Table of Contents

Assignemnt

PDE's

```
% Name         : Mohamed Mafaz
% Roll Number : AM25M009
% Department   : Applied Mechanics

clc;
clear;
close all;
```

Part 1 (Preprocessing)

Parameters

```
width = 1.0;
height = 1.5;
k = 0.4;
Q_dot = 100;
nx = 5;
ny = 7;

max_iterations = 100;

% Grid and Coefficients
dx = width / (nx - 1);
dy = height / (ny - 1);
alpha = 1 / (dx^2);
beta = 1 / (dy^2);
source = -Q_dot / k;

omega = 0.1;

T = zeros(ny,nx);
tolerance = 1e-2;

tic;
```

Part 2 (Processing / Using the Algorithms)

```
omegas = [0.25, 0.5, 1];
for p = 1:length(omegas)
```

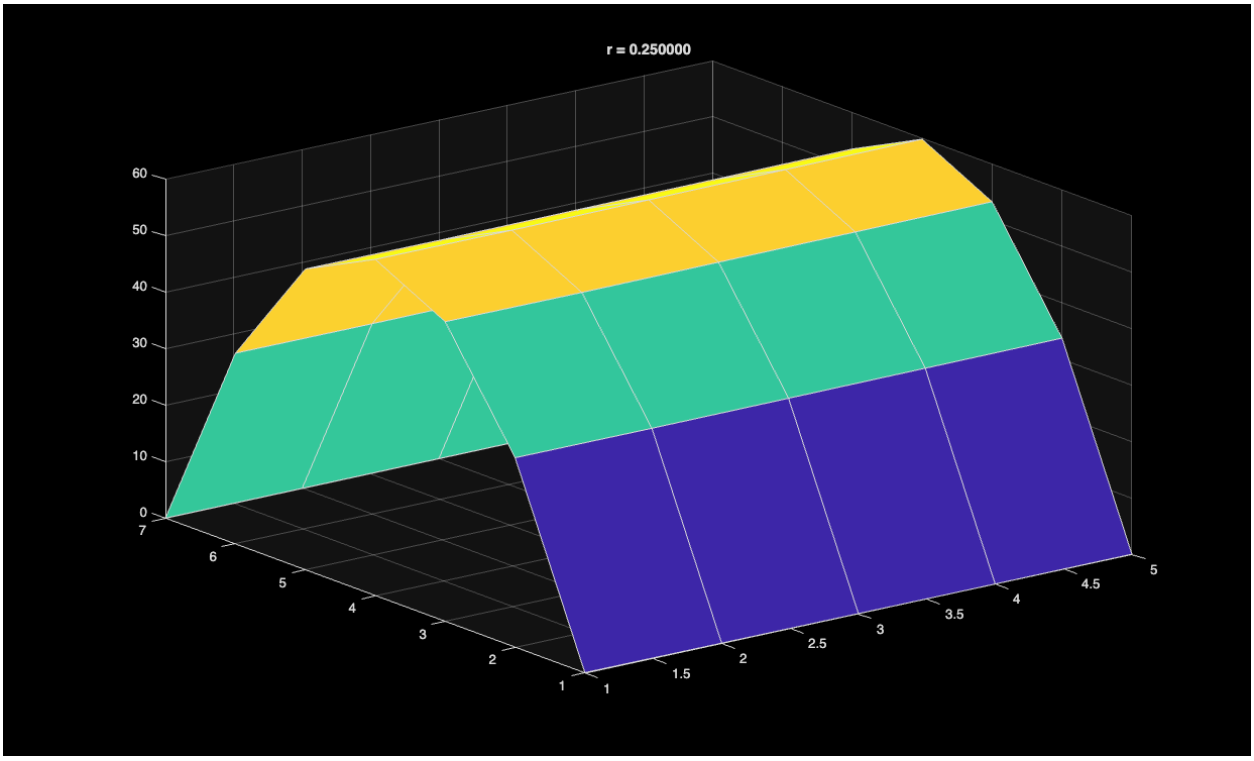
```
omega = omegas(p)
for iter = 1:max_iterations
    T_old = T;

    % Interior points
    for j = 2:ny-1
        for i = 2:nx-1
            T_new = (alpha * (T(j, i+1) + T(j, i-1)) + ...
                     beta  * (T(j+1, i) + T(j-1, i)) - ...
                     source) / (2*alpha + 2*beta);

            % relaxation
            T(j,i) = (1 - omega) * T(j,i) + omega * T_new;
        end
    end

    % Neumann boundaries
    for j = 2:ny-1
        % Left
        T_new_left = (2 * alpha * T(j, 2) + ...
                      beta * (T(j+1, 1) + T(j-1, 1)) - ...
                      source) / (2*alpha + 2*beta);
        T(j,1) = (1 - omega) * T(j,1) + omega * T_new_left;

        % Right
        T_new_right = (2 * alpha * T(j, nx-1) + ...
                       beta * (T(j+1, nx) + T(j-1, nx)) - ...
                       source) / (2*alpha + 2*beta);
        T(j,nx) = (1 - omega) * T(j,nx) + omega * T_new_right;
    end


    max_change = max(max(abs(T - T_old)));
    if max_change < tolerance
        fprintf('Converged after %d iterations\n', iter);
        break;
    end
end

elapsed_time = toc;
fprintf('Elapsed time: %f seconds\n for 5 points', elapsed_time);


disp('Temperature Distribution:');
disp(T);

figure;
surf(T)
title(sprintf("r = %f", omega))

x_coords = linspace(0, width, nx);
y_coords = linspace(0, height, ny);

[X, Y] = meshgrid(x_coords, y_coords);
```

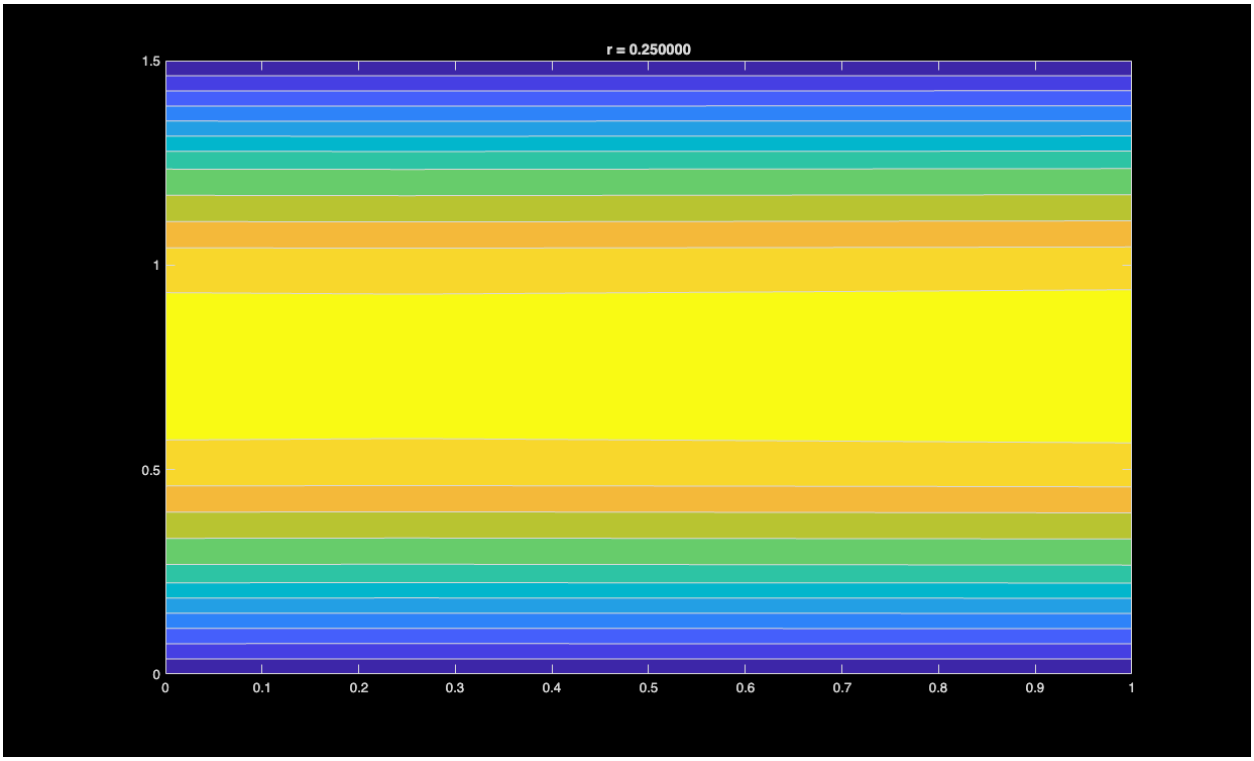
```
    figure;
    contourf(X, Y, T);
    title(sprintf("r = %f", omega))
    hold on;
    % [C, h] = contour(X, Y, T);
    % colorbar;
    % xlabel('x');
    % ylabel('y');
end


Width  = 1.0;
Height = 1.5;
alpha  = 1;
beta   = 1;
source = -source;
tolerance = 1e-6;
max_iterations = 10000;
omega = 0.1; % relaxation factor

T = zeros(ny, nx);

tic;
for iter = 1:max_iterations
    T_old = T;
    max_change = 0;

    % Interior points
    for j = 2:ny-1
        for i = 2:nx-1
            % 9-point
            T_new = ( ...
                4*(T(j, i+1) + T(j, i-1) + T(j+1, i) + T(j-1, i)) + ...
                (T(j+1, i+1) + T(j+1, i-1) + T(j-1, i+1) + T(j-1, i-1)) + ...
                6*source*dx^2 ) / 20;

            % relaxation
            T(j,i) = T(j,i) + omega * (T_new - T(j,i));

            % Track error
            err = abs(T(j,i) - T_old(j,i));
            if err > max_change
                max_change = err;
            end
        end
    end

    % Neumann boundaries
    for j = 2:ny-1
        % Left
        T_new_left = T(j,2);
        T(j,1) = (1 - omega) * T(j,1) + omega * T_new_left;

        % Right
```

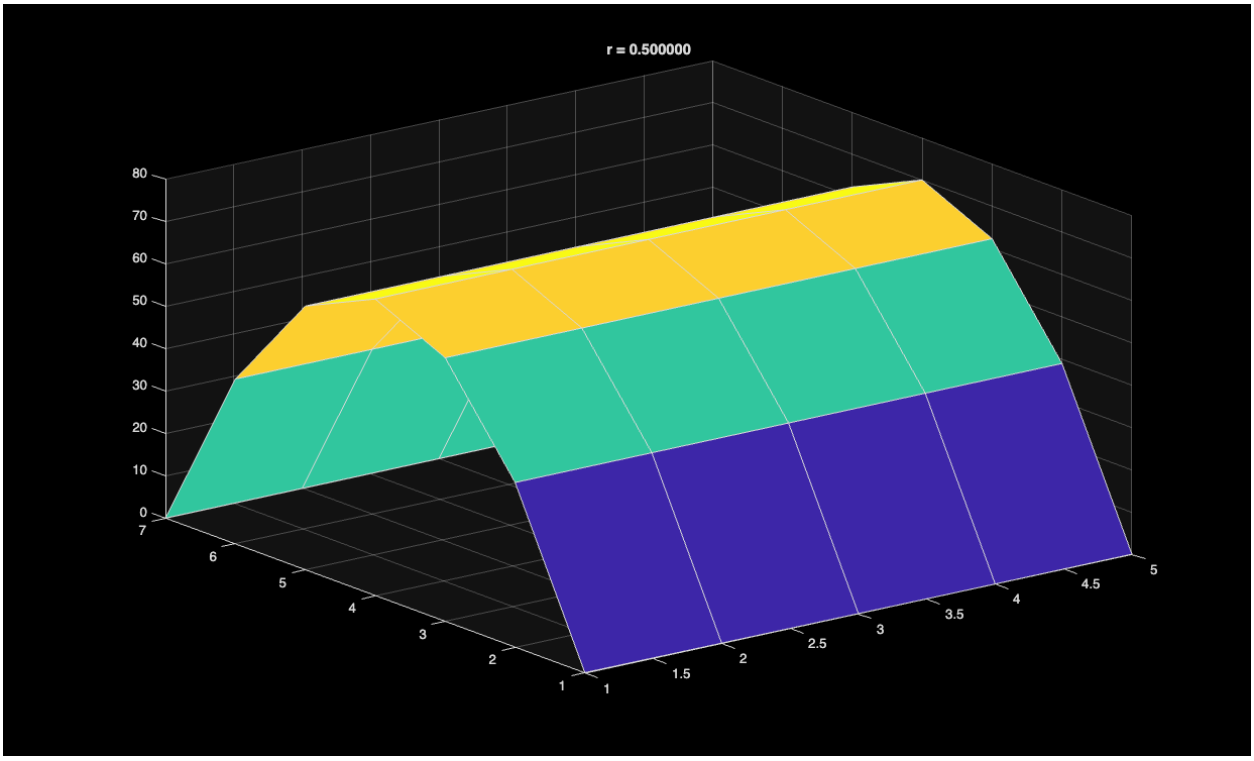
```
        T_new_right = T(j,nx-1);
        T(j,nx) = (1 - omega) * T(j,nx) + omega * T_new_right;
    end

    % Convergence check
    if max_change < tolerance
        break;
    end
end

elapsed_time = toc;
fprintf('Elapsed time: %f seconds\n for 9 points', elapsed_time);
T
```

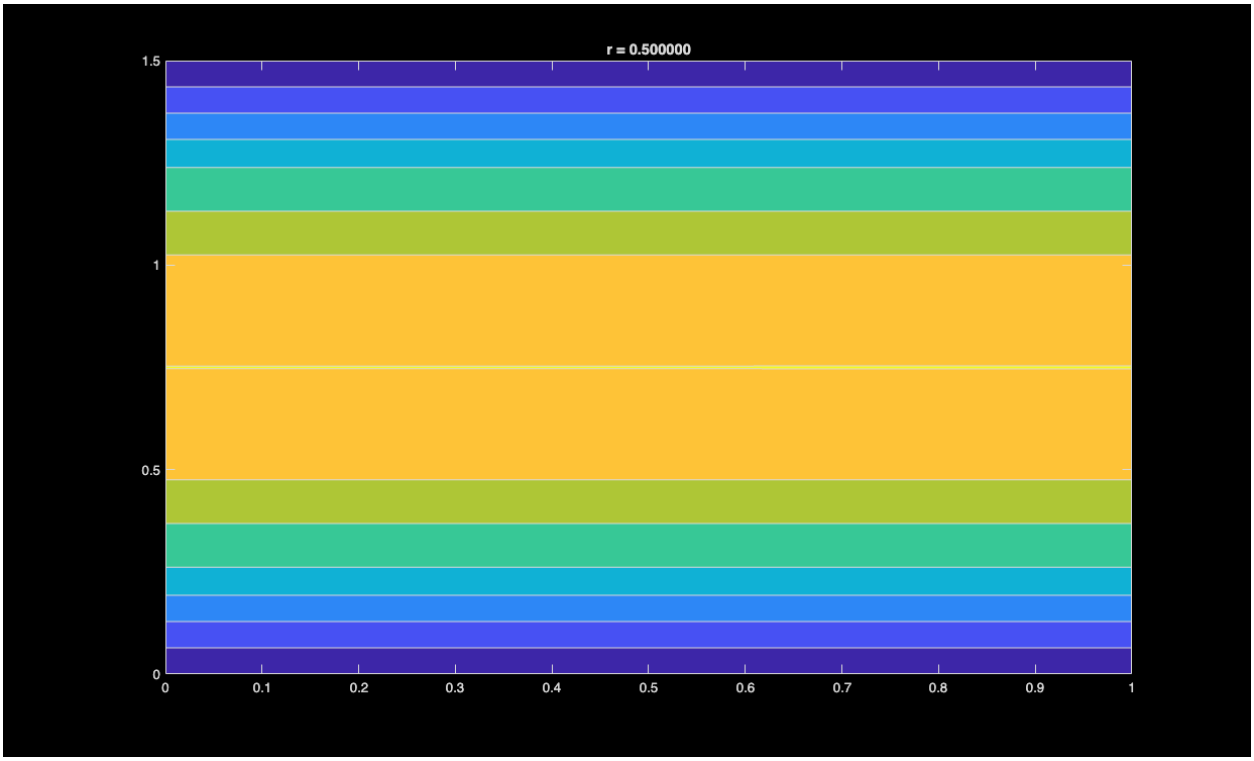
```
omega =

    0.2500

Elapsed time: 0.055747 seconds
 for 5 pointsTemperature Distribution:
         0         0         0         0         0
   33.6005   33.5477   33.5995   33.6509   33.7022
   53.1292   53.0386   53.1275   53.2156   53.3037
   59.5943   59.4907   59.5924   59.6933   59.7940
   53.3056   53.2166   53.3040   53.3905   53.4770
   33.8044   33.7536   33.8035   33.8530   33.9025
         0         0         0         0         0


omega =

    0.5000

Converged after 88 iterations
Elapsed time: 0.175215 seconds
 for 5 pointsTemperature Distribution:
         0         0         0         0         0
   38.9534   38.9509   38.9534   38.9558   38.9581
   62.3152   62.3110   62.3152   62.3193   62.3233
   70.1038   70.0991   70.1038   70.1084   70.1129
   62.3233   62.3193   62.3233   62.3272   62.3310
   38.9627   38.9605   38.9627   38.9649   38.9671
         0         0         0         0         0


omega =

     1

Converged after 9 iterations
Elapsed time: 0.222168 seconds
 for 5 pointsTemperature Distribution:
         0         0         0         0         0
```

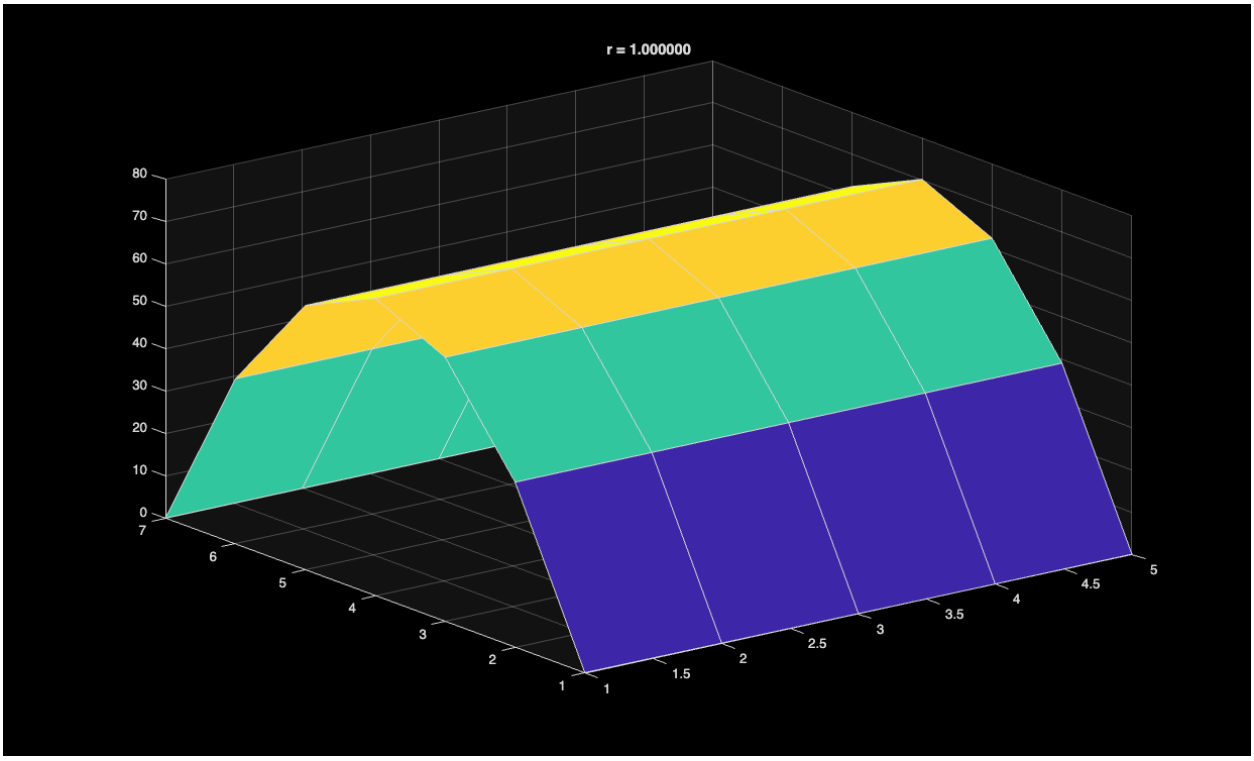
```
   39.0271   39.0245   39.0269   39.0292   39.0314
   62.4426   62.4385   62.4424   62.4461   62.4497
   70.2505   70.2460   70.2503   70.2544   70.2582
   62.4498   62.4462   62.4497   62.4530   62.4561
   39.0355   39.0335   39.0354   39.0372   39.0389
         0         0         0         0         0

Elapsed time: 0.025862 seconds
 for 9 points
T =

         0         0         0         0         0
   39.0624   39.0624   39.0624   39.0624   39.0624
   62.4999   62.4999   62.4999   62.4999   62.4999
   70.3123   70.3124   70.3124   70.3124   70.3123
   62.4999   62.4999   62.4999   62.4999   62.4999
   39.0624   39.0624   39.0624   39.0624   39.0624
         0         0         0         0         0
```

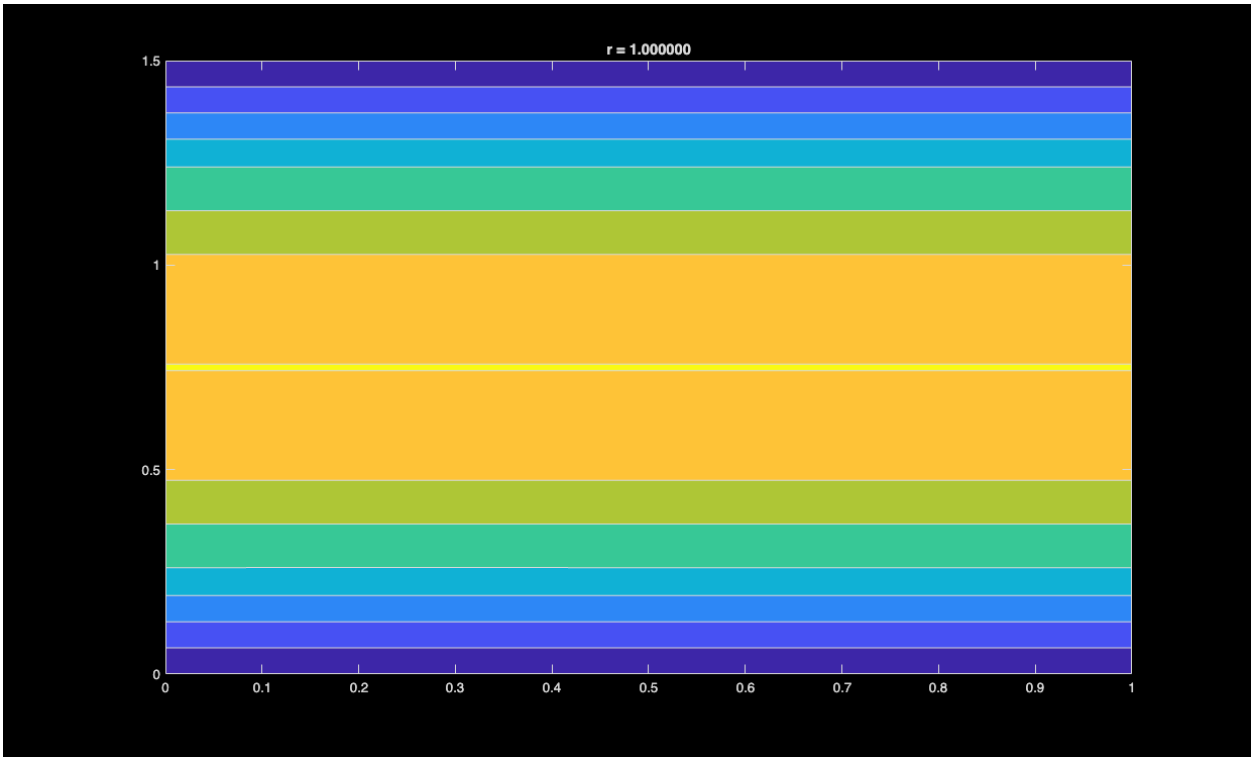



r = 0.250000
1.5
1
0.5
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1


r = 0.500000
80
70
60
50
40
30
20
10
0
7
6
5
4
3
2
1
1
1.5
2
2.5
3
3.5
4
4.5
5


r = 0.500000
1.5
1
0.5
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1


r = 1.000000
80
70
60
50
40
30
20
10
0
7
6
5
4
3
2
1
1
1.5
2
2.5
3
3.5
4
4.5
5



Part 3 (Post Processing / Plotting)

```
figure;
surf(T);
title('Temperature Distribution Crank Nicholson');
xlabel('x'); ylabel('y'); zlabel('T');

x_coords = linspace(0, Width, nx);
y_coords = linspace(0, Height, ny);
[X, Y] = meshgrid(x_coords, y_coords);

figure;
contourf(X, Y, T);
title('Temperature Distribution Crank Nicholson');
hold on;
colorbar;
```

Table of Contents

Assignemnt

PDE's

```
% Name         : Mohamed Mafaz
% Roll Number : AM25M009
% Department   : Applied Mechanics
```

Part 1 (Preprocessing)

```
clc;
clear;
close all;

% Parameters
alpha = 1;
h = 0.1;
k = 0.001;

x_start = 0;
x_end = 1;
t_start = 0;
t_end = 0.1;

% Discretize domain
xs = x_start:h:x_end;
ts = t_start:k:t_end;

nx = length(xs);
nt = length(ts);

% Initialize
T = zeros(nx, nt);       % FTCS
T_CN = zeros(nx, nt);    % Crank–Nicolson

% Initial condition
for i = 1:nx
    T(i,1) = sin(pi * xs(i)) + sin(2*pi * xs(i));
    T_CN(i,1) = T(i,1);
end

T(1,:) = 0;
```

```
T(end,:) = 0;
T_CN(1,:) = 0;
T_CN(end,:) = 0;
```

Part 2 (Processing / Using the Algorithms)

FTCS

```
for j = 1:nt-1
    for i = 2:nx-1
        T(i,j+1) = T(i,j) + (alpha * k / h^2) * (T(i+1,j) - 2*T(i,j) +
T(i-1,j));
    end
end
```

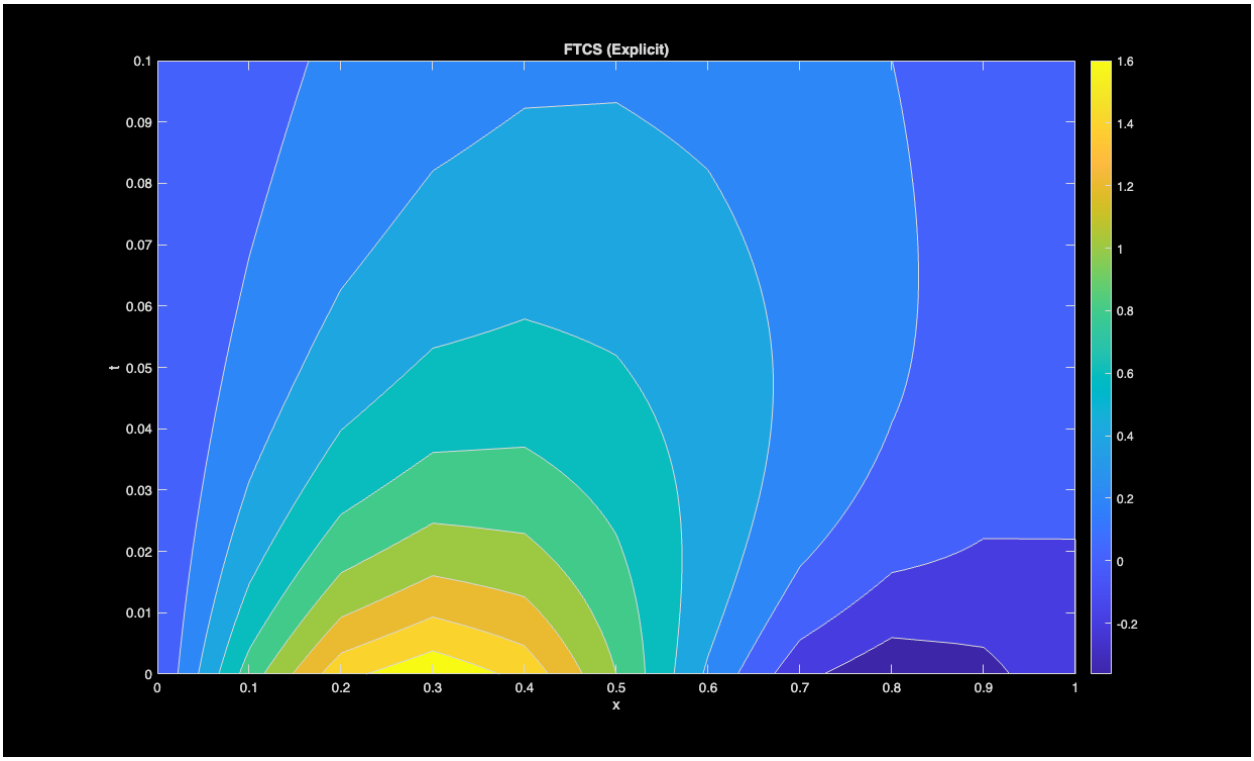
Crank–Nicolson

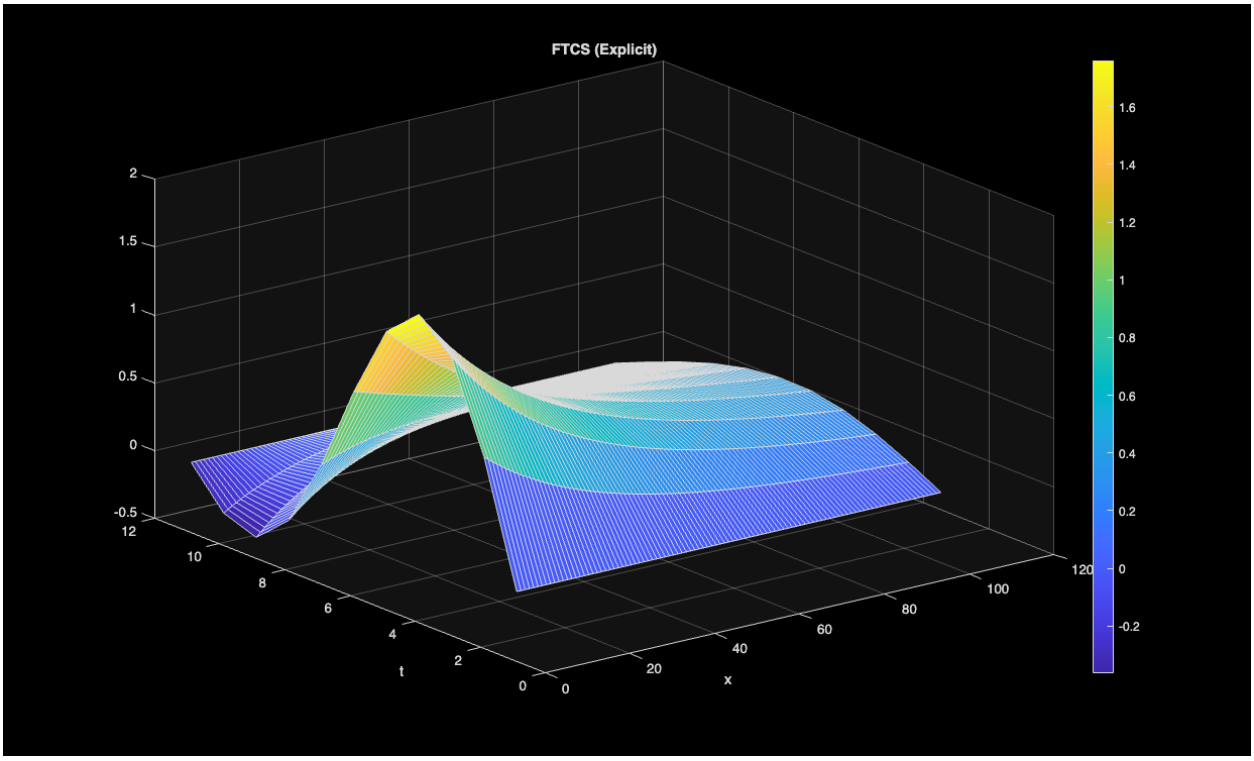
```
r = alpha * k / (2 * h^2);
A = diag((2 + 2*r) * ones(nx, 1)) + diag(-r * ones(nx-1, 1), 1) + diag(-r *
ones(nx-1, 1), -1);

% Boundary conditions
A(1,:) = 0;
A(1,1) = 1;
A(nx,:) = 0;
A(nx,nx) = 1;

for j = 1:nt-1
    Tj = T_CN(:, j);
    d = zeros(nx, 1);
    d(1) = 0;
    d(nx) = 0;
    for i = 2:nx-1
        d(i) = r*Tj(i-1) + (2 - 2*r)*Tj(i) + r*Tj(i+1);
    end
    T_CN(:, j+1) = A \ d;
end


% r = alpha * k / (2 * h^2);
% A = diag((2 + 2*r) * ones(nx, 1)) + diag(-r * ones(nx-1, 1), 1) + diag(-r
* ones(nx-1, 1), -1);
%
% Boundary conditions
% A(1,:) = 0;
% A(1,1) = 1;
% A(nx,:) = 0;
% A(nx,nx) = 1;
%
% tolerance = 1e-10;
% max_iter = 1000;
%
% T_history = [];
```

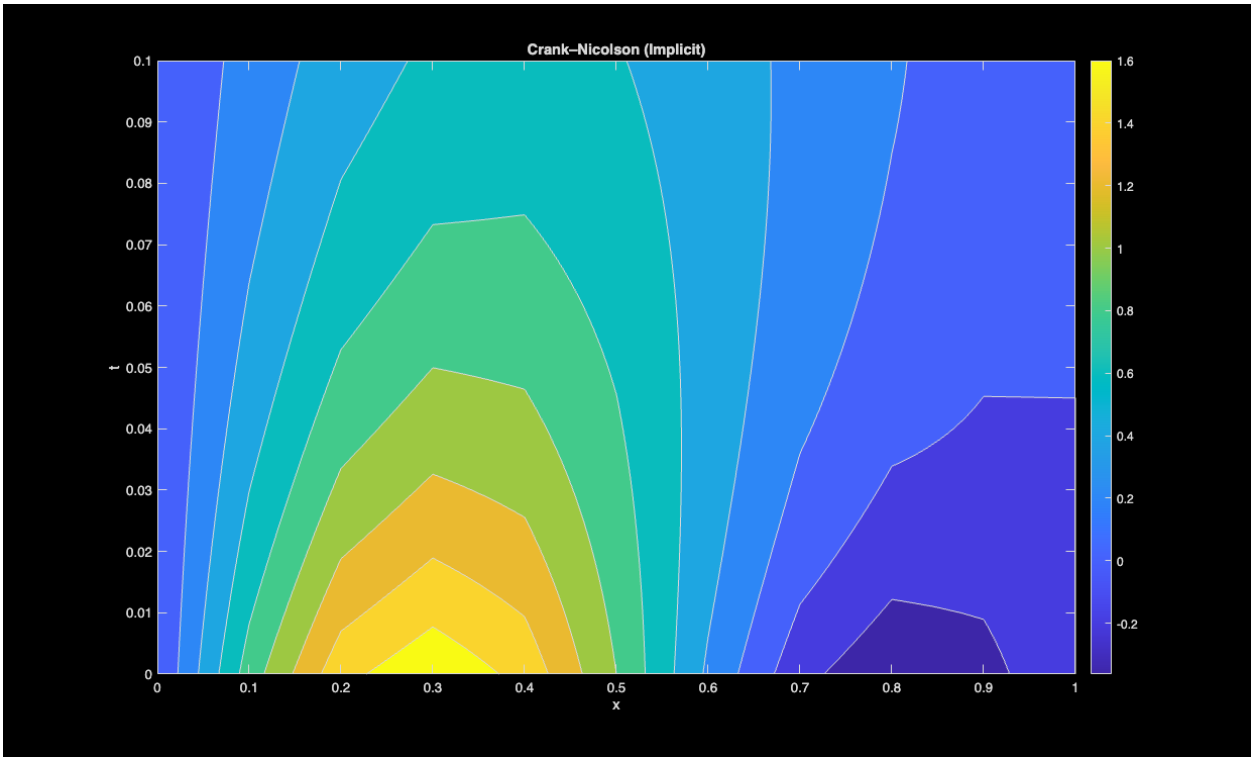
```
%
% for j = 1:nt-1
%     Tj = T_CN(:, j);
%     d = zeros(nx, 1);
%     d(1) = 0; d(nx) = 0;
%     for i = 2:nx-1
%         d(i) = r*Tj(i-1) + (2 - 2*r)*Tj(i) + r*Tj(i+1);
%     end
%
%     T_new = T_CN(:, j);  % initial guess
%     T_history = zeros(max_iter, nx);
%
%     for iter = 1:max_iter
%         T_old = T_new;
%         for i = 2:nx-1
%             T_new(i) = (r*T_new(i-1) + (2 - 2*r)*Tj(i) + r*T_new(i+1)) /
(2 + 2*r);
%         end
%         T_history(iter, :) = T_new;
%         if max(abs(T_new - T_old)) < tolerance
%             T_history = T_history(1:iter, :);
%             break;
%         end
%     end
%
%     T_CN(:, j+1) = T_new;
% end
```

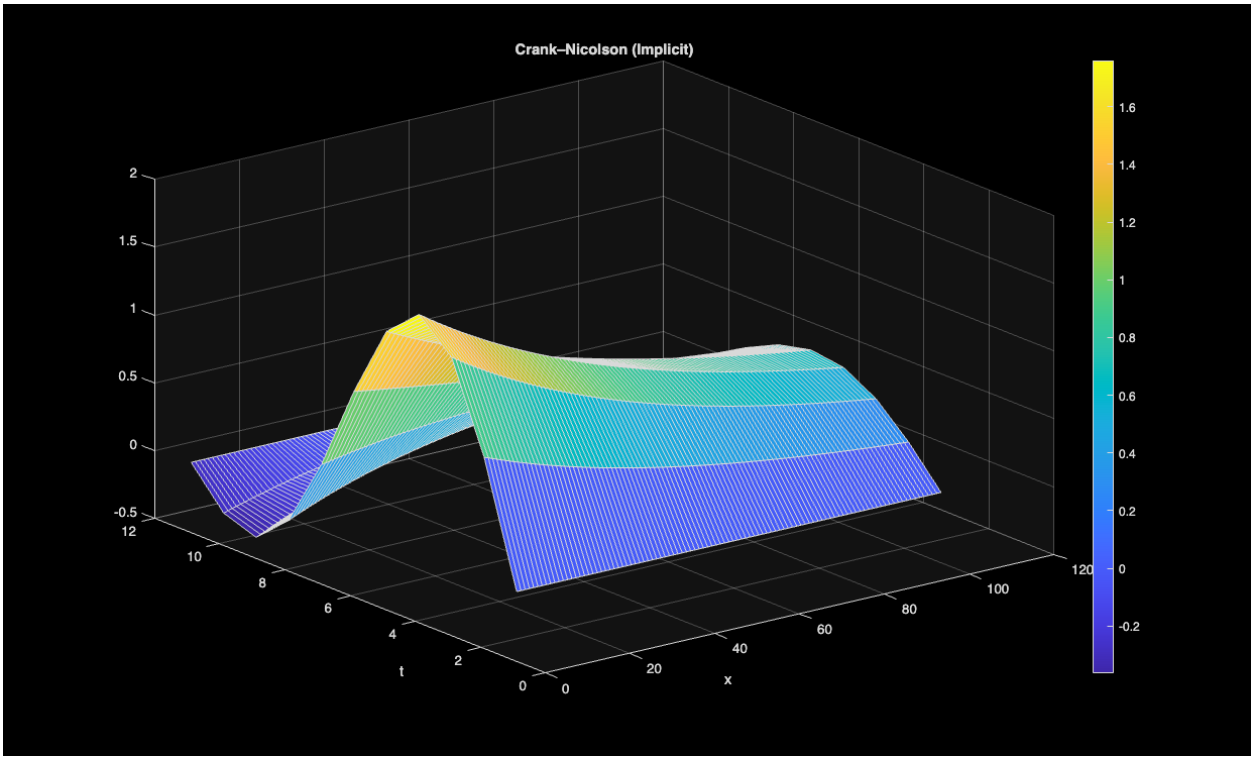
Part 3 (Post Processing / Plotting)

```
[X, Y] = meshgrid(xs, ts);

figure;
contourf(X, Y, T');
title('FTCS (Explicit)');
xlabel('x'); ylabel('t'); colorbar

figure;
surf(T);
title('FTCS (Explicit)');
xlabel('x'); ylabel('t'); colorbar

figure;
contourf(X, Y, T_CN');
title('Crank–Nicolson (Implicit)');
xlabel('x'); ylabel('t'); colorbar


figure;
surf(T_CN);
title('Crank–Nicolson (Implicit)');
xlabel('x'); ylabel('t'); colorbar
```


FTCS (Explicit)
t
x
0
0.01
0.02
0.03
0.04
0.05
0.06
0.07
0.08
0.09
0.1
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
-0.2


FTCS (Explicit)
2
1.5
1
0.5
0
-0.5
12
10
8
6
4
2
0
t
0
20
40
60
80
100
120
x
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
-0.2





Published with MATLAB® R2025a

Table of Contents

Assignemnt

PDE's

```
% Name         : Mohamed Mafaz
% Roll Number : AM25M009
% Department   : Applied Mechanics

clc;
clear;
close all;
```

Part 1 (Preprocessing)

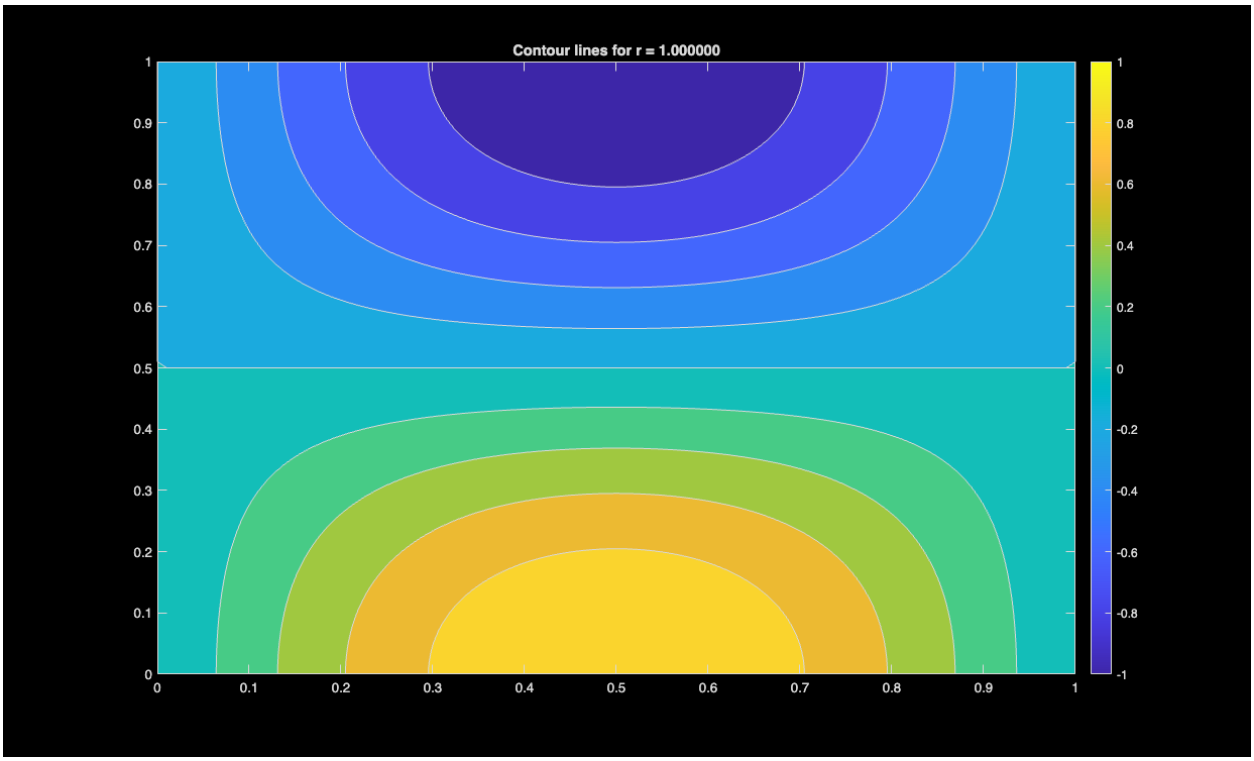
```
h = 0.01;
k = 0.01;

x_start = 0;
x_end = 1;
t_start = 0;
t_end = 1;

xs = x_start:h:x_end;
ts = t_start:k:t_end;

top_bc = 0;
bottom_bc = 0;

matrix = zeros(length(xs), length(ts));
matrix(1, :) = top_bc;
matrix(end, :) = bottom_bc;


for i = 1:length(xs)
    matrix(i, 1) = sin(pi * xs(i));
end
```

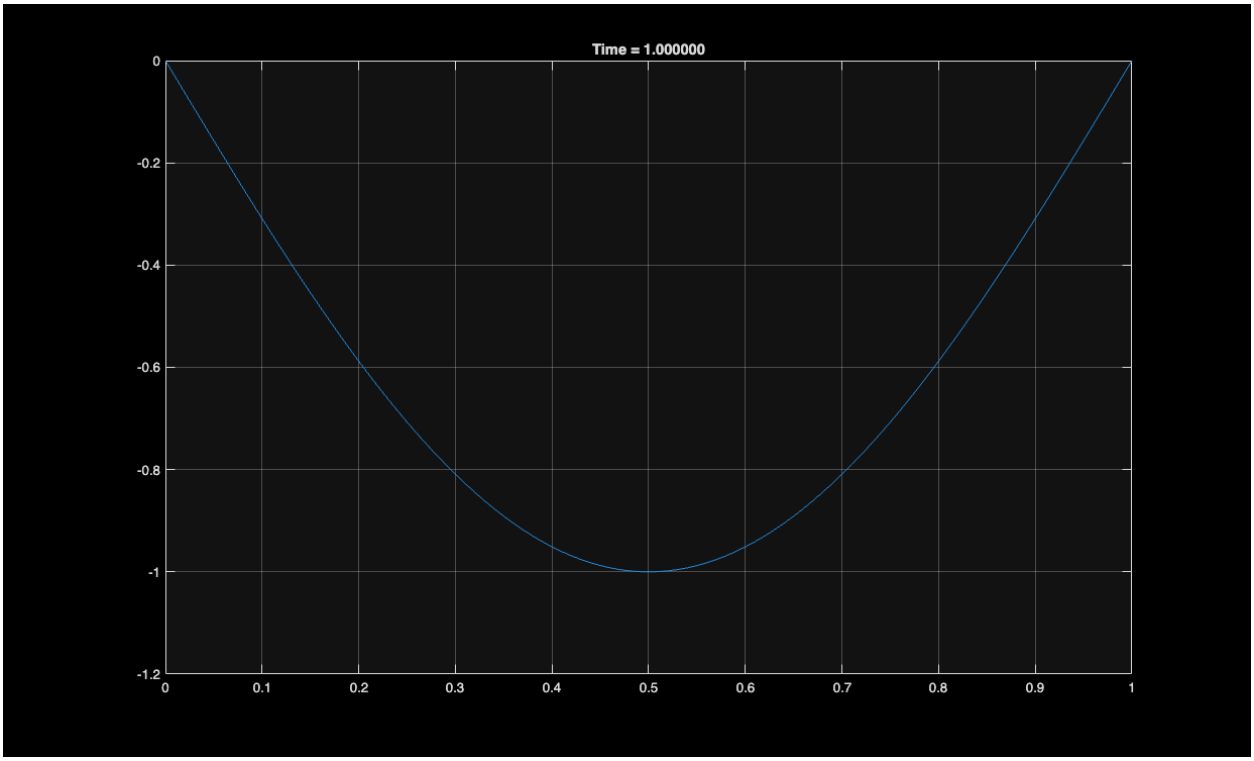
Part 2 (Processing / Using the Algorithms)

```
r = (k / h);
for i = 2:length(xs)-1
    matrix(i, 2) = matrix(i, 1) + (r^2)/2 * (matrix(i-1, 1) - 2 * matrix(i,
```

```
1) + matrix(i+1, 1));
end

for j = 2:length(ts)-1
    for i = 2:length(xs)-1

        matrix(i, j+1) = 2*matrix(i, j) - matrix(i, j-1) + r^2*(matrix(i+1,
j) - 2*matrix(i, j) + matrix(i-1, j));
    end
end


[X, Y] = meshgrid(xs, ts);
figure;
contourf(X, Y, matrix');
title(sprintf('Contour lines for r = %f', r));
colorbar;

figure;
surf(matrix)
title(sprintf('Surf for r = %f', r));
colorbar;

h = 0.01;
k = 0.01;

x_start = 0;
x_end = 1;
t_start = 0;
t_end = 1;

xs = x_start:h:x_end;
ts = t_start:k:t_end;

top_bc = 0;
bottom_bc = 0;

matrix = zeros(length(xs), length(ts));
matrix(1, :) = top_bc;
matrix(end, :) = bottom_bc;

for i = 1:length(xs)
    matrix(i, 1) = sin(pi * xs(i));
end

r = (k / h);
for i = 2:length(xs)-1
    matrix(i, 2) = matrix(i, 1) + (r^2)/2 * (matrix(i-1, 1) - 2 * matrix(i,
1) + matrix(i+1, 1));
end

for j = 2:length(ts)-1
    for i = 2:length(xs)-1
        matrix(i, j+1) = 2*matrix(i, j) - matrix(i, j-1) + r^2*(matrix(i+1,
```

```
j) - 2*matrix(i, j) + matrix(i-1, j));
    end

    plot(xs, matrix(:, j+1));
    title(sprintf('Time = %f', ts(j+1)));
    grid on;
    drawnow;
    pause(0.01)
end
```

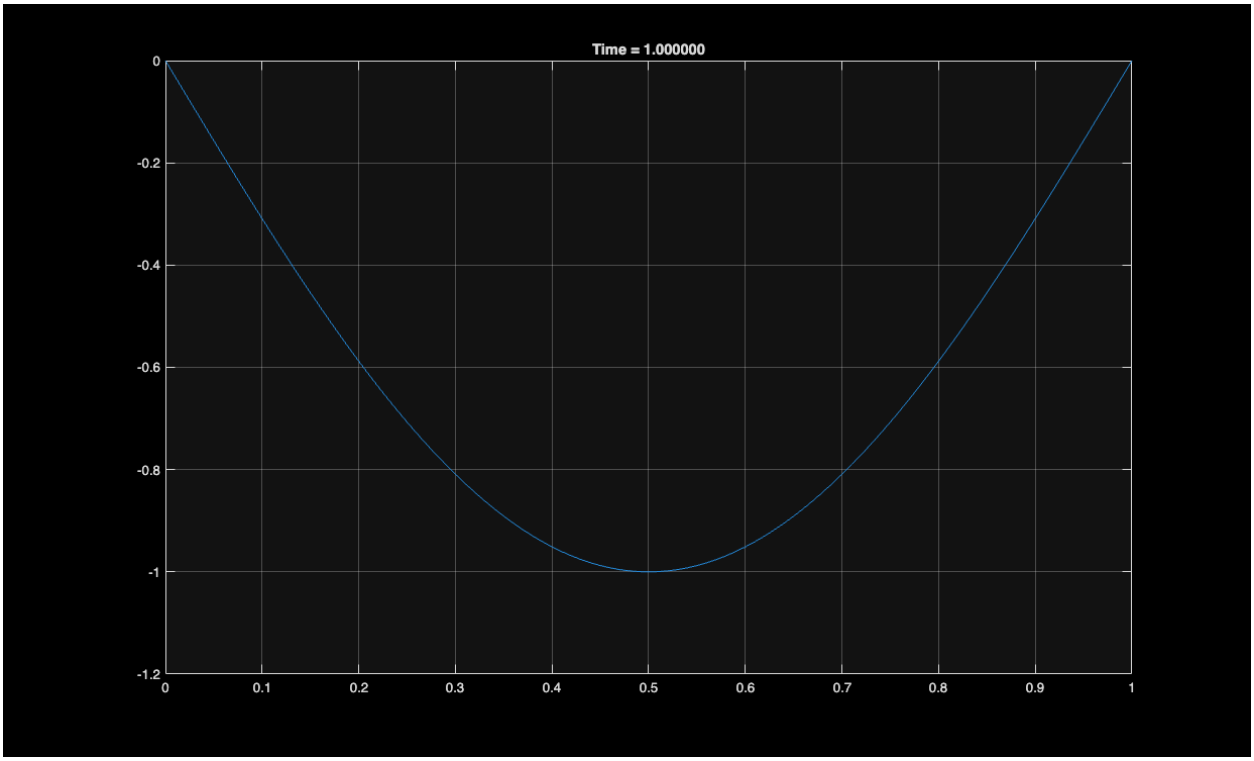


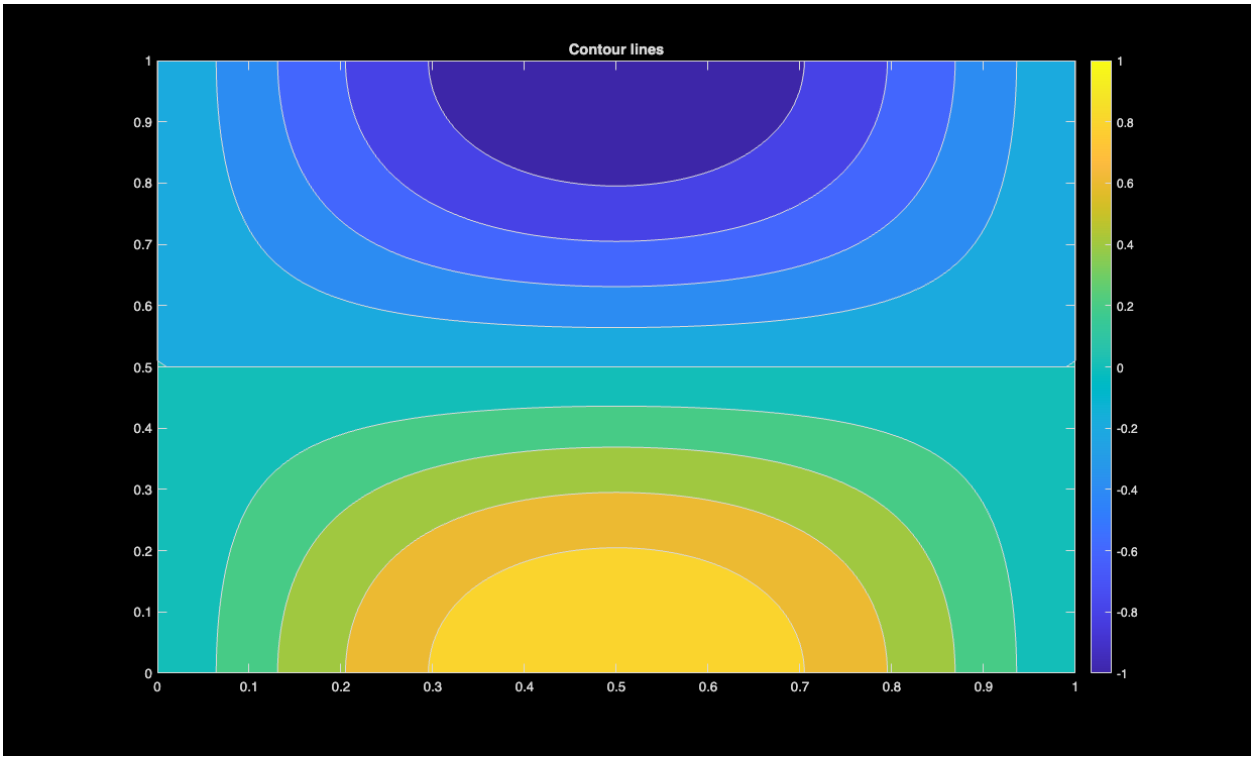


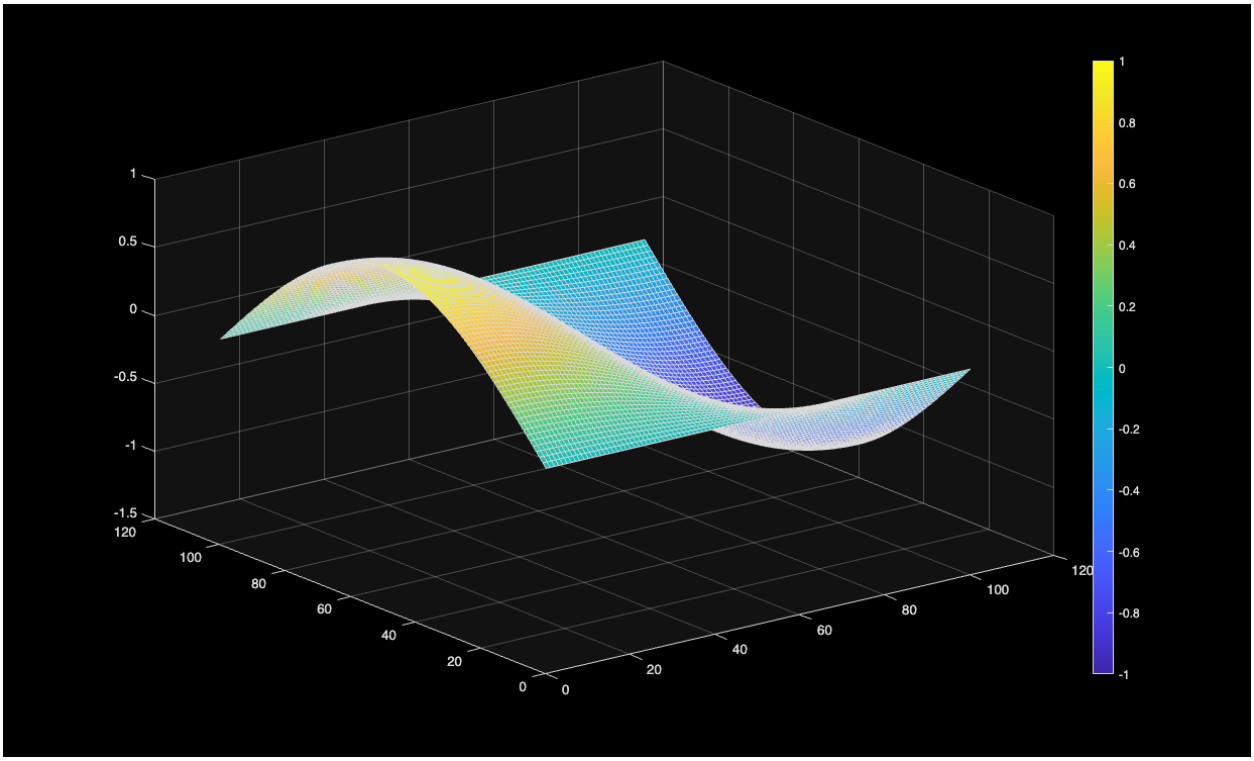
Part 3 (Post Processing / Plotting)

Final plots

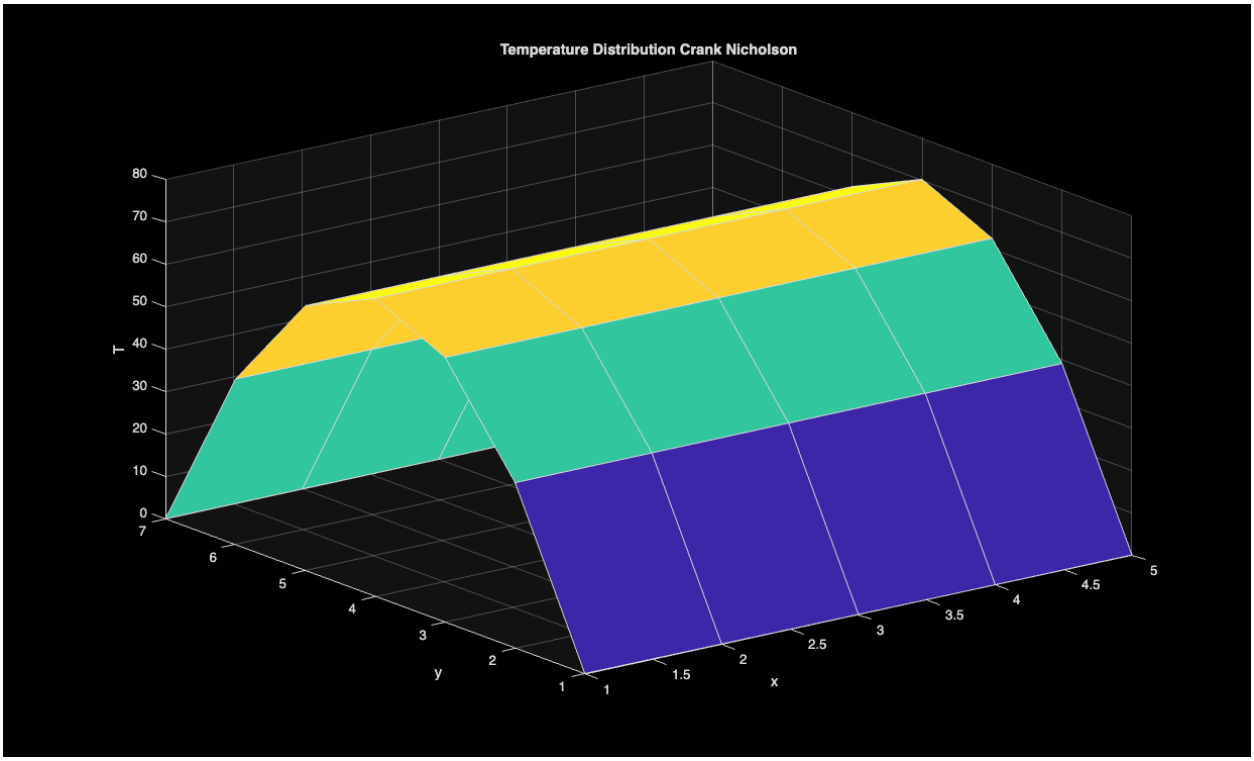
```
[X, Y] = meshgrid(xs, ts);
figure;
contourf(X, Y, matrix');
title('Contour lines');
colorbar;

figure;
surf(matrix)
colorbar;
```

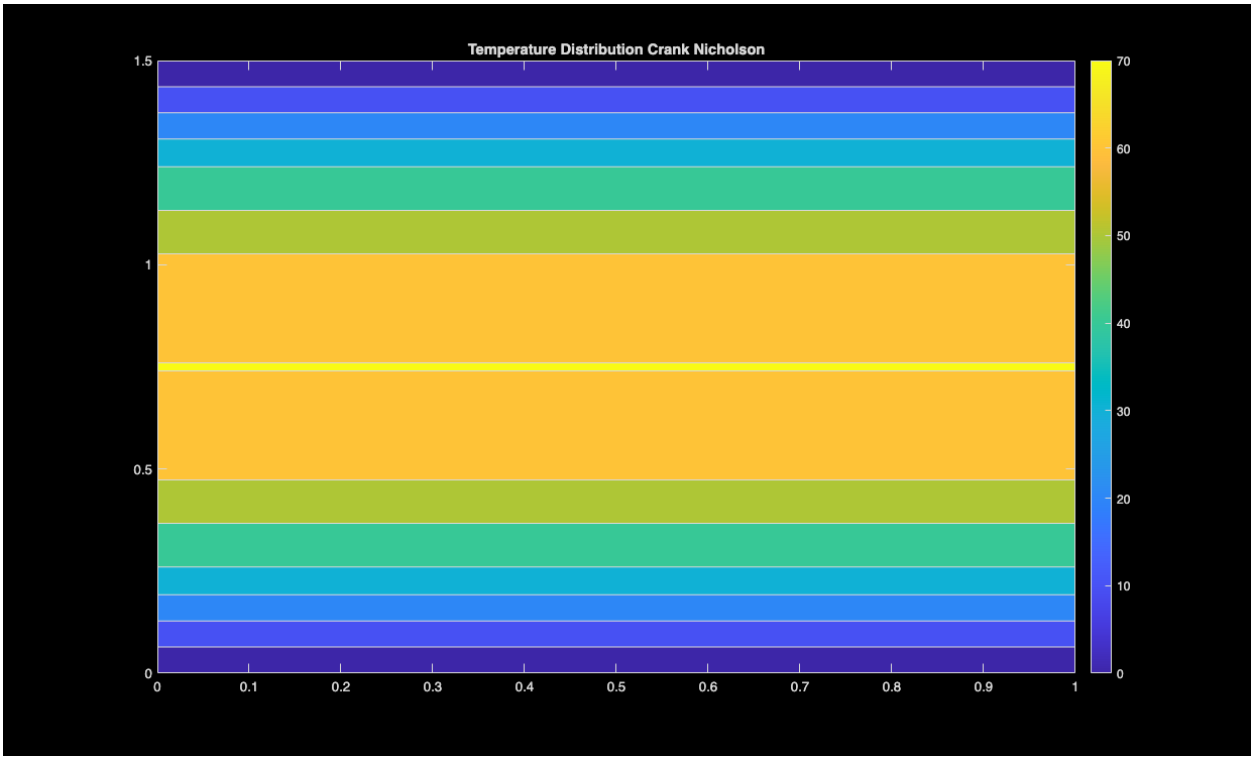

Time = 1.000000
0
-0.2
-0.4
-0.6
-0.8
-1
-1.2
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1


Contour lines
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1



Published with MATLAB® R2025a





Published with MATLAB® R2025a