Table of Contents

```
clear; % clear variables
clc; % clear command window
close all
```

4.1

```
clear; % clear variables
clc; % clear command window
close all
syms x t k
f = (exp(-t).*sin(2.*x));

d2f_dx2 = diff(f, x, 2);
df_dt = diff(f, t, 1);
disp(d2f_dx2);
disp(df_dt);
heat_eq = k*d2f_dx2 - df_dt;
disp(heat_eq);
```

```
-4*sin(2*x)*exp(-t)

-sin(2*x)*exp(-t)

sin(2*x)*exp(-t) - 4*k*sin(2*x)*exp(-t)
```

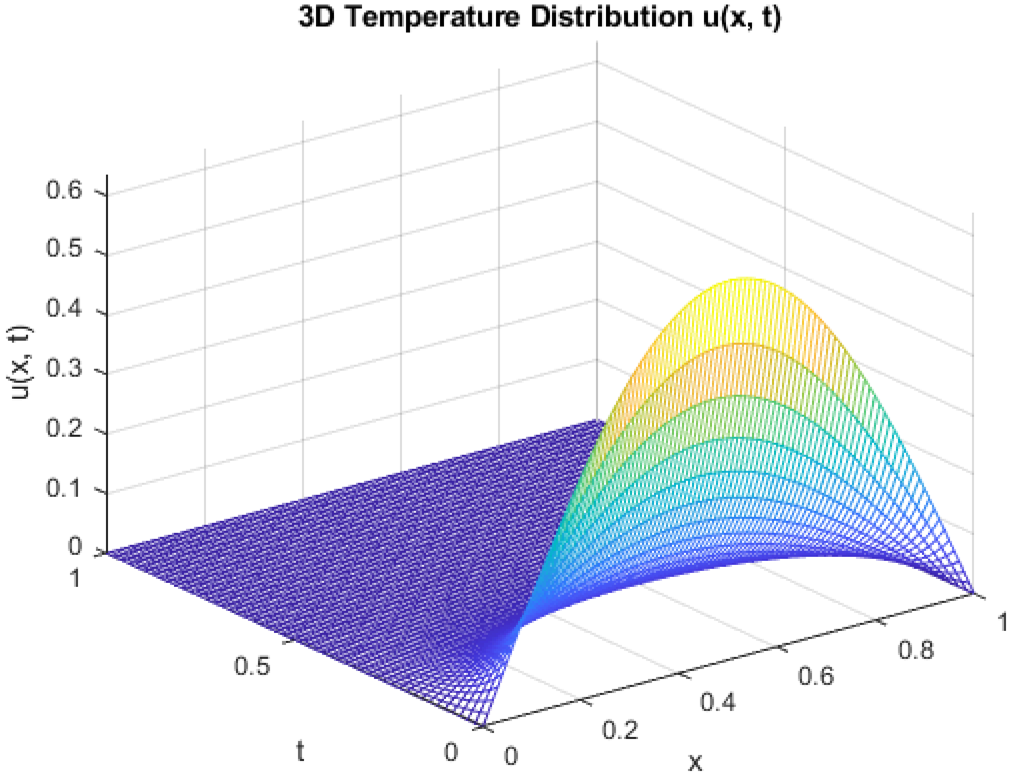
4.1

```
clear; % clear variables
clc; % clear command window
close all
syms x t k
f = x + 2 + (exp(-2.*t).*sin(2.*x));

d2f_dx2 = diff(f, x, 2);
df_dt = diff(f, t, 1);
disp(d2f_dx2);
disp(df_dt);
heat_eq = k*d2f_dx2 - df_dt;
disp(heat_eq);
```

```
-4*sin(2*x)*exp(-2*t)

-2*sin(2*x)*exp(-2*t)

2*sin(2*x)*exp(-2*t) - 4*k*sin(2*x)*exp(-2*t)
```

4.3

```
clear; % clear variables
clc; % clear command window
close all
% Constants and parameters
L = 1; % Length of the bar
k = 1; % Thermal diffusivity constant
n = 1; % Fourier series term (for b_n calculation)

% Step 1: Compute b_n
b_n = (2/L) * integral(@(x) x .* sin(n*pi*x/L), 0, L);

% Define spatial and temporal grids
x = linspace(0, L, 100); % Discretize x-axis
t = linspace(0, 1, 50); % Discretize time from 0 to 1

% Initialize a matrix to store temperature values
u_values = zeros(length(x), length(t));

% Calculate u(x, t) for all x and t values
for i = 1:length(t)
    u_values(:, i) = b_n * sin(n*pi*x/L) * exp(-(n^2*pi^2*k*t(i)/L^2));
end

% Create a 3D mesh plot
[X, T] = meshgrid(x, t);
figure;
mesh(X, T, u_values');
xlabel('x');
ylabel('t');
zlabel('u(x, t)');
title('3D Temperature Distribution u(x, t)');
```

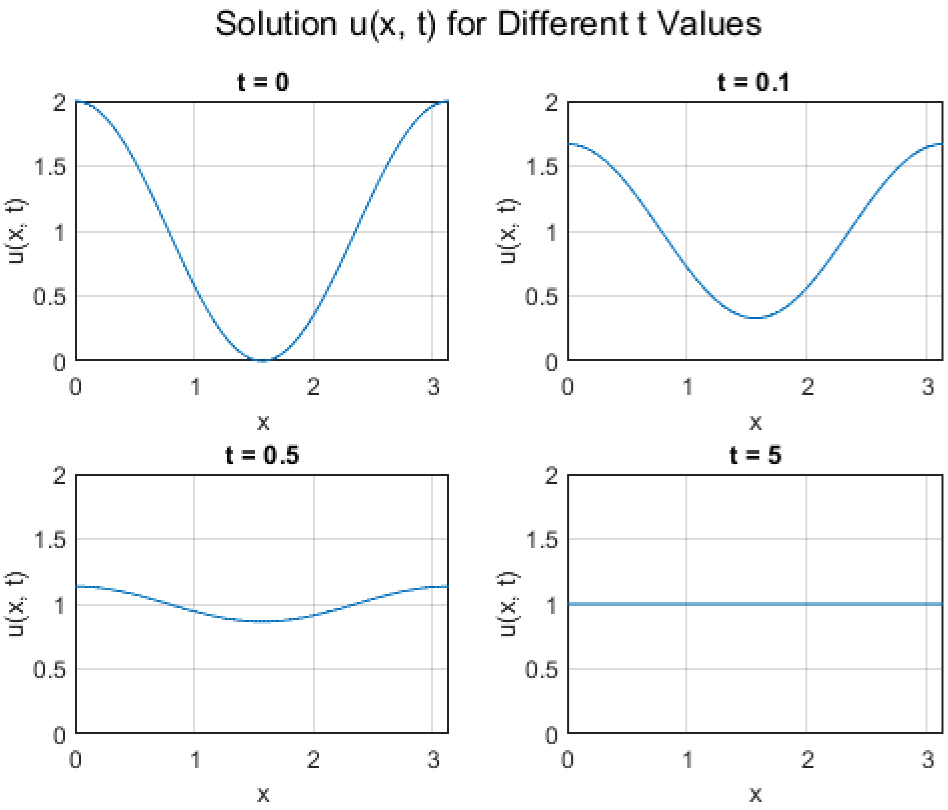


6 iv 1-D heat equation

Define the values of t you want to plot

```
t_values = [0, 0.1, 0.5, 5];

% Define the spatial variable x
x = linspace(0, pi, 100); % Adjust the number of points for accuracy

% Initialize a figure for the plots
figure;

% Loop through each value of t and plot u(x, t)
for i = 1:length(t_values)
    t = t_values(i);
    u = 1 + cos(2 * x) .* exp(-4 * t);

    subplot(2, 2, i); % Create a 2x2 grid of subplots
    plot(x, u);
    title(['t = ', num2str(t)]);
    xlabel('x');
    ylabel('u(x, t)');
    ylim([0, 2]); % Set the y-axis limits for consistency

    grid on;
end
```

```
% Adjust the overall plot title
sgtitle('Solution u(x, t) for Different t Values');
```



Published with MATLAB® R2023a