'string','Temperature (K)');

% Initial Temperature Profile
```

```
figure(2)
subplot(3,2,1)
T12 = surfc(T11(:,:,1));
axis ([0 m 0 n 300 1000]);
colorbar;
colormap(jet);
title(['Temperature Profile ', '@ time(t) = ', num2str(0), ' s'])
caxis([300 1000]);
set(get(colorbar,'label'),'string','Temperature (K)');

% Intermediate Temperature Profiles

subplot(3,2,2)
T16 = surfc(T11(:,:,2));
axis ([0 m 0 n 300 1000]);
colorbar;
colormap(jet);
title(['Temperature Profile ', '@ time(t) = ', num2str(0.4), ' s'])
caxis([300 1000]);
set(get(colorbar,'label'),'string','Temperature (K)');
subplot(3,2,3)
T13 = surfc(T11(:,:,151));
axis ([0 m 0 n 300 1000]);
colorbar;
colormap(jet);
title(['Temperature Profile ', '@ time(t) = ', num2str(60), ' s'])
caxis([300 1000]);
set(get(colorbar,'label'),'string','Temperature (K)');
subplot(3,2,4)
T14 = surfc(T11(:,:,901));
axis ([0 m 0 n 300 1000]);
colorbar;
colormap(jet);
title(['Temperature Profile ', '@ time(t) = ', num2str(360), ' s'])
caxis([300 1000]);
set(get(colorbar,'label'),'string','Temperature (K)');
subplot(3,2,5)
T15 = surfc(T11(:,:,2251));
axis ([0 m 0 n 300 1000]);
colorbar;
colormap(jet);
title(['Temperature Profile ', '@ time(t) = ', num2str(900), ' s'])
caxis([300 1000]);
set(get(colorbar,'label'),'string','Temperature (K)');

% Final Temperature Profile

subplot(3,2,6)
T17 = surfc(T11(:,:,p));
axis ([0 m 0 n 300 1000]);
colorbar;
colormap(jet);
title(['Temperature Profile ', '@ time(t) = ', num2str(t), ' s'])
caxis([300 1000]);
set(get(colorbar,'label'),'string','Temperature (K)');
```