

# MATLAB Workbook

## **CME104**

### **Linear Algebra and Partial Differential Equations for Engineers**



*Authors*

*Jennifer Chien*

*Perrine Pepiot*

*Vadim Khayms*

# Linear Algebra and Applications

## 1. Matrices and Vectors

<b>Commands :</b>	<b>*</b>	:	Matrix multiplier operator
	<b>size</b>	:	Returns the size of a vector or matrix
	<b>'</b>	:	Matrix transpose operator
	<b>round</b>	:	Returns integer value(s) of the argument

Multiply two matrices  $A = \begin{bmatrix} 1 & 2 \\ -1 & 3 \end{bmatrix}$  and  $B = \begin{bmatrix} 1 & 0 & 1 \\ 2 & 1 & 0 \end{bmatrix}$ . Determine the size of the resulting matrix. Display the bottom left-element and the second column on the screen.



### SOLUTION

#### CODE from M-file

```
clear

% Initialize matrices
A=[1 2; -1 3];
B=[1 0 1; 2 1 0];

% Calculate and display
product of matrices
C=A*B

% Calculate and display size
of new matrix
Csize = size(C)

% Display bottom left-
element and second column
C(2,1)
```

C(:,2)

#### OUTPUT

```
C =
     5     2     1
     5     3    -1

Csize =
     2     3

ans =
     5

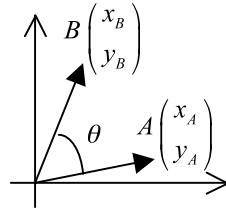
ans =
     2
     3
```



### YOUR TURN

- Find the dot product between the first and the second rows of matrix  $C = \begin{bmatrix} 5 & 2 & 1 \\ 5 & 3 & -1 \end{bmatrix}$ .
- Compute  $B^T A^T - B^T$  using the matrices  $A = \begin{bmatrix} 1 & 2 \\ -1 & 3 \end{bmatrix}$  and  $B = \begin{bmatrix} 1 & 0 & 1 \\ 2 & 1 & 0 \end{bmatrix}$ .
- A rectangle is formed by the four vertices: (1,0), (3,0), (1,2), and (3,2). Compute the co-ordinates of each vertex after a rotation of  $60^\circ$  (or  $\frac{\pi}{3}$  rad) using a rotation matrix  $R$ . Plot the rectangle before and after the rotation.

Hint :



$$\begin{pmatrix} x_B \\ y_B \end{pmatrix} = R \begin{pmatrix} x_A \\ y_A \end{pmatrix} \quad \text{with} \quad R = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix}$$

Hint: To plot any set of points linked together in MATLAB, create a matrix whose first line contains the x-components of each point and the second one, the y-components. Then plot the first line versus the second one. To close your rectangles, use the first vertex as first and last point.

- d) Each year, a college accepts 1000 freshmen and 200 sophomores into the school. Also, 10% of the freshman, 25% of the sophomores, 5% of the juniors, and 5% of the seniors repeat their year. The rest continue their studies. If there are 1160 freshmen, 1057 sophomores, 1183 juniors, and 1028 seniors this year and the trend holds, what is the school population in 10 years? The system of equations to use is as follows:

$$\begin{aligned} x_1(t+1) &= 0.1 \cdot x_1(t) + 1000 \\ x_2(t+1) &= 0.9 \cdot x_1(t) + 0.25 \cdot x_2(t) + 200 \\ x_3(t+1) &= 0.75 \cdot x_2(t) + 0.05 \cdot x_3(t) \\ x_4(t+1) &= 0.95 \cdot x_3(t) + 0.05 \cdot x_4(t) \end{aligned}$$

where the number is indicative of the student's year in college (1=freshman, 2=sophomore, etc). From the equations, we can derive an equation  $x^{(t+1)} = Ax^{(t)} + b$  with

$$\text{matrix } A = \begin{bmatrix} 0.1 & 0 & 0 & 0 \\ 0.9 & 0.25 & 0 & 0 \\ 0 & 0.75 & 0.05 & 0 \\ 0 & 0 & 0.95 & 0.05 \end{bmatrix}, \quad b = \begin{bmatrix} 1000 \\ 200 \\ 0 \\ 0 \end{bmatrix}, \quad \text{and } x_0 = \begin{bmatrix} 1160 \\ 1057 \\ 1183 \\ 1028 \end{bmatrix}.$$

Hint: When displaying the final population vector, use the **round** function to display integral values. We cannot have partial students.

## 2. Gaussian Elimination

**Commands :**

<b>rank</b>	:	Returns the rank of a matrix
<b>rref</b>	:	Returns the reduced echelon of a matrix
<b>inv</b>	:	Returns the inverse of a matrix

Find the rank and solution (if it exists) to the following system of equations:

$$\begin{aligned} 7y + 3z &= -12 \\ 2x + 8y + z &= 0 \\ -5x + 2y - 9z &= 26 \end{aligned}$$

using the reduced row echelon method, inverse method, and Gaussian elimination method.



### SOLUTION

CODE from M-file  
clear

% Initialize matrix. See note  
below  
A = [0,7,3,-12;2,8,1,0;-5,2,-9,26];

```
% Calculate and display rank of
matrix
Arank = rank(A)

% Calculate and display reduced
row echelon form
rref(A)

% Calculate and display inverse
method
A = [0,7,3,; 2,8,1; -5,2,-9];
b = [-12;0;26];
x = inv(A)*b

% Calculate and display Gaussian
elimination
x = Ab
```

```
Arank =
3
x =
1 0 0 2
0 1 0 0
0 0 1 -4
x =
2
0
-4
x =
2
0
-4
```

#### OUTPUT

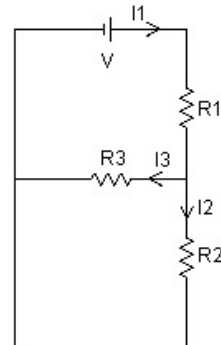
Note: The function **rref** needs a matrix input in the form of

$A = \begin{bmatrix} x_1 & y_1 & z_1 & \dots & c_1 \\ x_2 & y_2 & z_2 & \dots & c_2 \\ \dots & \dots & \dots & \dots & \dots \end{bmatrix}$ . For the matrix that **rref** produces, the last column contains the solutions for the matrix. The first element is the solution for x, the second for y, and the last for z.



### YOUR TURN

A loop is set up like in the figure to the right. The potential of the battery is  $V = 50$ . The resistors have values of  $R_1 = 3$ ,  $R_2 = 2$ , and  $R_3 = 14/5$ . Find the currents of the network, depicted on the right. The equations of the currents in this loop are  $I_1 - I_2 - I_3 = 0$ ,  $3I_1 + \frac{14}{5}I_3 = 50$  (top loop), and  $2I_2 - \frac{14}{5}I_3 = 0$  (bottom loop).



## 3. Least Squares

**Commands :** **polyfit** : Finds the coefficients of a polynomial to fit the data in a least squares sense.

Data from a current flowing across a capacitor is recorded. If the theoretical decrease in current is supposed to be linear, plot the graph that best represents the data: (1,0), (2,6), (3,5), (4,2), and (5,0).



### SOLUTION

#### CODE from M-file

```
clear
close

% Initialize raw data
t = [1,2,3,4,5];
ldata = [9,6,5,2,0];

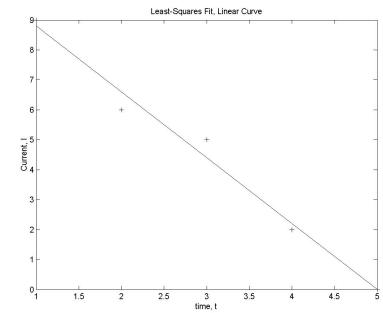
% Calculate coefficients of
polynomial
```

```
p = polyfit(t,ldata,1);
% Calculate range for best fit
l = p(1)*t+p(2);

% Plot raw data
plot(t,ldata,'+')
hold on
% Plot linear fit
plot(t,l)
```

```
title('Least-Squares Fit, Linear
Curve')
xlabel('time, t')
ylabel('Current, I')
```

### OUTPUT



*Note:* The vector  $p$  contains the coefficients of a linear function. The '1' represents a polynomial to the 1<sup>st</sup> degree.



## **YOUR TURN**

In a laser experiment, current and laser power is collected in a test run. Find the best quadratic curve to fit the data.

Current (milliamps)	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
Laser Power (microwatts)	0.000	0.000	0.001	0.002	0.018	0.056	0.115	0.201	0.301	0.523

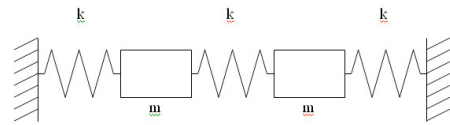
*Hint:* For a quadratic fit, find  $p$  such that  $y = p(1)x^2 + p(2)x + p(3)$ .

# Eigenvalues and Eigenvectors

## 4. Eigenvalues and Eigenvectors

**Commands :** `eig` : Returns eigenvalues and eigenvectors

Two blocks of mass  $m$  are placed in-between three springs with constant  $k$ . A system of equations can be set up such that  $m\ddot{x}_1 = -2kx_1 + kx_2$  and  $m\ddot{x}_2 = kx_1 - 2kx_2$ . If  $k = 1$  and  $m = 1$ , and an initial displacement of the blocks are  $x_1 = 0.1$ , and  $x_2 = -0.3$ , find the equation of the position of the block as a function of time. Plot the displacement of the blocks as a function of time with  $dt = 0.1$  for the first twenty seconds. Draw a picture of the mode shapes of the blocks.



*Note:* The negative sign means that the block is displaced to the left.



### SOLUTION

#### CODE from M-file

```
clear
close

% Initialize constants and matrix
m = 1;
k = 1;
A = [-2*k/m, k/m; k/m, -2*k/m];
% Calculate and display
eigenvalues and eigenvectors
[Q,d] = eig(A)
% Calculate constant C
x0 = [0.1; -0.3];
C = inv(Q)*x0;

% Calculate and plot motion of
blocks
t = linspace(0,20,200);
w=sqrt(-d);
x = C(1)*Q(:,1)*cos(w(1,1)*t) +
C(2)*Q(:,2)*cos(w(2,2)*t);
plot(t,x)
title('Displacement of blocks as a
function of time')
xlabel('Time')
ylabel('Displacement')
legend('x1','x2')
```

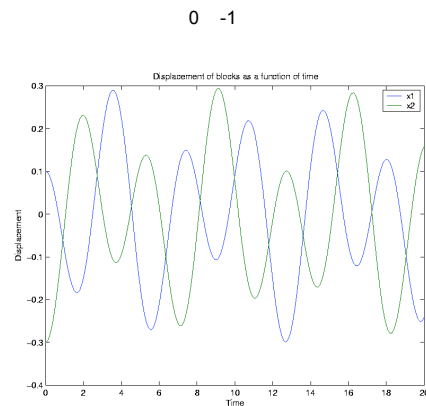
#### OUTPUT

```
Q =

    0.7071    0.7071
   -0.7071    0.7071

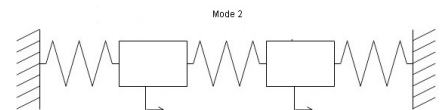
d =

   -3     0
```

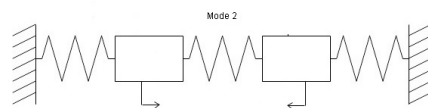


#### Drawing of mode shapes

Mode 1



Mode 2

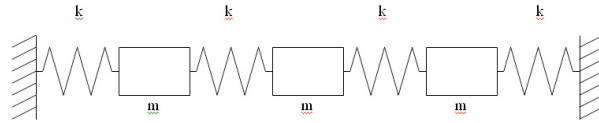




## YOUR TURN

Three blocks of mass  $m$  are placed in-between four springs with constant  $k$ . A system of equations can be set up such that

$m\ddot{x}_1 = -2kx_1 + kx_2$ ,  $m\ddot{x}_2 = kx_1 - 2kx_2 + kx_3$ , and  $m\ddot{x}_3 = kx_2 - 2kx_3$ . If  $k = 1.5$ ,  $m = 2$ , and the initial displacement of the blocks are  $x_1 = 0.1$ ,  $x_2 = -0.9$ , and  $x_3 = 0.3$ , find the equation of the position of the block as a function of time. Plot the displacement of the blocks as a function of time with  $dt = 0.01$  for the first thirty seconds. Draw a picture of the mode shapes of the blocks.



## 5. Power of Matrix

Given the matrix  $A = \begin{bmatrix} 2 & 9 \\ 4 & -3 \end{bmatrix}$ , calculate  $A^5$  directly, and then using the diagonalization method.



## SOLUTION

### CODE from M-file

```
clear

% Initialize variables
A = [2,9;4,-3];

% Direct computation and display answer
A5_1 = A^5

% Diagonalization
[Q,d] = eig(A);
% Display answer
A5_2 = Q*d^5*inv(Q)
```

### OUTPUT

```
A5_1 =
    212    17019
    7564    -9243

A5_2 =
    212    17019
    7564    -9243
```



## YOUR TURN

Suppose each year, 5% of the students at UC Berkeley transfer to Stanford and 1% of the students at Stanford transfer to Berkeley. Determine the population after 5 years if there were initially 50,000 students in Berkeley and 10,000 students in Stanford. The

transformation matrix to use is  $A = \begin{bmatrix} 0.99 & 0.05 \\ 0.01 & 0.95 \end{bmatrix}$ .

Note:  $x_1$  is the Stanford population and  $x_2$  is the Berkeley population.



# Fourier Series

## 6. Fourier Series

For the function  $f(x) = \begin{cases} -1 & -\pi < x < 0 \\ 1 & 0 < x < \pi \end{cases}$ , the odd Fourier expansion is given by

$$f(x) = \sum_{n=\text{odd}}^{\infty} \frac{4}{n\pi} \sin(nx).$$

Plot the actual function and first 3 partial sums over the domain

$-2\pi < x < 2\pi$ , all on the same set of axes. Observe how the Fourier series is accurate over many periods.



### SOLUTION

#### CODE from M-file

```
clear
close

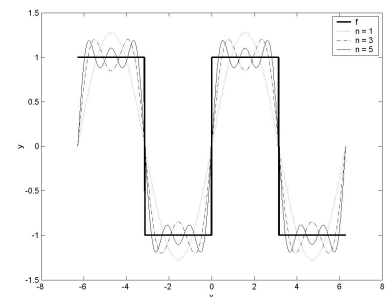
% Maximum iteration for the
fourier series
nmax = 5;
% Domain of plot
x = linspace(-2*pi,2*pi,1000);

% Plot of function f(x)
for i = 1:length(x)
    if x(i) <= -pi | (x(i) > 0 & x(i) <=
pi)
        f(i) = 1;
    end
    if (x(i) > -pi & x(i) <= 0) | x(i) > pi
        f(i) = -1;
    end
end
plot(x,f)
hold on

% Calculate and plot first 3 partial
sums
A0 = 0;
y = A0;
for n=1:2:nmax
```

```
An = 0;
Bn = 4/(n*pi);
y = y + An*cos(n*x) +
Bn*sin(n*x);
plot(x,y)
hold on
end
xlabel('x')
ylabel('y')
legend('f','n = 1','n = 3','n = 5')
```

#### OUTPUT



### YOUR TURN

For the function  $f(x) = 2x$ ,  $-4 < x < 4$ , given the Fourier expansion

$$f(x) = \sum_{n=1}^{\infty} \frac{8}{n\pi} (-1)^{n+1} \sin\left(\frac{n\pi x}{2}\right),$$

plot the actual function and first 5 partial sums, all on the

same set of axes.

## 7. Forced Vibrations

**Commands :** `stem` : plots a graph of discrete points

Forced oscillations of a body of mass  $m$  on a spring of modulus  $k$  are governed by the equation:

$$m \frac{d^2 y}{dt^2} + \lambda \frac{dy}{dt} + ky = f(t) \quad (1)$$

with  $y(t)$  displacement of the mass from rest

$\lambda$  damping constant

$f(t)$  external force depending on time applied to the mass-spring system

Using  $\lambda = 0.02$  gm/sec,  $k = 40$  gm/sec<sup>2</sup>,  $m = 0.05$  gm and the following external force:

$$f(t) = \begin{cases} t - \frac{1}{2} & 0 < t < 1 \\ -t - \frac{1}{2} & -1 < t < 0 \end{cases} \quad \text{and } f(t+2) = f(t)$$

- Solve equation (1) using Fourier series
- Plot :
  - the forcing function
  - the displacement of the mass
  - the contribution of each frequency component to the final solution.

What is the relation between the two last plots?

### SOLUTION

- The Fourier series of the external force is :  $f(t) = \sum_{n \text{ odd}} -\frac{4}{n^2 \pi^2} \cos(n\pi t)$ . The

differential equation is then :  $m \frac{d^2 y}{dt^2} + \lambda \frac{dy}{dt} + ky = \sum_{n \text{ odd}} -\frac{4}{n^2 \pi^2} \cos(n\pi t)$ . Assuming a

solution of the form:  $y(t) = \sum_{n=1}^{\infty} A_n \cos(n\pi t) + B_n \sin(n\pi t)$ , we obtain the following

Fourier coefficients:  $A_n = \frac{4}{n^2 \pi^2} \frac{mn^2 \pi^2 - k}{(mn^2 \pi^2 - k)^2 + (\lambda n \pi)^2}$  and

$B_n = -\frac{4}{n^2 \pi^2} \frac{\lambda n \pi}{(mn^2 \pi^2 - k)^2 + (\lambda n \pi)^2}$ . The contribution of each frequency

component to the final solution is given by  $C_n = \sqrt{A_n^2 + B_n^2}$

#### CODE from M-file

```
clear
close all

% Parameters of the system
m = 0.05; % Mass
k = 40; % Stiffness
L = 0.02; % Damping constant

% Time
t = -5:0.01:5;

% Initialization
f = zeros(1,length(t)); %
External force
y = zeros(1,length(t)); %
Displacement of the mass
```

```
C = zeros(100,1); % Amplitude
of each frequency
```

```
for n = 1:2:100
```

```
    f = f - 4/(n*pi)^2*cos(n*pi*t);
```

```
    An = 4/(n*pi)^2*(m*n^2*pi^2 - k)/((m*n^2*pi^2 - k)^2 + (L*n*pi)^2);
    Bn = An*(L*n*pi)/(m*n^2*pi^2 - k);
    C(n) = sqrt(An^2 + Bn^2);
```

```
    y = y + An * cos(n*pi*t) + Bn * sin(n*pi*t);
```

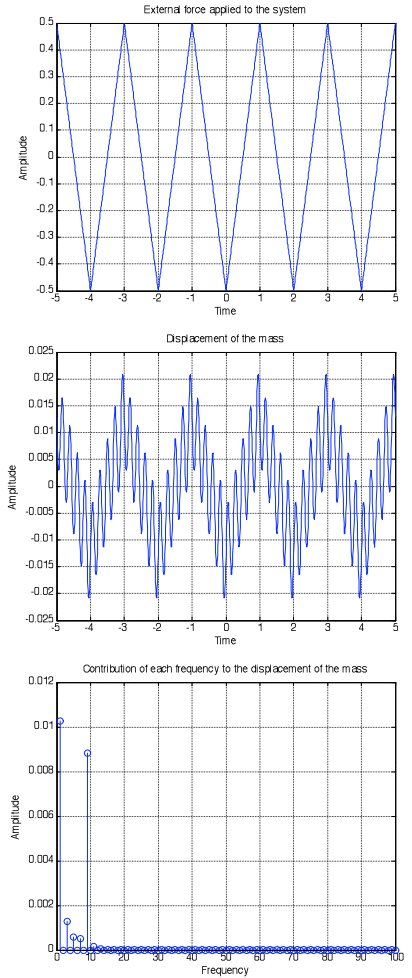
```
end
```

```
% Plot of the discrete amplitude of
each frequency
stem(C)
title('Contribution of each
frequency to the displacement of
the mass')
xlabel('Frequency')
ylabel('Amplitude')
grid on
```

```
% Plot of the external force
applied to the system
figure
plot(t,f)
title('External force applied to the
system')
xlabel('Time')
ylabel('Amplitude')
grid on
```

```
% Plot of the displacement of the
mass
figure
plot(t,y)
title('Displacement of the mass')
xlabel('Time')
ylabel('Amplitude')
grid on
```

### OUTPUT



We see on the last plot that 2 frequencies have larger contributions to the final response. These two frequencies appear clearly on the plot of the displacement : the signal is composed of a low frequency upon which a higher frequency is superposed.

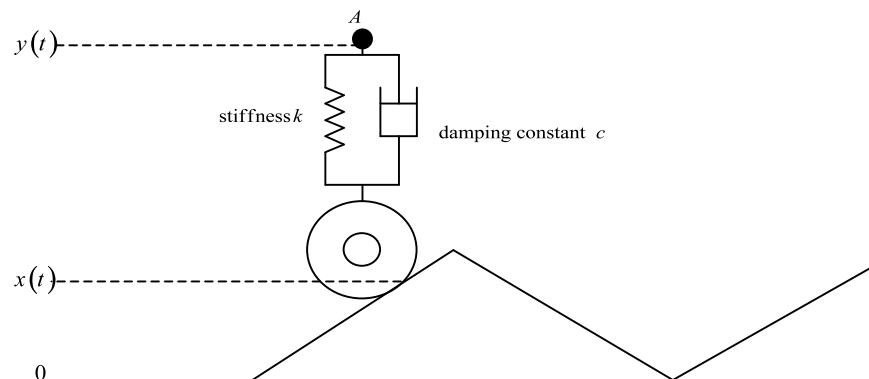


## YOUR TURN

You're in your car driving on a rough road. Suppose that the shape of the road is given by the equation :

$$x(t) = \begin{cases} \frac{t}{T} - \frac{1}{2} & 0 < t < T \\ -\frac{t}{T} - \frac{1}{2} & T < t < 2T \end{cases}$$

The mass of the car is  $m = 1.5 \times 10^3 \text{ kg}$ .



Newton's law applied to the point above the suspension gives:

$$m \frac{d^2 y}{dt^2} = -k(y - x) - c \left( \frac{dy}{dt} - \frac{dx}{dt} \right)$$

- with
- $y(t)$  displacement of the point A
  - $x(t)$  shape of the road

which can be re-written as  $m \frac{d^2 y}{dt^2} + c \frac{dy}{dt} + ky = kx + c \frac{dx}{dt}$

Assuming the solution of the form  $y(t) = \sum_{n=1}^{\infty} A_n \cos(n\pi t) + B_n \sin(n\pi t)$ , we obtain the following Fourier series representation for  $y$  :

$$y(t) = \sum_{n \text{ odd}} A_n \cos\left(\frac{n\pi t}{T}\right) + B_n \sin\left(\frac{n\pi t}{T}\right)$$

with  $A_n = \frac{C_2 C_3 - C_1 C_4}{C_1^2 + C_2^2}$  and  $B_n = \frac{C_3 - C_2 A_n}{C_1}$

$$C_1 = \frac{c\pi n}{T}, \quad C_2 = k - \frac{m\pi^2 n^2}{T^2}, \quad C_3 = -\frac{4k}{n^2 \pi^2} \text{ and } C_4 = \frac{4c}{n\pi T}$$

- a) Using the equations above, write a code which computes the displacement of point A.  
Test your code with the following cases:  $T = 5 \text{ sec}$  ,  $\Delta t = 0.05$

Case	Stiffness $k$	Damping Constant $c$
1	$10^4 \text{ m/sec}^2$	100m/sec
2	$10^5 \text{ m/sec}^2$	100m/sec
3	$10^6 \text{ m/sec}^2$	100m/sec
4	$10^5 \text{ m/sec}^2$	10m/sec
5	$10^5 \text{ m/sec}^2$	1000m/sec

In each case, plot the vertical displacement as a function of time for  $t$  between 0 and  $4T$  and the contribution of each frequency to the displacement using the **stem** command.

- b) What are the effects of  $k$  and  $c$  on the displacement of point A?

# Using the Diffusion Equation

## 8. 1D Heat Conduction Equation

The temperature of the ends of a rod, length 3, is heated such that

$$T(x,0) = \begin{cases} x & , \quad 0 \leq x \leq \frac{3}{2} \\ 3-x & , \quad \frac{3}{2} \leq x \leq 3 \end{cases} .$$

The ends of the rod are then connected to insulators to

maintain the ends at  $T(0,t) = 0$  and  $T(3,t) = 0$ . The solution to the heat conduction equation

$$\lambda \frac{\partial^2 T}{\partial x^2} = \frac{\partial T}{\partial t}, \text{ with } \lambda = 1, \text{ is: } T(x,t) = \sum_{n=1,5,9}^{\infty} \frac{4}{n^2 \pi^2} \sin\left(\frac{n\pi x}{3}\right) e^{-\frac{n^2 \pi^2}{9} t} \text{ and}$$

$$T(x,t) = \sum_{n=3,7,11}^{\infty} -\frac{4}{n^2 \pi^2} \sin\left(\frac{n\pi x}{3}\right) e^{-\frac{n^2 \pi^2}{9} t} .$$

Plot the temperature distribution of the rod at  $t = 0, 0.1, 1, 3,$  and  $10$ .



### SOLUTION

#### CODE from M-file

```
clear
close

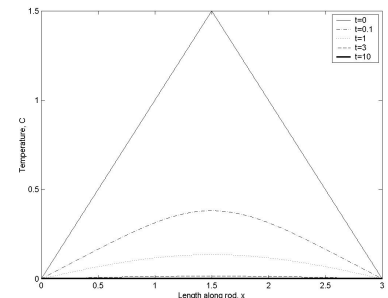
% Set maximum number of partial
sums
nmax = 11;
% Initialize domain
x = linspace(0,3,1000);

% Plot initial condition
for i = 1:length(x)
    if x(i) <= x(length(x))/2
        T(i) = x(i);
    else
        T(i) = 3-x(i);
    end
end
plot(x,T)
hold on

% Calculate temperature of rod at
various times
t=[0.1,1,3,10];
for k = 1:4
    T = 0;
    for n=1:4:nmax
        Bn = 4/(n^2*pi^2);
        T = T +
        Bn*sin(n*pi*x/3)*exp(-
        n^2*pi^2*t(k)/9);
    end
    plot(x,T,'--');
end
```

```
end
for n = 3:4:nmax
    Bn = -4/(n^2*pi^2);
    T = T +
    Bn*sin(n*pi*x/3)*exp(-
    n^2*pi^2*t(k)/9);
end
plot(x,T)
end
legend('t=0','t=0.1','t=1','t=3','t=10')
xlabel('Length along rod, x')
ylabel('Temperature, C')
```

#### OUTPUT





## YOUR TURN

A rod of length 2 has the following initial and boundary temperature conditions:

$T(x,0) = 50$ ,  $T(0,t) = 0$ , and  $T(2,t) = 100$ . The solution to the heat equation

$$\lambda \frac{\partial^2 T}{\partial x^2} + q = \frac{\partial T}{\partial t} \text{ with } \lambda = 1 \text{ is } T(x,t) = 50x + \sum_{\substack{n=\text{even} \\ n \neq 0}}^{\infty} \frac{200}{n\pi} \cdot \sin\left(\frac{n\pi x}{2}\right) \cdot e^{-\frac{n^2 \pi^2}{4} t}. \text{ Plot the}$$

temperature distribution of the rod at  $t = 0, 0.05, 0.1$ , and  $0.5$ .

## 9. 2D Laplace Equation

A rectangular plate is bounded by  $x = 0$ ,  $x = 1$ ,  $y = 0$ , and  $y = 1$  with the following boundary conditions:  $T(x,0) = 0$ ,  $T(1,y) = 0$ ,  $T(0,y) = 0$ , and  $T(x,1) = 100$ . The steady-

state temperature distribution is governed by  $\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} = 0$ . The solution is

$$T(x,y) = \sum_{n=\text{odd}}^{\infty} \frac{400}{n\pi \sinh(n\pi)} \sin(n\pi x) \sinh(n\pi y). \text{ Plot this temperature distribution.}$$



## SOLUTION

### CODE from M-file

```

clear
close

% Initialize square grid
[x,y] = meshgrid(0:0.01:1,0:0.01:1);
% Set maximum number of partial
sums
nmax = 9;

% Initialize T
T = zeros(length(x),length(y));
% Calculate steady state
temperature distribution
for n = 1:2:nmax
    T = T + 400/pi .*
    sinh(n*pi*y)/(n*sinh(n*pi)) .*
    sin(n*pi*x);
end

% Plot temperature distribution
surface(x,y,T)
shading('interp')

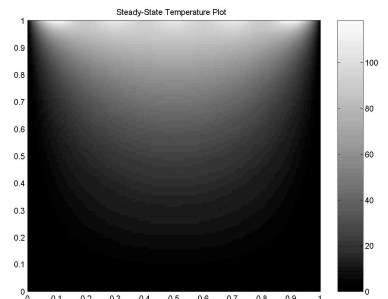
```

```

colorbar
colormap gray
title('Steady-State Temperature
Plot')

```

### OUTPUT



## YOUR TURN

A rectangular plate that is insulated on its two sides and is bounded by  $x = 0$ ,  $x = \pi$ ,

$y = 0$ , and  $y = \pi$ . The temperatures at the boundaries are  $T(x,0) = 0$ ,  $\left. \frac{\partial T}{\partial x} \right|_{x=\pi,y} = 0$ ,

$\left. \frac{\partial T}{\partial x} \right|_{x=0,y} = 0$ , and  $T(x,\pi) = 100 \sin(x)$ . The steady-state temperature distribution

governed by  $\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} = 0$ . The solution is

$$T(x, y) = \frac{200}{\pi^2} y - \sum_{n=\text{even}}^{\infty} \frac{400}{\pi} \cdot \frac{\sinh(ny)}{(n^2 - 1) \cdot \sinh(n\pi)} \cos(nx). \text{ Plot this temperature distribution.}$$

## 10. Wave motion

A string of length 4 is fixed at the two end points such that  $u(0, t) = u(4, t) = 0$ . At  $t = 0$ , the string is displaced and released from its equilibrium position such that  $u(x, 0) = 4 - (x - 2)^2$ ,

and  $\left. \frac{\partial u}{\partial t} \right|_{x, t=0} = 2$ . Given that the differential equation is  $c^2 \frac{\partial^2 u}{\partial x^2} = \frac{\partial^2 u}{\partial t^2}$ , with  $c^2 = 1$ , the

solution is  $u(x, t) = \sum_{n=\text{odd}}^{\infty} \left( \frac{128}{n^3 \pi^3} \cos\left(\frac{cn\pi t}{4}\right) + \frac{32}{cn^2 \pi^2} \sin\left(\frac{cn\pi t}{4}\right) \right) \sin\left(\frac{n\pi x}{4}\right)$ . Make a movie of the displacement of the string in the first 3 seconds.



### SOLUTION

#### CODE from M-file

```
clear

% Initialize constants and
variables
dt = 0.1;
tmax = 30;
nmax = 9;
c = 1;
nframes = tmax/dt;
fps = nframes/tmax;

% Initialize first picture at t = 0
x = linspace(0, 4, 100);
u = 4 - (x - 2).^2;
h = plot(x, u);
set(h);
axis([0, 4, -5, 5])
grid off

title('Movie of Oscilating String')
xlabel('Length along string')
```

```
ylabel('Displacement')

% Record movie
k = 1;
set(h, 'XData', x, 'YData', u)
M(k) = getframe;
for k = 2:nframes
    t = (k-1)*dt;
    u = 0;
    for n = 1:2:nmax
        u = u +
        (128/(n^3*pi^3))*cos(c*n*pi*t/4) +
        32/(c*n^2*pi^2)*sin(c*n*pi*t/4))*sin
        (n*pi*x/4);
    end
    set(h, 'XData', x, 'YData', u)
    M(k) = getframe;
end

% Play movie
movie(M, 1, fps)
```



### YOUR TURN

A square membrane bounded by  $x = 0$ ,  $x = 1$ ,  $y = 0$ , and  $y = 1$  is fixed at the

boundaries and governed by the differential equation is  $c^2 \left( \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right) = \frac{\partial^2 u}{\partial t^2}$ ,  $c^2 = 1$ .

At  $t = 0$ , the membrane is displaced and released from its equilibrium position such that

$u(x, y, 0) = \sin(2\pi x) \sin(2\pi y)$ , and  $\left. \frac{\partial u}{\partial t} \right|_{x, y, t=0} = 0$ . Make a movie of the displacement of

the membrane as a function of time  $u(x, y, t) = \cos 2\sqrt{2}\pi t \sin 2\pi x \sin 2\pi y$ .

**Note:** For the **set** function, you should add a new argument for the third dimension. The new function will look like this: **set(h, 'XData', x, 'YData', y, 'Zdata', u)**







# Boundary Value Problems

## 11. 1D Boundary Value Problem

A string of length  $L = 1\text{m}$  is fixed at both ends, such that  $y(0, t) = y(L, t) = 0$ . The motion of the string is governed by the wave equation  $c^2 \frac{\partial^2 y}{\partial x^2} = \frac{\partial^2 y}{\partial t^2}$ , with  $c = 1$ . Assume a solution of the form  $y(x, t) = F(x)G(t)$ . Substituting this solution into the wave equation and dividing by  $F(x)G(t)$ , we obtain:

$$\frac{F''(x)}{F(x)} = \frac{1}{c^2} \frac{G''(t)}{G(t)} = -k^2 \quad \text{with } k \text{ a constant.}$$

- Using a central difference scheme, discretize the differential equation for  $F$ . Set up a matrix equation for  $F$  of the form  $AF = \lambda F$ .
- Determine and plot the first three mode shapes of the string first using 10 nodes, then using 100 nodes. What's the value of  $k$  for each mode? What does it correspond to?



### SOLUTION

- The differential equation for  $x$  is  $F''(x) = -k^2 F(x)$ . It can be discretized using central difference:

$$\frac{F_{i+1} - 2F_i + F_{i-1}}{\Delta x^2} = -k^2 F_i$$

In matrix notation, using the fact that both boundary conditions are 0:

$$\frac{1}{\Delta x^2} \begin{bmatrix} -2 & 1 & 0 & \cdots & 0 \\ 1 & -2 & 1 & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & 1 & -2 & 1 \\ 0 & \cdots & 0 & 1 & -2 \end{bmatrix} \begin{bmatrix} F_1 \\ F_2 \\ \vdots \\ F_{N-3} \\ F_{N-2} \end{bmatrix} = -k^2 \begin{bmatrix} F_1 \\ F_2 \\ \vdots \\ F_{N-3} \\ F_{N-2} \end{bmatrix} \quad \text{of the form } AF = \lambda F,$$

which corresponds to an eigenvalue problem.

- The eigenvalues and eigenvectors of matrix  $A$  are obtained using MATLAB and the `eig` command.

#### CODE from M-file

```
clear
close
```

```
% Boundaries
a = 0;
```

```
b = 1;
% Number of points
N = 10;
% Location of the points
x = linspace(a,b,N);
% Distance between 2 points
h = (b-a)/(N-1);
```

```

% Creation of matrix A
A = zeros(N-2,N-2);
A(1,1) = -2/h^2;
A(1,2) = 1/h^2;

for n = 2:N-3
    A(n,n-1) = 1/h^2;
    A(n,n) = -2/h^2;
    A(n,n+1) = 1/h^2;
end

A(N-2,N-2) = -2/h^2;
A(N-2,N-3) = 1/h^2;

% Computation of the eigenvalues
and eigenvectors
[V,d]=eig(A);

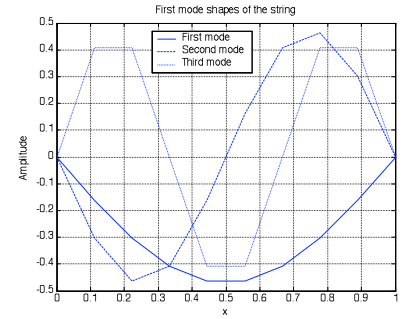
% Mode shapes
plot(x,[0;V(:,N-2);0],'-')
hold on
plot(x,[0;V(:,N-3);0],'-')
plot(x,[0;V(:,N-4);0],'-')

legend('First mode','Second mode','Third mode')
title('First mode shapes of the string')
xlabel('x')
ylabel('Amplitude')
grid on

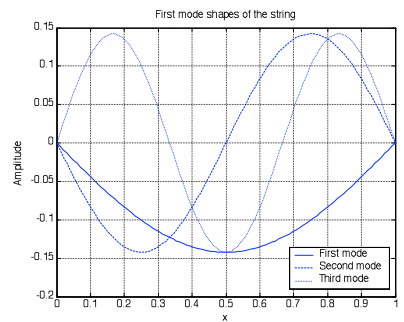
```

#### OUTPUT

**N = 10 nodes**



**N = 100 nodes**



MATLAB returns the eigenvalues sorted by increasing value, and in this case, they are all negative. What we call first modes corresponds to the smallest eigenvalues in magnitude. They correspond to the last columns of the matrix giving the eigenvectors.

The eigenvalues are  $\lambda = -k^2$ . For the first three modes, we have :

$$N = 10 \text{ nodes} \quad \begin{cases} \lambda_1 = -9.7698 \Rightarrow k_1 = 3.125 \\ \lambda_2 = -37.9008 \Rightarrow k_2 = 6.156 \\ \lambda_3 = -81.0000 \Rightarrow k_3 = 9 \end{cases}$$

$$N = 100 \text{ nodes} \quad \begin{cases} \lambda_1 = -9.8688 \Rightarrow k_1 = 3.141 \\ \lambda_2 = -39.465 \Rightarrow k_2 = 6.282 \\ \lambda_3 = -88.759 \Rightarrow k_3 = 9.421 \end{cases}$$

$$\text{These results have to be compared to the theoretical values : } \begin{cases} k_{1\text{exact}} = \frac{\pi}{L} = 3.141 \\ k_{2\text{exact}} = \frac{2\pi}{L} = 6.283 \\ k_{3\text{exact}} = \frac{3\pi}{L} = 9.425 \end{cases}$$

$k$  represents the angular frequency in radian per second with which the modes oscillate in space. The more points there are, the closest from the theoretical results the numerical values are.



## YOUR TURN

The same kind of matrix is used to solve for example the 1D steady heat equation :

$$\lambda^2 \frac{\partial^2 T}{\partial x^2} + q = 0.$$

A rod of length  $L = 1m$  is maintained at  $T = 0^\circ C$  at  $x = 0$  and is insulated at the other end ( $x = L$ ). This rod is heated uniformly with a source  $q = 1K/sec$ .

$$\lambda^2 \frac{\partial^2 T}{\partial x^2} + q = 0 \quad \text{with the following boundary conditions : } T(0) = 0 \quad \text{and} \quad \frac{\partial T}{\partial x}(L) = 0$$

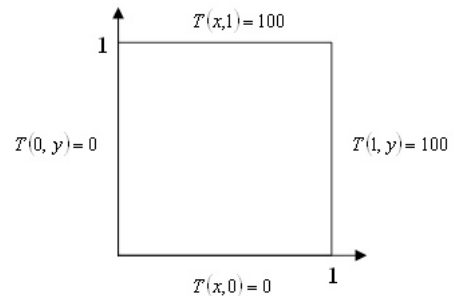
Assume  $\lambda = 1$ .

- Discretize the heat equation using a central difference scheme. Use the off-centred difference scheme to enforce the zero-gradient boundary condition at  $x = L$ . Set up the matrix equation of the form  $AT = b$ .
- Solve this matrix equation using the **inv** command. Use 100 points. Plot the temperature  $T$  as a function of  $x$ . Add labels and a title.
- Find the analytical solution and plot it on the same figure.

## 12. 2D Laplace Equation using Jacobi and Gauss-Seidel

A square cooling plate of length 1 is maintained at  $0^\circ C$ . When the machine turns on, two sides are heated to  $100^\circ C$ , as shown in the figure to the right. Using 10 nodes and 30 nodes, use the Jacobi iteration method to plot the steady-state temperature after the machine turns on, to within a 1% error. The iteration scheme is

$$u_{i,j}^{n+1} = \frac{u_{i-1,j}^n + u_{i+1,j}^n + u_{i,j-1}^n + u_{i,j+1}^n}{4}.$$



## SOLUTION

### CODE from M-file for 10 nodes

```

clear
close

% Initialize number of nodes
M = 10;
N = 10;
% Initialize function
u = zeros(M,N);
% Initialize boundary conditions
u(M,:) = 100;
u(:,N) = 100;
% Calculate temperature
distribution
flag = 1;
while flag == 1
    flag = 0;
    uold = u;
    for i = 2:M-1
        for j = 2:N-1

```

```

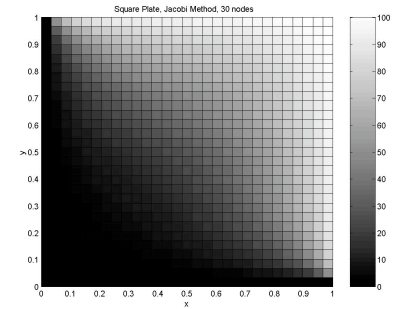
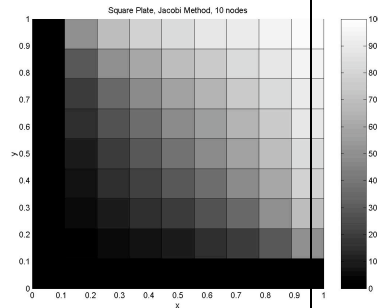
            u(i,j) = (uold(i-1,j) +
            uold(i+1,j) + uold(i,j-1) +
            uold(i,j+1))/4;
            error = abs((u(i,j)-
            uold(i,j))/uold(i,j));
            if error > 0.01
                flag = 1;
            end
        end
    end
end
% Plot temperature distribution
[x,y] = meshgrid(0:1/(M-1):1 ,
0:1/(N-1):1);
surface(x,y,u)
colorbar
colormap gray
title('Square Plate, Jacobi Method,
10 nodes')
xlabel('x')
ylabel('y')

```

#### CODE from M-file for 30 nodes

```
% same code with 30 nodes  
instead of 10
```

#### OUTPUT



*Note:* The **shading** function was not used in this instance, so that the node distribution can be seen. If you use the **shading** function, the two plots will look the same.



### YOUR TURN

Using the same criteria as the example problem, adjust the code for the Gauss-Seidel iteration method using 30 nodes. The iteration scheme is

$$u_{i,j}^{n+1} = \frac{u_{i-1,j}^{n+1} + u_{i+1,j}^n + u_{i,j-1}^{n+1} + u_{i,j+1}^n}{4}.$$

*Note:* The difference in efficiency may not be noticeable, but the Gauss-Seidel method will help the calculations reach the steady-state solution faster than with the Jacobi iteration.

## 13. Time-dependent heat equation

Heat generated from an electric wire is defined by the time-dependent heat equation:

$$\lambda^2 \frac{\partial^2 u}{\partial x^2} + q = \frac{\partial u}{\partial t} \text{ with } u(0,t) = u(L,t) = 0, \text{ and } u(x,0) = 0.$$

Use the discretization scheme  $u_i^{n+1} = u_i^n + \frac{\lambda^2 dt}{h^2} (u_{i-1}^n - 2u_i^n + u_{i+1}^n) + q \cdot dt$  with  $N = 10$ ,  $L = 1$ ,  $\lambda = 1$ , and  $q = 1$  to find the temperature distribution at  $t = 0, 0.01, 0.05, 0.1, 1$ . Take  $dt = 0.005$ .

The analytical solution for this problem through Fourier series is :

$$T(x,t) = -\frac{qx^2}{2} + \frac{qLx}{2} + \sum_{n=odd}^{\infty} \frac{4L^2 q}{n^3 \pi^3} \sin \frac{n\pi x}{L} e^{-\frac{n^2 \pi^2 t}{L^2}}$$

Plot the analytical solution at the same time steps using the first ten partial sums, and compare the two graphs.



### SOLUTION

#### CODE from M-file

```
clear  
close all
```

```
% Initialize number of nodes and  
constants  
N = 10;
```

```

lambda = 1;
L = 1;
q = 1;
h = L/(N-1);
dt = 0.005;

% Initialize domain
x = linspace(0,L,N);

% Initialize and plot heat
distribution at t = 0
u = zeros(1,N);
plot(x,u,'k')
hold on

% Calculate heat at specified
times
t = [0.02,0.05,0.1,1];
jmax = t/dt;
k = 1;
u = zeros(1,N);

for j = 1:jmax(4)
    uold = u;
    for i=2:N-1
        u(i) = uold(i) +
        lambda*dt/h^2*(uold(i-1)-
        2*uold(i)+uold(i+1)) + q*dt;
    end
    if j == jmax(k)
        plot(x,u)
        k = k+1;
    end
end

% Calculate with analytical
solution
nmax = 19;
L = 1;
q = 1;
x = linspace(0,L,1000);

% Calculate and plot analytical
temperature distribution at times, t

```

```

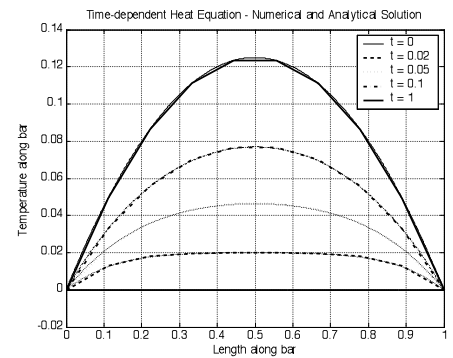
T = zeros(1000,1);
plot(x,T,'r')
hold on

t = [0.02,0.05,0.1,1];
for k = 1:length(t)
    T = -q*x.^2/2 + q*L*x/2;
    for n = 1:2:nmax
        T = T - 4*L^2*q/(n^3*pi^3) *
        sin(n*pi*x/L) * exp(-
        n^2*pi^2*t(k)/L^2);
    end
    plot(x,T,'r')
end

title('Time-dependent Heat
Equation - Numerical and
Analytical Solution')
xlabel('Length along bar')
ylabel('Temperature along bar')
legend('t = 0','t = 0.02','t = 0.05','t =
0.1','t = 1')
grid on

```

#### OUTPUT



## YOUR TURN

A rod of length 1, whose initial temperature profile is  $u(x,0) = 100 \sin \pi x$ , becomes attached on one end to a cooling reservoir, maintained at  $0^\circ\text{C}$  while the other end is insulated, which gives the following boundary condition :  $\left. \frac{dT}{dx} \right|_{x=L} = 0$ .

For the time-dependent heat equation,  $\lambda^2 \frac{\partial^2 u}{\partial x^2} + q = \frac{\partial u}{\partial t}$ , for the rod using 30 nodes with  $q = 1$ , the iteration scheme is :

$$u_i^{n+1} = u_i^n + \frac{L^2 dt}{h^2} (u_{i-1}^n - 2u_i^n + u_{i+1}^n) + q \cdot dt \text{ and } u_N^{n+1} = \frac{1}{3} (4u_{N-1}^n - u_{N-2}^n) \text{ at the insulated end.}$$

Plot the temperature profile of the rod at  $t = 0, 0.01, 0.05, 0.2, 0.5$ . Take  $dt = 0.0005$

## 14. Time-dependent heat equation – Implicit Scheme

Heat generated from an electric wire is defined by the time-dependent heat equation:

$$\lambda^2 \frac{\partial^2 u}{\partial x^2} + q = \frac{\partial u}{\partial t} \text{ with } u(0,t) = u(L,t) = 0, \text{ and } u(x,0) = 0.$$

Use the fully implicit discretization scheme with  $N = 10$ ,  $L = 1$ ,  $\lambda = 1$  and  $q = 1$ , and plot the temperature distribution at  $t = 0, 0.01, 0.05, 0.1, 1$ . Take  $dt = 0.01$

Note: The fully implicit distribution scheme is  $\lambda^2 \frac{u_{i-1}^{n+1} - 2u_i^{n+1} + u_{i+1}^{n+1}}{\Delta x^2} + 1 = \frac{u_i^{n+1} - u_i^n}{\Delta t}$ .



### SOLUTION

#### CODE from M-file

```
clear
close

% Initialize number of nodes and
constants
N = 10;
L = 1;
lambda = 1;
h = L/(N-1);
q = 1;

dt = 0.01;

% Initialize A
r = lambda^2*dt/(h^2);
A = zeros(N-2,N-2);
A(1,1) = 1+2*r;
A(1,2) = -r;
for i=2:N-3
    A(i,i-1) = -r;
    A(i,i) = 1+2*r;
    A(i,i+1) = -r;
end
A(N-2,N-3) = -r;
A(N-2,N-2) = 1+2*r;

% Initialize b
b = zeros(N-2,1);
for i = 1:N-2
    b(i,1) = q*dt;
end

% Initialize domain
x = linspace(0,L,N)';

% Initialize u at t = 0
u = zeros(N,1);
plot(x,u,'k')
hold on
```

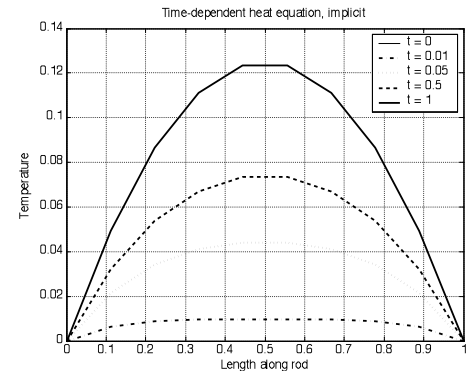
```
% Calculate and plot temperature
distribution at times, t
t = [0.01,0.05,0.1 1];
jmax = t/dt;
k = 1;
```

```
for j = 1:jmax(4)
    u(2:N-1) = inv(A)*(u(2:N-1)+b);
```

```
    if j == jmax(k)
        plot(x,u,'k')
        k = k+1;
    end
end
```

```
title('Time-dependent heat
equation, implicit')
xlabel('Length along rod')
ylabel('Temperature')
legend('t = 0','t = 0.01','t = 0.05','t =
0.5','t = 1')
```

#### OUTPUT



### YOUR TURN

A rod of length 1, whose initial temperature profile is  $u(x,0) = 100 \sin \pi x$ , becomes attached on one end to a cooling reservoir, maintained at  $0^\circ\text{C}$  the other end is insulated, which gives

the following boundary condition :  $\left. \frac{dT}{dx} \right|_{x=L} = 0$ .

For the time-dependent heat equation,  $\lambda^2 \frac{\partial^2 u}{\partial x^2} + q = \frac{\partial u}{\partial t}$ , using 30 nodes with  $q = 1$ , use the fully-implicit iteration scheme to plot the temperature profile of the rod at  $t = 0, 0.01, 0.05, 0.2, 0.5$ . Take  $dt = 0.01$