மனோன்மணியம் சுந்தரனார் பல்கலைக்கழகம்

MANONMANIAM SUNDARANAR UNIVERSITY

TIRUNELVELI-627 012

தொலைநிலை தொடர் கல்வி இயக்ககம்

DIRECTORATE OF DISTANCE AND CONTINUING EDUCATION

B.Sc. MATHEMATICS

III YEAR

MATLAB

Sub. Code: JNMA61

Prepared by

Dr.K. R. SELVAKUMAR

ASSISTANT PROFESSOR

DEPARTMENT OF MATHEMATICS

M.S.UNIVERSITY

TIRUNELVELI-12.

Chapter l

Starting with MATLAB

This chapter begins by describing the characteristics and purpose of the different windows in MATLAB. Next, the Command Window is introduced in detail. The chapter shows how to use MATLAB for arithmetic operations with scalars in much to the way that a calculator is used. This includes the use of elementary math functions with scalars. The chapter then shows how to define scalar vari- ables (the assigmnent operator) and how to use these variables in arithmetic calcu- lations. The last section in the chapter introduces script files. It shows how to write, save, and execute simple MATLAB programs.

1.1 STARTING MATLAB, MATLAB WINDOWS

It is assumed that the software is installed on the computer, and that the user can start the program. Once the program starts, the MATLAB desktop window opens with the default layout, Figure 1-1. The layout has a Toolstrip at the top, the Current Folder Toolbar below it, and four windows underneath. At the top of the Toolstrip there are three tabs: HOME, PLOTS, and APPS. Clicking on the tabs changes the icons in the Toolstrip. Commonly, MATLAB is used with the HOME tab selected. The associated icons are used for executing various commands, as explained later in this chapter. The PLOTS tab can be used to create plots, as explained in Chapter 5 (Section 5.12), and the APPS tab can be used for opening additional applications and Toolboxes of MATLAB.

The default layout

The default layout (Figure 1-1) consists of the following four windows that are displayed under the Toolstrip: the Command Window (larger window at the cen- ter), the Curent Folder Window (on the left) and the Workspace and Command History windows (on the right). A list of several MATLAB windows and their purposes is given in Table 1-1.

Four of the windows-the Command Window, the Figure Window, the Editor Window, and the Help Window-are used extensively throughout the book and

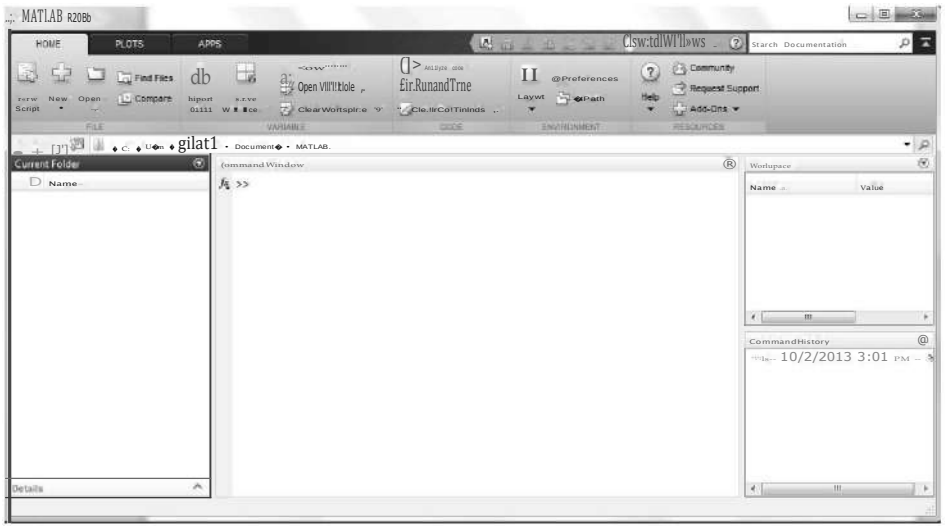


Figure 1-1: The default view ofMATLAB desktop.

are briefly described on the following pages. More detailed descriptions are included in the chapters where they are used. The Command History Window, Current Folder Window, and the Workspace Window are described in Sections 1.2, 1.8.4, and 4.1, respectively.

Command Window: The Command Window is MATLAB 's main window and opens when MATLAB is started. It is convenient to have the Command Window as the only visible window. This can be done either by closing all the other win- dows, or by selecting Command Window Only in the menu that opens when the Layout icon on the Toolstrip is selected. To close a window, click on the pull- down menu at the top right-hand side of the window and then select Close. Work- ing in the Command Window is described in detail in Section 1.2.

Table 1-1: MATLAB windows

Window	Purpose
Command Window	Main window, enters variables, runs programs.
Figure Window	Contains output from graphic commands.
Editor Window	Creates and debugs script and function files.
Help Window	Provides help information.
Command History Window	Logs commands entered in the Command Window.

Table 1-1: MATLAB windows

Window	Purpose
Workspace Window	Provides information about the variables that are stored.
Current Folder Window	Shows the files in the current folder.

Figure Window: The Figure Window opens automatically when graphics com- mands are executed, and contains graphs created by these commands. An example of a Figure Window is shown in Figure 1-2. A more detailed description of this window is given in Chapter 5.

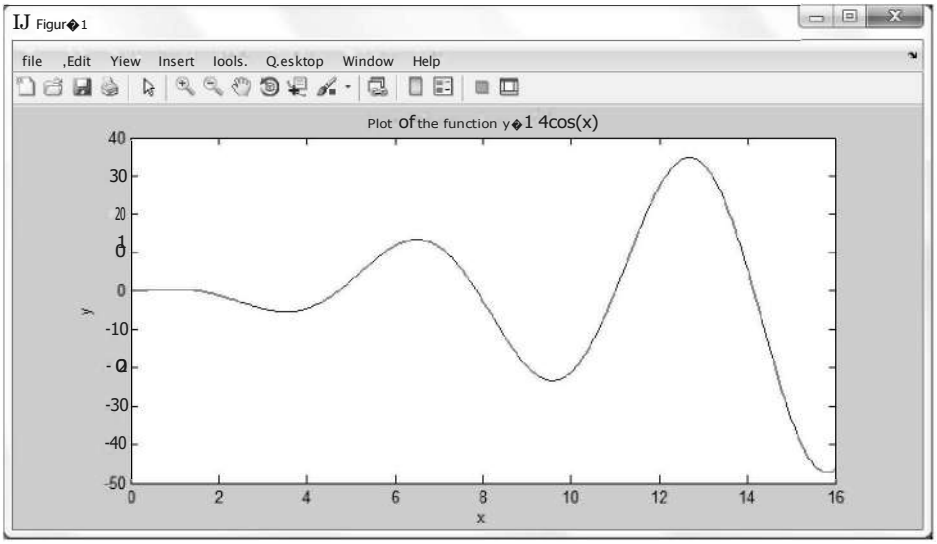


Figure 1-2: Example of a Figure Window.

Editor Window: The Editor Window is used for writing and editing programs. This window is opened by clicking on the New Script icon in the Toolstrip, or by clicking on the New icon and then selecting Script from the menu that opens. An example of an Editor Window is shown in Figure 1-3.More details on the Editor Window are given in Section 1.8.2, where it is used for writing script files, and in Chapter 7, where it is used to write function files.

Help Window: The Help Window contains help information. This window can be opened from the Help icon in the Toolstrip of the Command Window or the toolbar of any MATLAB window. The Help Window is interactive and can be used to obtain information on any feature ofMATLAB. Figure 1-4 shows an open Help Window.

When MATLAB is started for the first time, the screen looks like that shown in Figure 1-1. For most beginners it is probably more convenient to close all the

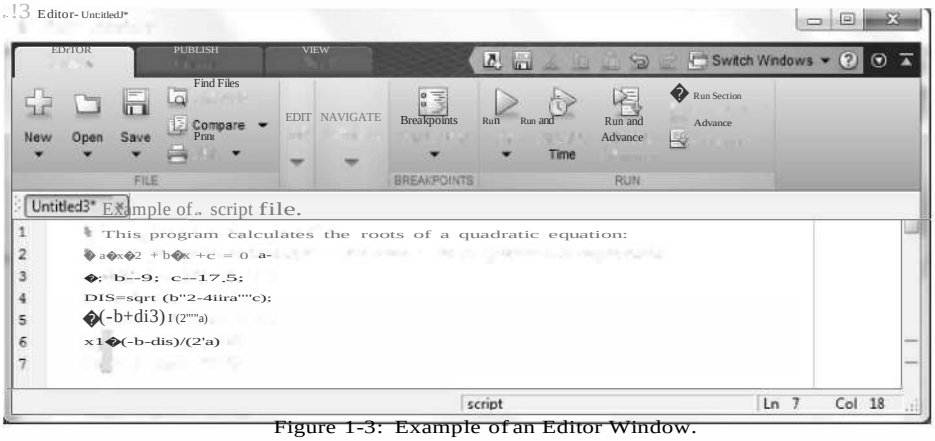


Figure 1-3: Example of an Editor Window.

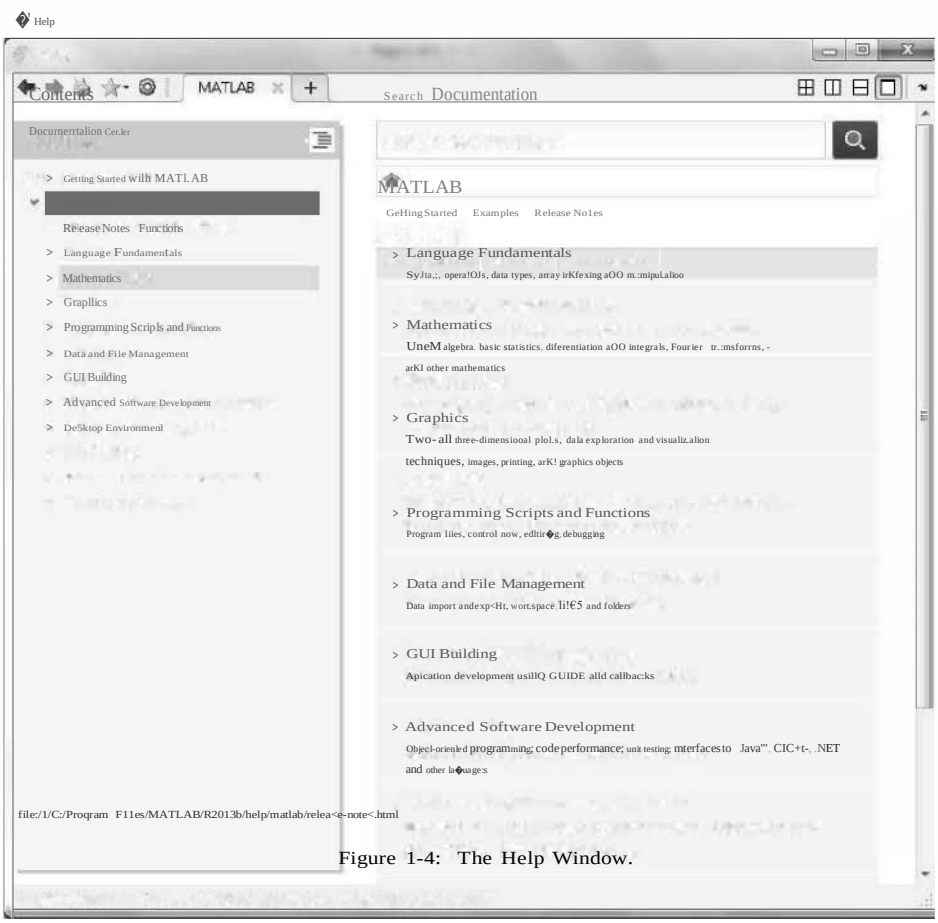


Figure 1-4: The Help Window.

windows except the Command Window. The closed windows can be reopened by selecting them from the layout icon in the Toolstrip. The windows shown in Fig- ure 1-1 can be displayed by clicking on the layout icon and selecting Default in the menu that opens. The various windows in Figure 1-1 are docked to the desk-
top. A window can be undocked (become a separate, independent window) by dragging it out. An independent window can be redocked by clicking on the pull- down menu at the top right-hand side of the window and then selecting Dock.

1.2 WORKING IN THE COMMAND WINDOW

The Command Window is MATLAB's main window and can be used for execut- ing commands, opening other windows, running programs written by the user, and managing the software. An example of the Command Window, with several sim-
ple commands that will be explained later in this chapter, is shown in Figure 1-5.

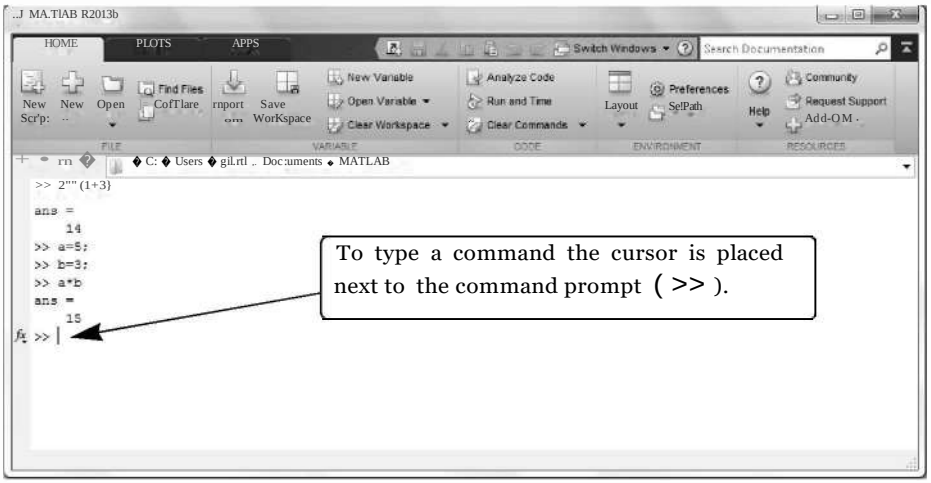


Figure 1-5: The Command Window.

Notes for working in the Command Window:

- To type a command, the cursor must be placed next to the command prompt (>>).
- Once a command is typed and the Enter key is pressed, the command is executed. However, only the last command is executed. Everything executed previously (that might be still displayed) is unchanged.
- Several commands can be typed in the same line. This is done by typing a comma between the commands. When the Enter key is pressed, the commands are exe- cuted in order from left to right.
- It is not possible to go back to a previous line that is displayed in the Command Window, make a correction, and then re-execute the command.

- A previously typed command can be recalled to the command prompt with the up- arrow key (t). When the command is displayed at the command prompt, it can be modified if needed and then executed. The down-arow key (.t) can be used to move down the list of previously typed commands.
- If a command is too long to fit in one line, it can be continued to the next line by typing three periods ... (called an ellipsis) and pressing the Enter key. The con- tinuation of the command is then typed in the new line. The command can con- tinue line after line up to a total of 4,096 characters.

The semicolon (;):

When a command is typed in the Command Window and the Enter key is pressed, the command is executed. Any output that the command generates is dis- played in the Command Window. If a semicolon (;) is typed at the end of a com- mand, the output of the command is not displayed. Typing a semicolon is useful when the result is obvious or known, or when the output is very large.

If several commands are typed in the same line, the output from any of the commands will not be displayed if a semicolon instead of a comma is typed between the commands.

Typing%:

When the symbol% (percent) is typed at the beginning of a line, the line is desig- nated as a comment. This means that when the Enter key is pressed the line is not executed. The% character followed by text (comment) can also be typed after a command (in the same line). This has no effect on the execution of the command.

Usually there is no need for comments in the Command Window. Comments, however, are frequently used in a program to add descriptions or to explain the program (see Chapters 4 and 6).

The clc command:

The clc command (type clc and press Enter) clears the Command Window. After typing in the Command Window for a while, the display may become very long. Once the clc command is executed, a clear window is displayed. The com- mand does not change anything that was done before. For example, if some vari- ables were defined previously (see Section 1.6), they still exist and can be used. The up-arrow key can also be used to recall commands that were typed before.

The Command History Window:

The Command History Window lists the commands that have been entered in the Command Window. This includes commands from previous sessions. A com- mand in the Command History Window can be used again in the Command Win- dow. By double-clicking on the command, the command is reentered in the Command Window and executed. It is also possible to drag the command to the Command Window, make changes if needed, and then execute it. The list in the Command History Window can be cleared by selecting the lines to be deleted and

then right-clicking the mouse and selecting Delete Selection. The whole history can be deleted by right-clicking the mouse and selecting choose Clear Command History in the menu that opens.

1.3 *ARITHMETIC OPERATIONS WITH SCALARS*

In this chapter we discuss only arithmetic operations with scalars, which are num- bers. As will be explained later in the chapter, numbers can be used in arithmetic calculations directly (as with a calculator) or they can be assigned to variables, which can subsequently be used in calculations. The symbols of arithmetic opera- tions are:

Operation	Symbol	Example
Subtraction		5-3
Multiplication	*	5*3
Right division	I	5/3
Left division	\	5\3=3/5
Exponentiation	A	5 A 3 (means $5^3 = 125$)

It should be pointed out here that all the symbols except the left division are the same as in most calculators. For scalars, the left division is the inverse of the right division. The left division, however, is mostly used for operations with arays, which are discussed in Chapter 3.

1.3.1 *Order ofPrecedence*

MATLAB executes the calculations according to the order of precedence dis- played below. This order is the same as used in most calculators.

Precedence	Mathematical Operation
First	Parentheses. For nested parentheses, the innermost are executed ftrst.
Second	Exponentiation.
Third	Multiplication, division (equal precedence). Addition
Fourth	and subtraction.

In an expression that has several operations, higher-precedence operations are executed before lower-precedence operations. If two or more operations have the same precedence, the expression is executed from left to right. As illustrated in the next section, parentheses can be used to change the order of calculations.

1.3.2 Using MATLAB as a Calculator

The simplest way to use MATLAB is as a calculator. This is done in the Command Window by typing a mathematical expression and pressing the Enter key. MATLAB calculates the expression and responds by displaying ans = followed by the numerical result of the expression in the next line. This is demonstrated in Tutorial 1-1.

Tutorial1-1: Using MATLAB as a calculator.

```
» 7+8/2
ans =
    11
» (7+8)/2
ans =
    7.5000
» 4+5/3+2
ans =
    7.6667
» 5..3/2
ans =
   62.5000
» 27..(1/3)+32..0.2
ans =
     5
» 27..1/3+32..0.2
ans =
    11

» 0.7854-(0.7854)..3/(1*2*3)+0.785..5/(1*2*3*4*5) . . .
-(0.785).. 7/(1*2*3*4*5*6*7)

ans =
    0.7071
»
```

Type and press Enter.

8/2 is executed first.

Type and press Enter.

7+8 is executed first.

5/3 is executed first.

5A3 is executed first, /2 is executed next.

1/3 is executed frrst, 27A(1/3) and 32A0.2 are executed next, and + is executed last.

27A1 and 32A0.2 are executed frrst, /3 is executed next, and + is executed last.

Type three periods ... (and press Enter) to continue the expression on the next line.

The last expression is the first four terms of the Taylor series for sin(1t/4).

1.4 DISPLAY FORMATS

The user can control the format in which MATLAB displays output on the screen. In Tutorial 1-1, the output format is fixed-point with four decimal digits (called short), which is the default format for numerical values. The format can be

changed with the format command. Once the format command is entered, all the output that follows is displayed in the specified format. Several of the avail- able formats are listed and described in Table 1-2.

MATLAB has several other formats for displaying numbers. Details of these formats can be obtained by typing help format in the Command Window. The format in which numbers are displayed does not affect how MATLAB computes and saves numbers.

Table 1-2: Display formats

Command	Description	Example
format short	Fixed-point with 4 decimal digits for: 0.001 :5: number :5: 1000 Otherwise display format short e.	>> 290/7 ans = 41.4286
format long	Fixed-point with 15 decimal digits for: 0.001 :5: number :5: 100 Otherwise display format long e.	>> 290/7 ans = 41.428571428571431
format short e	Scientific notation with 4 decimal digits.	>> 290/7 ans = 4.1429e+001
format long e	Scientific notation with 15 decimal digits.	>> 290/7 ans = 4.142857142857143e+001
format short g	Best of 5-digit fixed or floating point.	>> 290/7 ans = 41.429
format long g	Best of 15-digit fixed or floating point.	>> 290/7 ans = 41.4285714285714
format bank	Two decimal digits.	>> 290/7 ans = 41.43
format compact	Eliminates empty lines to allow more lines with information displayed on the screen.	
format loose	Adds empty lines (opposite of compact).	

1.5 ELEMENTARY MATH BUILT-IN FUNCTIONS

In addition to basic arithmetic operations, expressions in MATLAB can include functions. MATLAB has a very large library of built-in functions. A function has a name and an argument in parentheses. For example, the function that calculates the square root of a number is sqrt(x). Its name is sqrt, and the argument is x. When the function is used, the argument can be a number, a variable that has been assigned a numerical value (explained in Section 1.6), or a computable expression that can be made up of numbers and/or variables. Functions can also be included in arguments, as well as in expressions. Tutorial1-2 shows examples of using the function sqrt(x) when MATLAB is used as a calculator with sca- lars.

Tutorial1-2: Using the sqrt built-in function.

```
>> sqrt(64)                          Argument is a number.
ans =
     8

>> sqrt(50+14*3)                     Argument is an expression.
ans =
    9.5917

>> sqrt(54+9*sqrt(100))              Argument includes a function.
ans =
    12

>> (15+600/4)/sqrt(121)              Function is included in an expression.
ans =
    15
>>
```

Some commonly used elementary MATLAB mathematical built-in functions are given in Tables 1-3 through1-5. A complete list of functions organized by cat- egory can be found in the Help Window.

Table 1-3: Elementary math functions

Function	Description	Example
sqrt(x)	Square root.	>> sqrt(81) ans = 9
nthroot(x,n)	Real nth root of a real number x. (If x is negative n must be an odd integer.)	>> nthroot(80,5) ans = 2.4022
exp(x)	Exponential (eX) .	>> exp(5) ans = 148.4132

Table 1-3: Elementary math functions (Continued)

Function	Description	Example

`abs(x)`	Absolute value.	`>> abs(-24)`
		`ans =`
		`24`

`log(x)`	Natural logarithm. Base e logarithm (In).	`>> log(1000)`
		`ans =`
		`6.9078`

`log10(x)`	Base 10 logarithm.	`>> log10(1000)`
		`ans =`
		`3.0000`

`factorial(x)`	The factorial function x! (x must be a positive integer.)	`>> factorial(5) ans =` `120`

Table 1-4: Trigonometric math functions

Function	Description	Example

`sin(x)`
`sind(x)`
	Sine of angle x (x in radians). Sine of angle x (x in degrees).	`>> sin(pi/6)`
		`ans =`
		`0.5000`

`cos(x)`
`cosd(x)`
	Cosine of angle x (x in radians).
	Cosine of angle x (x in degrees).
		`>> cosd(30)`
		`ans =`
		`0.8660`

`tan(x)`
`tand(x)`
	Tangent of angle x (x in radians). Tangent of angle x (x in degrees).	`>> tan(pi/6)`
		`ans =`
		`0.5774`

`cot(x)`
`cotd(x)`
	Cotangent of angle x (x in radians).
	Cotangent of angle x (x in degrees).
		`>> cotd(30)`
		`ans =`
		`1.7321`

The inverse trigonometric functions are `asin(x)`, `acos(x)`, `atan(x)`, `acot(x)` for the angle in radians; and `asind(x)`, `acosd(x)`, `atand(x)`, `acotd(x)` for the angle in degrees. The hyperbolic trigonometric functions are `sinh(x)`, `cosh(x)`, `tanh(x)`, and `coth(x)`. Table 1-4 uses `pi`, which is equal to 1t (see Section 1.6.3).

Table 1-5: Rounding functions

Function	Description	Example
`round(x)`	Round to the nearest integer.	`>> round(17/5) ans =` `3`

`fix(x)`	Round toward zero.	`>> fix(13/5)`
		`ans =`
		`2`

Table 1-5: Rounding functions (Continued)

Function	Description	Example
ceil(x)	Round toward infinity.	>> ceil(11/5) ans = 3
floor(x)	Round toward minus infinity.	>> floor(-9/4) ans = -3
rem(x,y)	Returns the remainder after x is divided by y.	>> rem(13,5) ans = 3
sign(x)	Signum function. Returns 1 if x>0 ,-1 ifx<O,andOif x = 0.	>> sign(5) ans = 1

1.6 DEFINING SCALAR VARIABLES

A variable is a name made of a letter or a combination of several letters (and dig- its) that is assigned a numerical value. Once a variable is assigned a numerical value,it can be used in mathematical expressions,in functions,and in any MAT- LAB statements and commands. A variable is actually a name of a memory loca- tion. When a new variable is defined,MATLAB allocates an appropriate memory space where the variable's assignment is stored. When the variable is used the stored data is used. If the variable is assigned a new value the content of the memory location is replaced. (In Chapter 1 we consider only variables that are assigned numerical values that are scalars. Assigning and addressing variables
that are arrays is discussed in Chapter 2.)

1.6.1 The Assignment Operator

In MATLAB the = sign is called the assignment operator. The assignment opera- tor assigns a value to a variable.

```
Variable_name = A numerical value,or a computable expression
```

- The left-hand side of the assignment operator can include only one variable name. The right-hand side can be a number,or a computable expression that can include numbers and/or variables that were previously assigned numerical values. When the Enter key is pressed the numerical value of the right-hand side is assigned to the variable, and MATLAB displays the variable and its assigned value in the next two lines.

  The following shows how the assignment operator works.

```
>> X=15
X =
     15

>> x=3*x-12
X =
     33
>>
```

The number 15 is assigned to the variable x.

MATLAB displays the variable name and its assigned value.

A new value is assigned to x. The new value is 3 times the previous value ofx minus 12.

The last statement (x . 3x -12) illustrates the difference between the assignment operator and the equal sign. If in this statement the _ sign meant equal, the value of X would be 6 (solving the equation for x).

The use of previously defmed variables to defme a new variable is demon- strated next.

```
>> a=12
a =
     12
>> B=4
B =
      4
>> C=(a-B)+40-a/B*10
c =
     18
```

Assign 12 to a.

Assign 4 to B.

Assign the value of the expres- sion on the right-hand side to the variable C.

- If a semicolon is typed at the end of the command, then when the Enter key is pressed, MATLAB does not display the variable with its assigned value (the vari- able still exists and is stored in memory).
- If a variable already exists, typing the variable's name and pressing the Enter key will display the variable and its value in the next two lines.

As an example, the last demonstration is repeated below using semicolons.

```
>> a=12;
>> B=4;
>> C=(a-B)+40-a/B*10;
>> c
c =
     18
```

The variables a, B, and C are defined but are not displayed, since a semicolon is typed at the end of each statement.

The value of the variable C is displayed by typing the name of the variable.

- Several assignments can be typed in the same line. The assignments must be sepa- rated with a comma (spaces can be added after the comma). When the Enter key is pressed, the assignments are executed from left to right and the variables and

their assignments are displayed. A variable is not displayed if a semicolon is typed instead of a comma. For example, the assignments of the variables a, B, and C above can all be done in the same line.

```
>> a=12, B=4; C=(a-B)+40-a/B*10
a =
     12
C =
     18
```

The variable B is not displayed because a semicolon is typed at the end of the assignment.

- A variable that already exists can be reassigned a new value. For example:

```
>> ABB=72;
>> ABB=9;
>> ABB
ABB =
      9
>>
```

A value of72 is assigned to the variable ABE.

A new value of9 is assigned to the variable ABE.

The curent value of the variable is displayed when the name of the variable is typed and the Enter key is pressed.

- Once a variable is defined it can be used as an argument in functions. For exam- ple:

```
>> X=0.75;
>> E=sin(x)A2+cos(x)A2
E =
      1
>>
```

1.6.2 Rules About Variable Names

A variable can be named according to the following rules:

- Must begin with a letter.
- Can be up to 63 characters long.
- Can contain letters, digits, and the underscore character.
- Cannot contain punctuation characters (e.g., period, comma, semicolon).
- MATLAB is case-sensitive: it distinguishes between uppercase and lowercase let- ters. For example, AA, Aa, aA, and aa are the names of four different variables.
- No spaces are allowed between characters (use the underscore where a space is desired).
- Avoid using the name of a built-in function for a variable (i.e., avoid using cos, sin, exp, sqrt, etc.). Once a function name is used to for a variable name, the function cannot be used.

1.6.3 *Predefined Variables and Keywords*

There are 20 words, called keywords, that are reserved by MATLAB for various purposes and cannot be used as variable names. These words are:

break case catch classdef continue else elseif end for function global if otherwise parfor persistent return spmd switch try while

When typed, these words appear in blue. An error message is displayed if the user tries to use a keyword as a variable name. (The keywords can be displayed by typ- ing the command iskeyword.)

A number of frequently used variables are already defmed when MATLAB is started. Some of the predefmed variables are:

ans A variable that has the value of the last expression that was not assigned to a specific variable (see Tutoriall-1). If the user does not assign the value of an expression to a variable, MATLAB automatically stores the result in ans.

pi The number 1t.

eps The smallest difference between two numbers. Equal to 2A(-52), which is approximately 2.2204e-O16.

inf Used for infmity.

i Defmed as ./-1, which is: 0 + l.OOOOi.

j Same as i.

NaN Stands for Not-a-Number. Used when MATLAB cannot determine a valid numeric value. Example: 0/0.

The predefmed variables can be redefined to have any other value. The vari- abies **pi, eps,** and **inf,** are usually not redefmed since they are frequently used in many applications. Other predefmed variables, such as **i** and **j** , are sometime redefined (commonly in association with loops) when complex numbers are not involved in the application.

1.7 *USEFUL COMMANDS FOR MANAGING VARIABLES*

The following are commands that can be used to eliminate variables or to obtain information about variables that have been created. When these commands are typed in the Command Window and the Enter key is pressed, either they provide information, or they perform a task as specified below.

Command	Outcome
clear	Removes all variables from the memory.

Command	Outcome
`clear x y z`	Removes only variables x, y, and z from the memory.
`who`	Displays a list of the variables currently in the memory.
`whos`	Displays a list of the variables currently in the memory and their sizes together with information about their bytes and class (see Section 4.1).

1.8 *SCRIPT FILES*

So far all the commands were typed in the Command Window and were executed when the Enter key was pressed. Although every MATLAB command can be executed in this way, using the Command Window to execute a series of commands-especially if they are related to each other (a program)-is not convenient and may be difficult or even impossible. The commands in the Command Window cannot be saved and executed again. In addition, the Command Window is not interactive. This means that every time the Enter key is pressed only the last command is executed, and everything executed before is unchanged. If a change or a correction is needed in a command that was previously executed and the result of this command is used in commands that follow, all the commands have to be entered and executed again.

A different (better) way of executing commands with MA TLAB is first to create a file with a list of commands (program), save it, and then run (execute) the file. When the file runs, the commands it contains are executed in the order that they are listed. If needed, the commands in the file can be corrected or changed and the file can be saved and run again. Files that are used for this purpose are called script files.

IMPORTANT NOTE: This section covers only the minimum required in order to run simple programs. This will allow the student to use script mes when practicing the material that is presented in this and the next two chap- ters (instead of typing repeatedly in the Command Window). Script mes are considered again in Chapter 4, where many additional topics that are essen- tial for understanding MATLAB and writing programs in script me are cov- ered.

1.8.1 *Notes About Script Files*

- A script file is a sequence ofMATLAB commands, also called a program.
- When a script file runs (is executed),MATLAB executes the commands in the order they are written, just as if they were typed in the Command Window.

- When a script file has a command that generates an output (e.g., assignment of a value to a variable without a semicolon at the end), the output is displayed in the Command Window.
- Using a script file is convenient because it can be edited (corrected or other- wise changed) and executed many times.
- Script files can be typed and edited in any text editor and then pasted into the MATLAB editor.
- Script files are also called M-files because the extension .m is used when they are saved.

1.8.2 *Creating and Saving a Script File*

In MATLAB script files are created and edited in the Editor/Debugger Window. This window is opened from the Command Window by clicking on the New Script icon in the Toolstrip, or by clicking New in the Toolstrip and then selecting Script from the menu that open. An open Editor/Debugger Window is shown in Figure 1-6.

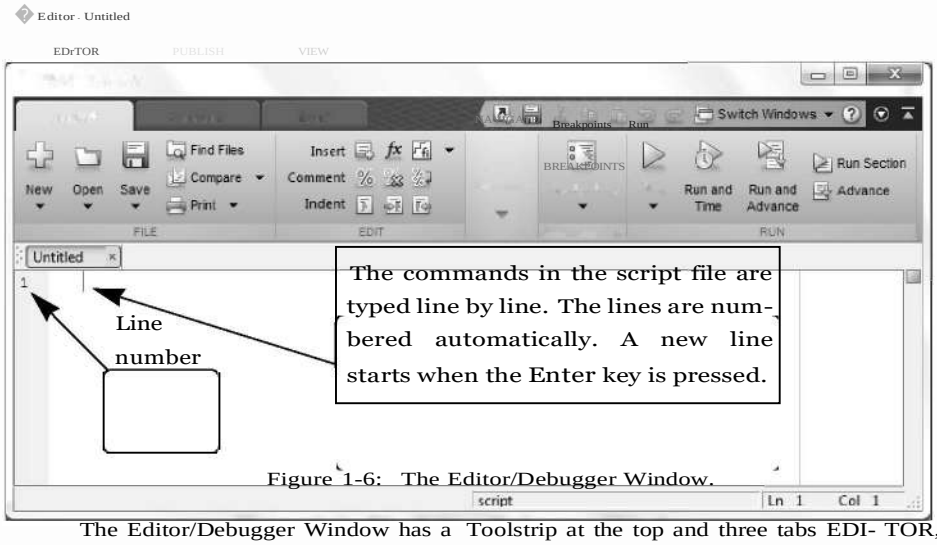


Figure 1-6: The Editor/Debugger Window.

The Editor/Debugger Window has a Toolstrip at the top and three tabs EDI- TOR, PUBLISH, and VIEW above it. Clicking on the tabs changes the icons in the Toolstrip. Commonly, MATLAB is used with the HOME tab selected. The associated icons are used for executing various commands, as explained later in the Chapter. Once the window is open, the commands of the script file are typed line by line. MATLAB automatically numbers a new line every time the Enter key is pressed. The commands can also be typed in any text editor or word proces- sor program and then copied and pasted in the Editor/Debugger Window. An example of a short program typed in the Editor/Debugger Window is shown in Figure 1-7. The first few lines in a script file are typically comments (which are

not executed, since the first character in the line is *%)* that describe the program written in the script file.

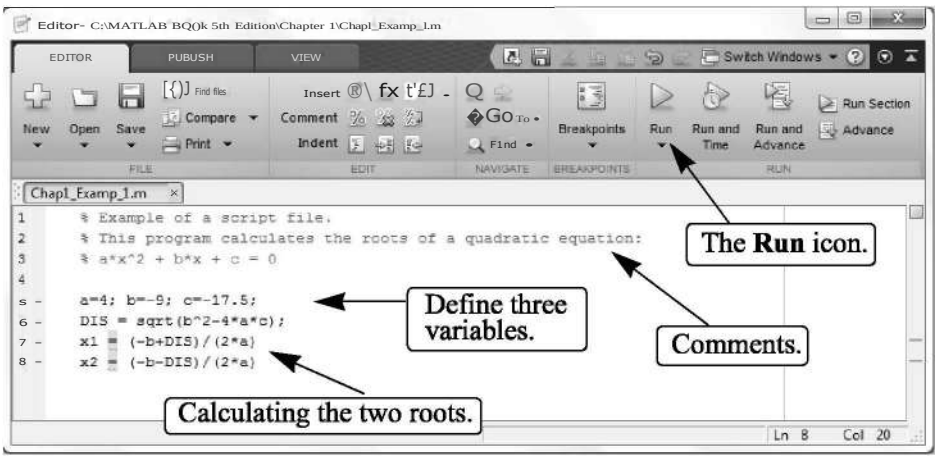


Figure 1-7: A program typed in the Editor/Debugger Window.

Before a script file can be executed it has to be saved. This is done by click- ing Save in the Toolstrip and selecting Save As... from the menu that opens. When saved,MATLAB adds the extension .m to the name. The rules for naming a script file follow the rules of naming a variable (must begin with a letter, can include digits and underscore, no spaces, and up to 63 characters long). The names of user-defmed variables, predefined variables, and MATLAB commands or func- tions should not be used as names of script files.

1.8.3 *Running (Executing)* a *Script File*

A script file can be executed either directly from the Editor Window by clicking on the Run icon (see Figure *1-7)* or by typing the file name in the Command Win-
dow and then pressing the Enter key. For a file to be executed, MATLAB needs to know where the file is saved. The file will be executed if the folder where the file is saved is the current folder ofMATLAB or if the folder is listed in the search path, as explained next.

1.8.4 *Current Folder*

The current folder is shown in the "Curent Folder" field in the desktop toolbar of the Command Window, as shown in Figure 1-8. If an attempt is made to execute a script file by clicking on the Run icon (in the Editor Window) when the curent
folder is not the folder where the script file is saved, then the prompt shown in Figure *1-9* opens. The user can then change the current folder to the folder where the script file is saved, or add it to the search path. Once two or more different cur-
rent folders are used in a session, it is possible to switch from one to another in the

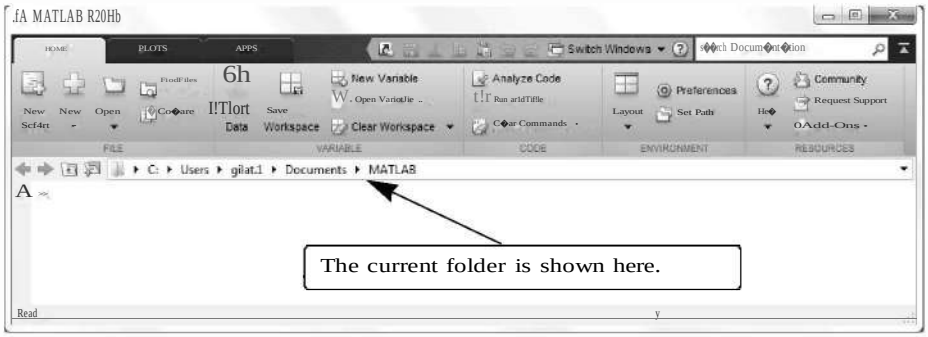


Figure 1-8: The Current folder field in the Command Window.

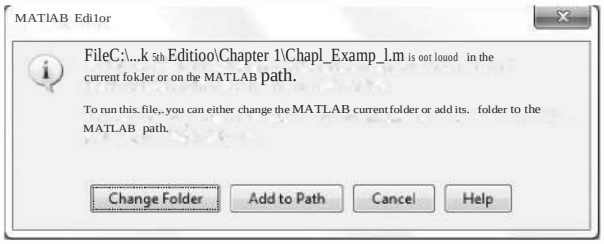


Figure 1-9: Changing the current directory.

Current Folder field in the Command Window. The current folder can also be changed in the Current Folder Window, shown in Figure 1-10, which can be opened from the Desktop menu. The Current Folder can be changed by choosing the drive and folder where the file is saved.

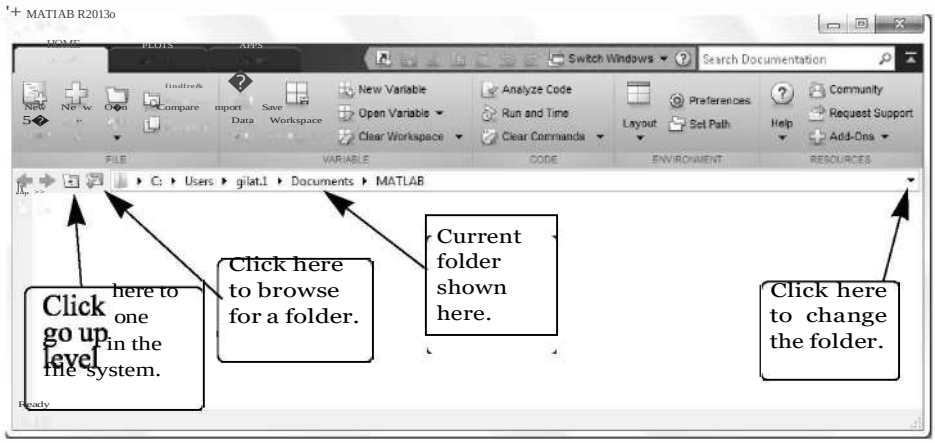


Figure 1-10: The Current Folder Window.

An alternative simple way to change the curent folder is to use the cd com- mand in the Command Window. To change the current folder to a different drive, type cd, space, and then the name of the directory followed by a colon: and press the Enter key. For example, to change the current folder to drive E (e.g., the flash drive) type cd E:. If the script file is saved in a folder within a drive, the path to that folder has to be specified. This is done by typing the path as a string in the cd command. For example, cd (1 E:\Chapter 1 1) sets the path to the folder Chapter 1 in drive F. The following example shows how the curent folder is changed to be drive E. Then the script file from Figure 1-7, which was saved in drive E as ProgramExample.m, is executed by typing the name of the file and pressing the Enter key.

```
>> cd { 'E:\Chapter 1')

>> Chapl_Exampl
x1 =
     3.5000
x2 =
    -1.2500
```

The curent directory is changed to drive E.

The script file is executed by typing the name of the file and pressing the Enter key.

The output generated by the script file (the roots x1 and x2) is displayed in the Command Window.

1.9 EXAMPLES OF MATLAB APPLICATIONS

Sample Problem 1-1: Trigonometric identity

A trigonometric identity is given by:

$$\cos^{z:!}_{2} - \frac{\tan x + \sin x}{2\tan x}$$

Verify that the identity is correct by calculating each side of the equation, substi- tutmg x – 5.

Solution

The problem is solved by typing the following commands in the Command Win- dow.

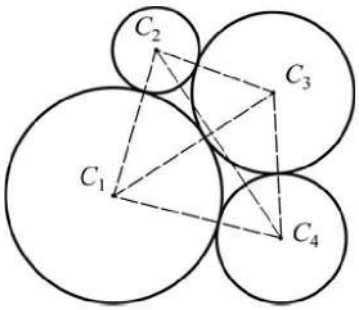
```
>> X=pi/5;
>> LHS=cos(x/2)A2
LHS =
     0.9045
>>  RHS=(tan(x)+sin(x))/(2*tan(x))
RHS =
     0.9045
```

Define x. Calculate the left-hand side.

Calculate the right-hand side.

Sample Problem 1-2: Geometry and trigonometry

Four circles are placed as shown in the figure. At each point where two circles are in contact, they are tangent to each other. Determine the distance between the centers C2 and C4.

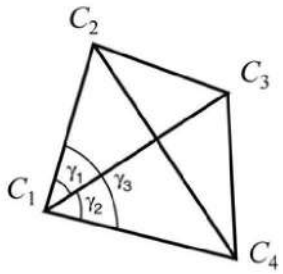


The radii of the circles are:

R1 = 16mm, R2 = 6.5mm, R3 = 12 mm, and R4 = 9.5mm.

Solution

The lines that connect the centers of the circles create four triangles. In two of the triangles, AC1C2C3 and AC1C3C4, the lengths of all the sides are known. This information is used to calculate the angles y1 and y2 in these triangles by using the law of cosines. For example, y1 is calculated from:



$$(C2C3)^2 = (C1C2)^2 + (C1C3)^2 - 2(C1C2)(C1C3)\cos y1$$

Next, the length of the side C2C4 is calculated by considering the triangle AC1C2C4. This is done, again, by using the law of cosines (the lengths clCz and c1c4 are known and the angle '¥3 is the sum of the angles 'YI and y2).

The problem is solved by writing the following program in a script file:

```
% Solution of Sample Problem 1-2
R1=16; R2=6.5; R3=12; R4=9.5;
ClC2=R1+R2; ClC3=R1+R3; ClC4=R1+R4;
C2C3=R2+R3; C3C4=R3+R4;
Gamal=acos((ClC2A2+ClC3A2-C2C3A2)/(2*ClC2*ClC3));
Gama2=acos((ClC3A2+ClC4A2-C3C4A2)/(2*ClC3*ClC4));
Gama3=Gamal+Gama2;

C2C4=sqrt(ClC2A2+ClC4A2-2*ClC2*ClC4*cos(Gama3))
```

Define the R's.

Calculate the lengths of the sides.

Calculate γ_1, γ_2, and γ_3.

Calculate the length of side C2C4.

When the script file is executed, the following (the value of the variable C2C4) is displayed in the Command Window:

```
C2C4 =
    33.5051
```

Sample Problem 1-3: Heat transfer

An object with an initial temperature of T0 that is placed at time t =0 inside a chamber that has a constant temperature of T3 will experience a temperature change according to the equation

$$T = T3+(T0-T3)e^{-kt}$$

where Tis the temperature of the object at timet, and k is a constant. A soda can at a temperature of 120° F (after being left in the car) is placed inside a refrigerator where the temperature is 38°F. Determine, to the nearest degree, the temperature of the can after three hours. Assume k=0.45. First define all of the variables and then calculate the temperature using one MATLAB command.

Solution

The problem is solved by typing the following commands in the Command Win- dow.

```
>> Ts=38;  T0=120;  k=0.45;  t=3;

>> T=round(Ts+(TO-Ts)*exp(-k*t))

T =
     59
```

Round to the nearest integer.

Sample Problem 1-4: Compounded interest

The balance Bof a savings account aftert years when a principal P is invested at an annual interest rate r and the interest is compounded n times a year is given by:

$$B=P\left(t+\frac{r}{n}\right)^{t} \qquad (1)$$

If the interest is compounded yearly, the balance is given by:

$$B = P(1+r)t \qquad (2)$$

Suppose $5,000 is invested for 17 years in one account for which the interest is compounded yearly. In addition, $5,000 is invested in a second account in which the interest is compounded monthly. In both accounts the interest rate is 8.5%. Use MATLAB to determine how long (in years and months) it would take for the balance in the second account to be the same as the balance of the first account after 17 years.

Solution

Follow these steps:

(a) Calculate Bfor $5,000 invested in a yearly compounded interest account after 17 years using Equation (2).

(b) Calculate t for the B calculated in part (a), from the monthly compounded interest formula, Equation (1).

(c) Determine the number of years and months that correspond tot.

The problem is solved by writing the following program in a script file:

```
% Solution of Sample Problem 1-4
P=SOOO;  r=0.085;    ta=l7;  n=l2;
B=P*{l+r)"ta
t=log{B/P)/{n*log{l+r/n))

years=fix{t)
months=ceil{{t-years)*12)
```

Step (a): Calculate B from Eq. (2).

Step (b): Solve Eq. (1) fort, and calculate t.

Step (c): Determine the number of years.

Determine the number of months.

When the script file is executed, the following (the values of the variables B, t, years, and months) is displayed in the Command Window:

```
>> format short g B
   =        20011

t =
         16.374

years
     16=

 months
      5 =
```

The values of the variables B, t, years, and months are displayed (since a semicolon was not typed at the end of any of the commands that calculate the values).

1.10 PROBLEMS

The following problems can be solved by writing commands in the Command Window, or by writing a program in a script file and then executing the file.

1. Calculate:

   (a) $\frac{22 + 5.12}{50 \_ 6.32}$

   (b) $\frac{44 + 8^2 - 99}{7 \quad 5 \quad 3.92}$

2. Calculate:

   (a) $\frac{5.22}{e5 - 100.53}$ / $\sqrt{41^2 -}$

   (b) $\text{Vil} + \frac{\ln(500)}{8}$

3. Calculate:

(a) $\dfrac{14.83\text{-}6.32}{(\text{Jf3} + 5)^2}$ (b) $45\left(\dfrac{288}{9.3} - 4.6^2\right) - 1065e^{-1.5}$

4. Calculate:

(a) $\dfrac{24.5 + 64/3.52 + 8.3 \cdot 12.53}{\text{J76A.-28/15}}$ (b) $(5.9^2 - 2.4^2)/3 + \left(\dfrac{\log_{10}12890}{e^{0.3}}\right)^2$

5. Calculate:

(a) $\cos\left(\dfrac{7\pi}{9}\right) + \tan\left(\dfrac{7}{15}\pi\right)\sin(15°)$ (b) $\sin^2 80° - \dfrac{(\cos 14° \sin 80°)^2}{\sqrt[3]{0.18}}$

6. Defme the variable x as x= 6.7, then evaluate:

(a) 0.01x5-1.4x3 + 80x+ 16.7 (b) $\sqrt{x^3 + e^x - 51/x}$

7. Defme the variable t as t= 3.2, then evaluate:

(a) $56t\text{-}^{9}\ .81_{2}^{t2}$ (b) 14e-O.ltsin(21tt)

8. Defme the variables x andy as x= 5.1 andy= 4.2, then evaluate:

(a) $\frac{3}{4}xy\text{-}\frac{7x}{y2} + \sqrt{xy}$ (b) (xy)2- $\dfrac{x+y}{(x\text{-}y)12} + \sqrt{\dfrac{x+y}{2x\text{-}y}}$

9. Defme the variables a, b, c, and d as:

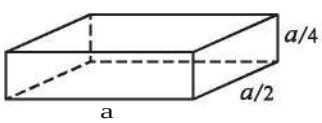
a= 12, b = 5.6, c = ❓❓ ,and d = $\dfrac{(a-b)^c}{c}$, then evaluate:

(a) $\frac{a}{b} + \frac{d\text{-}e}{d+c}\text{-}(d\text{-}b)2$ (b) $e^{\frac{d\text{-}e}{a-2b}} + \ln\left(\left|c - d + \dfrac{b}{a}\right|\right)$

10. A sphere has a radius of24 em. A rectangular prism has sides of a, a/2, and a/4 .

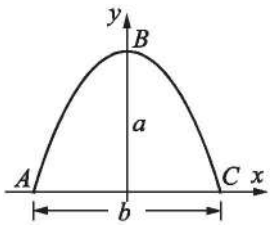
(a) Determine a of a prism that has the same volume as the sphere.

(b) Determine a of a prism that has the same surface area as the sphere.



11. The arc length of a segment of a parabola ABC of an ellipse with semi-minor axes a and b is given approximately by:

$$L_{ABC} = \frac{1}{2}\sqrt{b^2+16a^2} + \frac{b^2}{8a}\ln\left(\frac{4a+\sqrt{b^2+16a^2}}{b}\right).$$



(a) Determine L_{ABC} if a = 11 in. and b = 9 in.

12. Two trigonometric identities are given by:

(a) $\sin 5x = 5\sin x - 20\sin^3 x + 16\sin^5 x$ (b) $\sin^2 x\cos^2 x = \frac{1-\cos 4x}{8}$

For each part, verify that the identity is correct by calculating the values of the left and right sides of the equation, substituting x = ?.

13. Two trigonometric identities are given by:

(a) $\tan 3x = \frac{3\tan x - \tan^3 x}{1-3\tan^2 x}$ (b) $\cos 4x = 8(\cos^4 x - \cos^2 x) + 1$

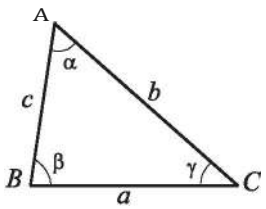
For each part, verify that the identity is correct by calculating the values of the left and right sides of the equation, substituting $x = 24^\circ$.

14. Defme two variables: alpha=rt/6, and beta= 3rt/8. Using these variables, show that the following trigonometric identity is correct by calculating the values of the left and right sides of the equation.

$$\sin\alpha + \sin\beta = 2\sin\left(\frac{\alpha+\beta}{2}\right)\cos\left(\frac{\alpha-\beta}{2}\right)$$

15. Given: $\int x^2 \cos ax\,dx = \frac{2x\cos ax}{a^2} + \frac{(a^2x^2-2)\sin ax}{a^3}$. Use MATLAB to calculate the following definite integral: $\int_3^5 2x\sin(0.6x)\,dx$.

16. In the triangle shown a = 5.3in., $\gamma = 42^\circ$, and in. b = 6 Defme a, γ, and b as variables, and then:



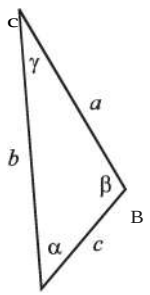
(a) Calculate the length b by using the Law of Cosines.

(Law of Cosines: $c^2 = a^2 + b^2 - 2ab\cos\gamma$)

(b) Calculate the angles ? andy (in degrees) using the Law of Cosines.

(c) Check that the sum of the angles is 180°.

17. In the triangle shown a =5 in., b =7 in., and y =25°. Define a, b, andy as variables, and then:



(a) Calculate the length of c by substituting the variables in the Law of Cosines.

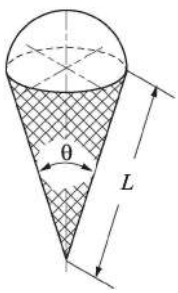
(Law of Cosines: c2 =a2+b2-2abcosy)

(b) Calculate the angles a and (in degrees) using the Law of Sines.

(c) Verity the Law of Tangents by substituting the results from part (b) into the right and left sides of the equation.

Law of Tangents: $\frac{a-b}{a+b} = \frac{\tan\left[\frac{1}{2}(\alpha-\beta)\right]}{[(a+)]}$

18. In the ice cream cone shown, L =4 in. and e =35°. The cone is filled with ice cream such that the portion above the cone is a hemisphere. Determine the volume of the ice cream.



19. For the triangle shown, a =48 mm, b = 34 mm, and y =83o. Define a, b, and y as variables, and then:

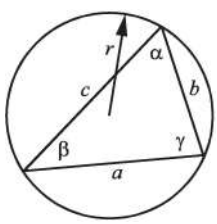
(a) Calculate c by substituting the variables in the Law of Cosines.

(Law of Cosines: c2 =a2+b2-2abcosy)

(b) Calculate the radius r of the circle circumscribing the triangle using the formula:

$$r = \frac{abc}{4\sqrt{s(s-a)(s-b)(s-c)}}$$

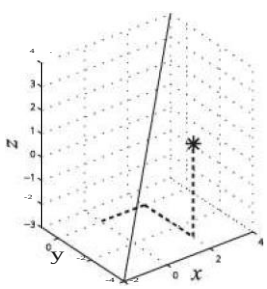
where s =(a+b+c)/2.



20. The parametric equations of a line in space are: x =x0+at, y =y0+bt, and z =z0+ct. The distance dfrom a point A (xA, yA, zA) to the line can be calculated by:

$$d = dA_o \sin\left[\operatorname{acos}\left(\frac{(x_A-x_0)a+(y_A-y_0)b+(z_A-z_0)c}{d_{A0}\sqrt{a^2+b2+c2}}\right)\right]$$

where $dAo = \sqrt{(x_A-x_0)^2+(y_A-y_0)^2+(z_A-z_0)^2}$.

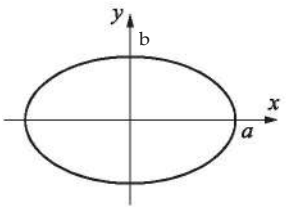


Determine the distance of the point A (2,-3, 1) from the line x =-4+0.6t, y =-2+0.5t, and z =- 3+0.7t. First define the variables x0, y0, z0, a, b, and c, then use the variable (and the coordinates of point A) to calculate the variable dAo, and finally calculate d.

21. The circumference of an ellipse can be approximated by:

$$C = 1t[3(a+b) - \sqrt{(3a+b)(a+3b)}]$$

Calculate the circumference of an ellipse with $a = 16$ in. and $b = 11$ in.



22. 315 people have to be transported using buses that have 37 seats. By typing one line (command) in the Command Window, calculate how many seats will remain empty if enough buses will be ordered to transport all the people. (Hint: use MATLAB built-in function ceil.)

23. 739 apples are to be packed and shipped such that 54 are placed in a box. By typing one line (command) in the Command Window, calculate how many apples will remain unpacked if only full boxes can be shipped. (Hint: use MATLAB built-in function fix.)

24. Assign the number 316,501.673 to a variable, and then calculate the following by typing one command:

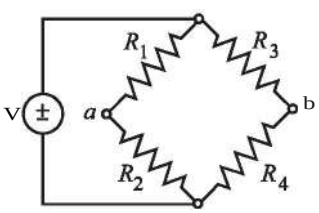
(a) Round the number to the nearest hundredth.

(b) Round the number to the nearest thousand.

25. The voltage difference V_{ab} between points a and b in the Wheatstone bridge circuit is:

$$V_{ab} = V\left(\frac{R_1R_3 - R_2R_4}{(R_1+R_2)(R_3+R_4)}\right)$$

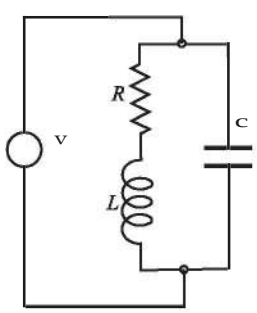
Calculate the voltage difference when V = 14 volts, R1 = 120.6 ohms, R2 = 119.3 ohms, R3 = 121.2 ohms, and R4 = 118.8 ohms.



26. The resonant frequency f (in Hz) for the circuit shown is given by:

$$f = \frac{1}{2\pi}\sqrt{\frac{1}{LC} - \left(\frac{R}{L}\right)^2}$$

Calculate the resonant frequency when L = 0.15 henrys, R = 14 ohms, and C = 2.6 x IQ-6farads.



27. The number of combinations $C_{n,r}$ of taking r objects out of n objects is given by:

$$C_{n,r} = \frac{n!}{r!(n-r)!}$$

(a) Determine how many combinations are possible in a lottery game for selecting 6 numbers that are drawn out of 49.

(b) Using the following formula, determine the probability of guessing two out of the six drawn numbers.

$$\frac{c6,2\,c43,4\;c49,6}{}$$

(Use the built-in function factorial.)

28. The formula for changing the base of a logarithm is:

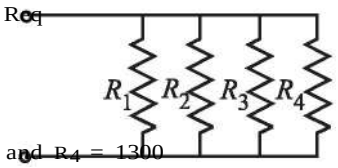
$$\log_a N = \frac{\log_b N}{\log_b a}$$

(a) Use MATLAB's function log (x) to calculate $\log_4 0.085$.

(b) Use MATLAB's function loglO (x) to calculate $\log_6 1500$.

29. The equivalent resistance, Req, of four resistors, R1, R2, R3, and R4, that are connected in parallel is given by:

$$Req = \frac{1}{\frac{1}{RI} + \frac{1}{R2} + \frac{1}{R3} + \frac{1}{R4}}$$

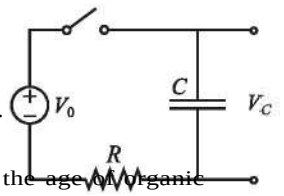


Calculate Req if R1 = 1200, R2 = 2200, R3 = 750, and R4 = 1300

30. The voltage Vc t seconds after closing the switch in the circuit shown is:

$$Vc = Vo(1 - e^{-ti(RC)})$$

Given Vc = 36 V, R = 2500 O, and C = 1600 JlF, calculate the current 8 seconds after the switch is closed.



31. Radioactive decay of carbon-14 is used for estimating the age of organic material. The decay is modeled with the exponential function f(t) . f(O)ekt, where tis time, f(O) is the amount of material at t _ 0, f(t) is the amount of material at time t, and k is a constant. Carbon-14 has a half-life of approximately 5,730 years. A sample taken from the ancient footprints of Acahual- inca in Nicaragua shows that 77.45% of the initial (t _ 0) carbon-14 is

present. Determine the estimated age of the footprint. Solve the problem by writing a program in a script file. The program first determines the constant k, then calculates t for f(t) = 0.7745f(O), and fmally rounds the answer to the nearest year.

32. The greatest common divisor is the largest positive integer that divides the numbers without a remainder. For example, the GCD of8 and 12 is 4. Use the MATLAB Help Window to fmd a MATLAB built-in function that determines the greatest common divisor of two numbers. Then use the function to show that the greatest common divisor of:
    (a) 91 and 147 is 7.
    (b) 555 and 962 is 37.

33. The Moment Magnitude Scale (MMS), denoted Mw, which measures the total energy released by an earthquake, is given by:

$$Mw = \frac{2}{3}\log 10M0 - 10.7$$

    where M0 is the magnitude of the seismic moment in dyne-em (measure of the energy released during an earthquake). Determine how many times more energy was released from the largest earthquake in the world, in Chile (Mw = 9.5), 1960, than the earthquake in Rat Island, Alaska (Mw = 8.7), in 1965.

34. According to special relativity, a rod of length L moving at velocity v will shorten by an amount c, given by:

$$c - L\left(1 - \sqrt{1 - \square}\right)$$

    where cis the speed of light (about 300 x 106 m/s). Calculate how much a rod 2 m long will contract when traveling at 5,000 m!s.

35. The value B of a principalPthat is deposited in a saving account with a fixed annual interest rate r after n years can be calculated by the formula:

$$B = P\left(1 + \frac{r}{m}\right)^{nm}$$

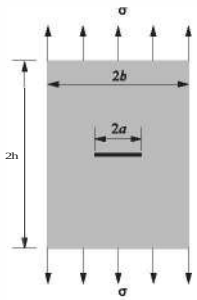
    where m is the number of times that the interest is compounded annually. Consider a $80,000 deposit for 5 years. Determine how much more money will be earned if the interest is compounded daily instead of yearly.

36. Newton's law of cooling gives the temperature T(t) of an object at time tin terms of T0, its temperature at t - 0, and Ts, the temperature of the surroundings.

$$T(t) - T3 + (T0-T3)e\text{-}kt$$

A police officer arrives at a crime scene in a hotel room at 9:18PM, where he fmds a dead body. He immediately measures the body's temperature and fmd it to be 79.5° F. Exactly one hour later he measures the temperature again, and fmd it to be 78.0° F. Determine the time of death, assuming that victim body temperature was normal (98.6° F) prior to death, and that the room tempera- ture was constant at 69° F.

37. The stress intensity factor K predicts the stress state (stress intensity) near a crack tip. F or a plate with a crack and loading shown in the figure, K is given by:



$$K = \text{cr}\,\text{JitQ}\left[\frac{1\text{-}fz; + 0.326\left(\blacklozenge\right)^{2}}{\sqrt{1-\frac{a}{b}}}\right]$$

Determine K for the case where cr= 12000psi, h= 5in., and b - 4 in., and a - 1.5 in.

38. The spread of a computer virus through a computer network can be modeled by:

$$N(t) = 20e^{O.I5t}$$

where N(t) is the number of computers infected and t time in minutes.

(a) Determine how long it takes for the number of infected computers to dou- ble.

(b) Determine how long it takes for 1,000,000 computers to be infected.

39. Use the Help Window to find a display format that displays the output as a ratio of integers. For example, the number 3.125 will be displayed as 25/8. Change the display to this format and execute the following operations:

(a) $5/8 + 16/6$ (b) $1/3 - 11/13 + 2.72$

40. Stirling's approximation for large factorials is given by:

$$n! = \frac{\left(\blacklozenge\right)_{n}}{\sqrt{2\pi n}}$$

Use the formula for calculating 20!. Compare the result with the true value obtained with MATLAB's built-in function **factorial** by calculating the error (Error= (TrueVal-ApproxVal)/ TrueVal).

Unit 2

Creating Arrays

The array is a fundamental form that MATLAB uses to store and manipulate data. An aray is a list of numbers aranged in rows and/or columns. The simplest array (one-dimensional) is a row or a column of numbers. A more complex array (two- dimensional) is a collection of numbers arranged in rows and columns. One use of arrays is to store information and data, as in a table. In science and engineering, one-dimensional arrays frequently represent vectors, and two-dimensional arrays often represent matrices. This chapter shows how to create and address arrays, and Chapter *3* shows how to use arrays in mathematical operations. *In* addition to arrays made of numbers, arrays in MATLAB can also be a list of characters,
which are called strings. Strings are discussed in Section 2.10.

2.1 CREATING A ONE-DIMENSIONAL *ARRAY* {VECTOR)

A one-dimensional array is a list of numbers arranged in a row or a column. One example is the representation of the position of a point in space in a three-dimen-
sional Cartesian coordinate system. As shown in Figure 2-1, the position of point *A* is defmed by a list of the three numbers 2, 4, and 5, which are the coordinates of the point.

The position of point *A* can be expressed in terms of a position vector:

$$r_A = 2i + 4j + 5k$$

where i, j, and *k* are unit vectors in the direction of the *x, y,* and z axes, respectively. The numbers 2, 4, and 5 can be used to defme a row or a column vector.

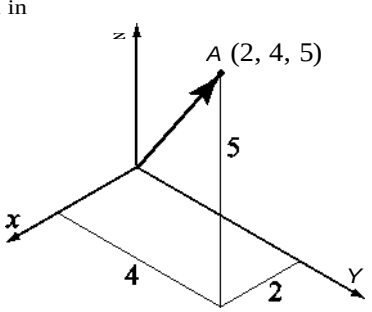


Figure 2-1: Position of a point.

Any list of numbers can be set up as a vector. For example, Table 2-1 con- tains population growth data that can be used to create two lists of numbers--one of the years and the other of the population values. Each list can be entered as elements in a vector with the numbers placed in a row or in a column.

Table 2-1: Population data

Year	1984	1986	1988	1990	1992	1994	1996
Population (millions)	127	130	136	145	158	178	211

In MATLAB, a vector is created by assigning the elements of the vector to a variable. This can be done in several ways depending on the source of the information that is used for the elements of the vector. When a vector contains specific numbers that are known (like the coordinates of point A), the value of each element is entered directly. Each element can also be a mathematical expression that can include predefined variables, numbers, and functions. Often, the elements of a row vector are a series of numbers with constant spacing. In such cases the vector can be created with MATLAB commands. A vector can also be created as the result of mathematical operations as explained in Chapter 3.

Creating a vector from a known list of numbers:

The vector is created by typing the elements (numbers) inside square brackets [ ].

```
variable_name = [ type vector elements ]
```

Row vector: To create a row vector type the elements with a space or a comma between the elements inside the square brackets.

Column vector: To create a column vector type the left square bracket [and then enter the elements with a semicolon between them, or press the Enter key after each element. Type the right square bracket] after the last element.

Tutorial 2-1 shows how the data from Table 2-1 and the coordinates of point A are used to create row and column vectors.

Thtorial 2-1: Creating vectors from given data.

```
>> yr=[1984 1986 1988 1990 1992 1994 1996]

yr =
    1984    1986    1988    1990    1992    1994    1996

>> pop=[127; 130; 136; 145; 158; 178; 211]

pop =
    127
    130
    136
    145
    158
```

The list of years is assigned to a row vector named yr.

The population data is assigned to a column vector named pop.

Tutorial2-1: Creating vectors from given data. (Continued)

```
      178
      211
>> pntAH=[2,      4,     5]
pntAH =
       2       4       5
>> pntAV=[2
4
5]
pntAV
       2
       4
       5
>>
```

The coordinates of point A are assigned to a row vector called pntAH.

The coordinates of point A are assigned to a column vector called pntAV. (The Enter key is pressed after each element is typed.)

Creating a vector with constant spacing by specifying the first term, the spac- ing, and the last term:

In a vector with constant spacing, the difference between the elements is the same. For example, in the vector v = 2 4 6 8 10, the spacing between the elements is 2. A vector in which the ftrst term ism, the spacing is q, and the last term is n is created by typing:

m:q:n

variable_name = [m:q:n] or variable_name = (The brackets are optional)

Some examples are:

```
>> X=[1:2:13)
X =
      1       3       5       7       9      11      13
>> y=[1.5:0.1:2.1]
y =
   1.5000   1.6000   1.7000   1.8000   1.9000   2.0000   2.1000
>> Z=(-3:7)
    -3    -2    -1     0     1     2     3     4     5     6
7
z =
>> xa=[21:-3:6]
```

First element 1, spacing 2, last element

13.

First element 1.5, spacing 0.1, last element

21.

First element -3, last term 7.
If spacing is omitted, the default is 1.

First element 21, spacing -3, last term

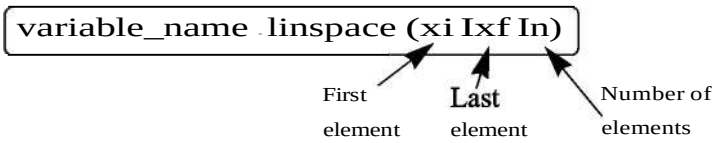
6.

```
xa =
      21     18     15     12      9      6
>>
```

- If the numbersm, q, and n are such that the value of n cannot be obtained by adding q's tom, then (for positive n) the last element in the vector will be the last number that does not exceed n.
- If only two numbers (the first and the last terms) are typed (the spacing is omit- ted), then the default for the spacing is 1.

Creating a vector with linear (equal) spacing by specifying the first and last terms, and the number of terms:

A vector with n elements that are linearly (equally) spaced in which the first ele- ment is xi and the last element is xfcan be created by typing the linspace com- mand (MATLAB determines the correct spacing):



When the number of elements is omitted, the default is 100. Some examples are:

```
>> va=linspace(0,8,6)          6 elements, first element 0, last element 8.
va =
      0    1.6000    3.2000    4.8000    6.4000    8.0000
>> vb=linspace(30,10,11)       11 elements, first element 30, last element 10.
vb =
    30    28    26    24    22    20    18    16    14    12    10
>> u=linspace(49.5,0.5)        First element 49.5, last element 0.5.
                               When the number of elements is
u =                            omitted, the default is 100.
  Columns 1 through 10
   49.5000   49.0051   48.5101   48.0152   47.5202   47.0253
46.5303   46.0354   45.5404   45.0455
.............
Columns 91 through 100         100 elements are displayed.
    4.9545    4.4596    3.9646    3.4697    2.9747    2.4798
1.9848    1.4899    0.9949    0.5000
>>
```

2.2 CREATING A TwO-DIMENSIONAL ARRAY (MATRIX)

A two-dimensional array, also called a matrix, has numbers in rows and columns. Matrices can be used to store information like the arrangement in a table. Matrices play an important role in linear algebra and are used in science and engineering to describe many physical quantities.

In a square matrix the number of rows and the number of columns is equal. For example, the matrix

```
7  4  9
3  8  1     3 x 3 matrix
6  5  3
```

is square, with three rows and three columns. In general, the number of rows and columns can be different. For example, the matrix:

```
31  26  14  18   5  30
 3  51  20  11  43  65      4 x 6 matrix
28   6  15  61  34  22
14  58   6  36  93   7
```

has four rows and six columns. A $m \times n$ matrix has m rows and n columns, and m by n is called the size of the matrix.

A matrix is created by assigning the elements of the matrix to a variable. This is done by typing the elements, row by row, inside square brackets [ ]. First type the left bracket [then type the first row, separating the elements with spaces or commas. To type the next row type a semicolon or press Enter. Type the right bracket] at the end of the last row.

```
variable_name= [1st row elements; 2nd row elements; 3rd
                row elements;  . . . ; last row elements]
```

The elements that are entered can be numbers or mathematical expressions that may include numbers, predefined variables, and functions. All *the rows must have the same number of elements.* If an element is zero, it has to be entered as such. MATLAB displays an error message if an attempt is made to defme an incomplete matrix. Examples of matrices defmed in different ways are shown in Tutorial 2-2.

Tutorial2-2: Creating matrices.

```
>> a=[S   35   43;   4   76   81;   21   32   40]
a =
     5     35     43
     4     76     81
    21     32     40
>> b = [7   2   76   33   8
1   98   6   25   6
5   54   68   9   0]
```

A semicolon is typed before a new line is entered.

The Enter key is pressed before a new line is entered.

Tutorial 2-2: Creating matrices. (Continued)

```
b =
     7     2    76    33     8
     1    98     6    25     6
     5    54    68     9     0
>> cd=6; e=3; h=4;
>> Mat=[e, cd*h, cos(pi/3); hA2, sqrt(h*h/cd), 14]
Mat =  3.0000   24.0000    0.5000
      16.0000    1.6330   14.0000

>>
```

Three variables are defined.

Elements are defmed by mathematical expressions.

Rows of a matrix can also be entered as vectors using the notation for creat- ing vectors with constant spacing, or the `linspace` command. For example:

```
>> A=[1:2:11; 0:5:25; 1inspace(10,60,6);  67 2 43 68 4 13]
A =
     1     3     5     7     9    11
     0     5    10    15    20    25
    10    20    30    40    50    60
    67     2    43    68     4    13

>>
```

In this example the frrst two rows were entered as vectors using the notation of constant spacing, the third row was entered using the `linspace` command, and in the last row the elements were entered individually.

2.2.1 The zeros, ones and, eye Commands

The `zeros (m, n)`, `ones (m, n)`, and `eye (n)` commands can be used to create matrices that have elements with special values. The `zeros (m, n)` and the `ones (m, n)` commands create a matrix with `m` rows and `n` columns in which all elements are the numbers 0 and 1, respectively. The `eye ( n)` command creates a square matrix with `n` rows and `n` columns in which the diagonal elements are equal to 1 and the rest of the elements are 0. This matrix is called the identity matrix. Examples are:

```
>> zr=zeros(3,4)
zr =  0     0     0     0
      0     0     0     0
      0     0     0     0


>> ne=ones(4,3)
```

```
ne =
     1     1     1
     1     1     1
     1     1     1
     1     1     1
>> idn=eye(S)
idn =
     1     0     0     0     0
     0     1     0     0     0
     0     0     1     0     0
     0     0     0     1     0
     0     0     0     0     1
>>
```

Matrices can also be created as a result of mathematical operations with vectors and matrices. This topic is covered in Chapter 3.

2.3 NOTES ABOUT VARIABLES IN MATLAB

- All variables in MATLAB are arrays. A scalar is an array with one element, a vector is an array with one row or one column of elements, and a matrix is an array with elements in rows and columns.
- The variable (scalar, vector, or matrix) is defined by the input when the vari- able is assigned. There is no need to defme the size of the array (single element for a scalar, a row or a column of elements for a vector, or a two-dimensional array of elements for a matrix) before the elements are assigned.
- Once a variable **exists-as** a scalar, vector, or **matrix-it** can be changed to any other size, or type, of variable. For example, a scalar can be changed to a vec- tor or a matrix; a vector can be changed to a scalar, a vector of different length, or a matrix; and a matrix can be changed to have a different size, or be reduced to a vector or a scalar. These changes are made by adding or deleting elements. This subject is covered in Sections 2.7 and 2.8.

2.4 THE TRANSPOSE OPERATOR

The transpose operator, when applied to a vector, switches a row (column) vector to a column (row) vector. When applied to a matrix, it switches the rows (col- umns) to columns (rows). The transpose operator is applied by typing a single quote ' following the variable to be transposed. Examples are:

```
>> aa=[3  8   1]
aa =
     3     8     1
>> bb=aa'
```

Defme a row vector aa.

Defme a column vector bb as the transpose of vector aa.

```
bb =
     3
     8
     1
>> C=[2 55 14 8; 21 5 32 11; 41 64 9 1]
C =
     2    55    14     8
    21     5    32    11
    41    64     9     1
>> D=C'
D =
     2    21    41
    55     5    64
    14    32     9
     8    11     1
>>
```

Defme a matrix C with 3 rows and 4 columns.

Defme a matrix D as the transpose of matrix C. (D has 4 rows and 3 columns.)

2.5 ARRAYADDRESSING

Elements in an array (either vector or matrix) can be addressed individually or in subgroups. This is useful when there is a need to redefme only some of the ele- ments, when specific elements are to be used in calculations, or when a subgroup of the elements is used to defme a new variable.

2.5.1 Vector

The address of an element in a vector is its position in the row (or column). For a vector named ve, ve (k) refers to the element in position k. The first position is 1. For example, if the vector ve has nine elements:

ve _ 35 46 78 23 5 14 81 3 55

then

ve(4) _ 23, ve(7) _ 81, and ve(1) _ 35.

A single vector element, v(k), can be used just as a variable. For example, it is possible to change the value of only one element of a vector by assigning a new value to a specific address. This is done by typing: v(k) _ value. A single element can also be used as a variable in a mathematical expression. Examples are:

```
» VCT=[35 46 78 23 5 14 81 3 55]
VCT =
    35    46    78    23     5    14    81     3    55
» VCT(4)
```

Defme a vector.

Display the fourth element.

```
ans = 23
>> VCT(6)=273
VCT
    35    46    78    23     5   273    81     3    55
```

Assign a new value to the sixth element.

The whole vector is displayed.

```
>> VCT(2)+VCT(8)
ans =
    49
>> VCT(5)AVCT(8)+sqrt(VCT(7))
ans = 134
>>
```

Use the vector elements in mathematical expressions.

2.5.2 Matrix

The address of an element in a matrix is its position, defined by the row number and the column number where it is located. For a matrix assigned to a variable ma, ma(k,p) refers to the element in row k and columnp.

For example, if the matrix is: $ma = \begin{bmatrix} 1 & & 5 \\ 71 & 160 & 2 \\ 13 & 9 & 0 & 8 \end{bmatrix}$

then ma(1,1) = 3 and ma(2,3) = 10.

As with vectors, it is possible to change the value of just one element of a matrix by assigning a new value to that element. Also, single elements can be used like variables in mathematical expressions and functions. Some examples are:

```
>> MAT=[3  11  6  5;  4  7  10  2;  13  9  0  8]
MAT =
     3    11     6     5
     4     7    10     2
    13     9     0     8
>> MAT(3,1)=20
MAT =
     3    11     6     5
     4     7    10     2
    20     9     0     8
>> MAT(2,4)-MAT(1,2)
ans =
```

Create a 3 X 4 matrix.

Assign a new value to the (3,1) element.

Use elements in a mathematical expression.

2.6 USING A COLON : IN ADDRESSING ARRAYS

A colon can be used to address a range of elements in a vector or a matrix.

For a vector:

va(:) Refers to all the elements of the vector *va* (either a row or a column vector).

va(m:n) Refers to elements *m* through *n* of the vector *va.*

Example:

```
>> V=[4  15  8  12  34  2  50  23  11]
V =
     4    15     8    12    34     2    50    23    11
>> U=V(3:7)
U =
     8    12    34     2    50
>>
```

A vector V is created.

A vector u is created from the elements 3 through 7 of vector v.

For a matrix:

A(:,n) Refers to the elements in all the rows of column *n* of the matrix *A. A(n,:)* Refers to the elements in all the columns of row *n* of the matrix *A.*

A(:,m:n) Refers to the elements in all the rows between columns *m* and *n* of the matrix *A.*

A(m:n,:) Refers to the elements in all the columns between rows *m* and *n* of the matrix *A.*

A(m:n,p:q) Refers to the elements in rows *m* through nand columns p through *q* of the matrix *A.*

The use of the colon symbol in addressing elements of matrices is demon- strated in Tutorial2-3.

Tutorial2-3: Using a colon in addressing arrays.

```
>> A=[1  3  57  9  11;  2  4  6  8  10  12;  3  6  9  12  15  18;  4  8  12  16
20  24;  5  10  15  20  25  30]
```

Defme a matrix A with 5 rows and 6 columns.

```
A
     1     3     5     7     9    11
     2     4     6     8    10    12
     3     6     9    12    15    18
     4     8    12    16    20    24
     5    10    15    20    25    30
>> B=A(: , 3)
```

Defme a column vector B from the elements in all of the rows of column 3 in matrix A.

Tutorial2-3: Using a colon in addressing arrays. (Continued)

```
B =
     5
     6
     9
    12
    15
>> C=A(2,:)
C =
     2     4     6     8    10    12
>> E=A(2:4,:)
E =
     2     4     6     8    10    12
     3     6     9    12    15    18
     4     8    12    16    20    24
>> F=A(1:3,2:4)
F =
     3     5     7
     4     6     8
     6     9    12
>>
```

Define a row vector C from the elements in all of the columns of row2 in matrix A.

Defme a matrix E from the elements in rows2 through 4 and all the columns in matrix A.

Create a matrix F from the elements in rows 1 through 3 and columns 2 through 4 in matrix A.

In Tutorial2-3 new vectors and matrices are created from existing ones by using a range ofelements, or a range ofrows and columns (using :). It is possible, however, to select only specific elements, or specific rows and columns of existing variables to create new variables. This is done by typing the selected elements or rows or columns inside brackets, as shown below:

```
>> V=4:3:34
V =
     4     7    10    13    16    19    22    25    28    31    34
>> U=V([3 5 7:10])
u =
    10    16    22    25    28    31
>> A=[10:-1:4; cmes(1,7); 2:2:14; zeros(1,7)]
A =
    10     9     8     7     6     5     4
     1     1     1     1     1     1     1
     2     4     6     8    10    12    14
     0     0     0     0     0     0     0
>> B = A([1,3],[1,3,5:7])
```

Create a vector V with 11 elements.

Create a vector u from the 3rd, the 5th, and the 7th through lOth elements ofv.

Create a 4 x 7 matrix A.

Create a matrix B from the 1st and 3rd rows, and 1st, 3rd, and the 5th through 7th columns of A.

```
B =
    10     8     6     5     4
     2     6    10    12    14
```

2.7 ADDING ELEMENTS TO EXISTING VARIABLES

A variable that exists as a vector, or a matrix, can be changed by adding elements to it (remember that a scalar is a vector with one element). A vector (a matrix with a single row or column) can be changed to have more elements, or it can be changed to be a two-dimensional matrix. Rows and/or columns can also be added to an existing matrix to obtain a matrix of different size. The addition of elements can be done by simply assigning values to the additional elements, or by append- ing existing variables.

Adding elements to a vector:

Elements can be added to an existing vector by assigning values to the new ele- ments. For example, if a vector has 4 elements, the vector can be made longer by assigning values to elements 5, 6, and so on. If a vector has n elements and a new value is assigned to an element with an address of n + 2 or larger, MATLAB assigns zeros to the elements that are between the last original element and the new element. Examples:

```
>> DF=1:4                                  Defme vector DF with 4 elements.
DF
     1     2     3     4
>> DF(5:10)=10:5:35                        Adding 6 elements starting with the 5th.
DF
     1     2     3     4    10    15    20    25    30    35
>> AD=[5    7    2]                        Defme vector AD with 3 elements.
AD
     5     7     2
>> AD(8)=4                                 Assign a value to the 8th element.
AD                                         MATLAB assigns zeros to
     5     7     2     0     0     0     0     4   the 4th through 7th elements.
>> AR(5)=24                                Assign a value to the 5th element of a new vector.
AR                                         MATLAB assigns zeros to the
     0     0     0     0    24             1st through 4th elements.
>> =
```

Elements can also be added to a vector by appending existing vectors. Two exam- ples are:

```
>> RE=[3    8  1    24];                  Defme vector RE with 4 elements.
```

```
>> GT=4:3:16;
>> KNB=[RE  GT]
KNB =
     3     8     1    24     4     7    10    13    16
>> KNV=[RE'; GT']
KNV =
     3
     8
     1
    24
     4
     7
    10
    13
    16
```

Defme vector GT with 5 elements.

Defme a new vector KNH by appending RE and GT.

Create a new column vector KNV by appending RE 1 and GT 1.

Adding elements to a matrix:

Rows and/or columns can be added to an existing matrix by assigning values to the new rows or columns. This can be done by assigning new values, or by appending existing variables. This must be done carefully since the size of the added rows or columns must fit the existing matrix. Examples are:

```
>> E=[1 2 3 4; 5 6 7 8]
E =
     1     2     3     4
     5     6     7     8
>> E(3,:)=[10:4:22]
E =
     1     2     3     4
     5     6     7     8
    10    14    18    22
>> K=eye(3)
K =
     1     0     0
     0     1     0
     0     0     1
>> G=[E  K]
G =
     1     2     3     4     1     0     0
     5     6     7     8     0     1     0
    10    14    18    22     0     0     1
```

Defme a 2 x 4 matrix E.

Add the vector 10 14 18 22 as the third row of E.

Defme a 3 x 3 matrix K.

Append matrix K to matrix E. The numbers of rows in E and K must be the same.

If a matrix has a size of m x n and a new value is assigned to an element with an address beyond the size of the matrix, MATLAB increases the size of the matrix to include the new element. Zeros are assigned to the other elements that are added. Examples:

```
>> AW=[3 6 9; 8 5 11]
AW =
     3     6     9
     8     5    11
>> AW(4,5)=17
AW =
     3     6     9     0     0
     8     5    11     0     0
     0     0     0     0     0
     0     0     0     0    17
>> BG(3,4)=15
BG =
     0     0     0     0
     0     0     0     0
     0     0     0    15
>>
```

Defme a 2 x 3 matrix.

Assign a value to the (4,5) element. MATLAB changes the matrix size to 4 x 5, and assigns zeros to the new elements.

Assign a value to the (3,4) element of a new matrix.

MATLAB creates a 3 x 4 matrix and assigns zeros to all the elements except BG(3,4).

2.8 DELETING ELEMENTS

An element, or a range of elements, of an existing variable can be deleted by re- assigning nothing to these elements. This is done by using square brackets with nothing typed in between them. By deleting elements, a vector can be made shorter and a matrix can be made smaller. Examples are:

```
>> kt=[2 8 40 65 3 55 23 15 75 80]
kt =
     2     8    40    65     3    55    23    15    75    80
>> kt(6)=[]
kt =
     2     8    40    65     3    23    15    75    80
>> kt(3:6)=[]
kt =
     2     8    15    75    80
>> mtr=[5 78 4 24 9; 4 0 36 60 12; 56 13 5 89 3]
```

Defme a vector with 10 elements.

Eliminate the 6th element.

The vector now has 9 elements.

Eliminate elements 3 through 6.

The vector now has 5 elements.

Define a 3 x 5 matrix.

```
mtr =
     5    78     4    24     9
     4     0    36    60    12
    56    13     5    89     3
>> mtr(:,2:4) = []
mtr =
     5     9
     4    12
    56     3
>>
```

Eliminate all the rows of columns 2 through 4.

2.9 BUILT-IN FUNCTIONS FOR HANDLING ARRAYS

MATLAB has many built-in functions for managing and handling arrays. Some of these are listed below:

Table 2-2: Built-in functions for handling arrays

Function	Description	Example
length(A)	Returns the number of elements in the vector A.	>> A=[5 9 2 41 ; >> length(A) ans - 4
size(A)	Returns a row vector [m,n], where m and n are the size m x n of the array A.	>> A=[6 1 4 o 12; s 19 6 8 2] A - 6 1 4 0 12 5 19 6 8 2 >> size(A) ans - 2 5
reshape(A, m,n)	Creates a m by n matrix from the elements of matrix A. The elements are taken column after column. Matrix A must have m times n elements.	>> A=[5 1 6; 8 0 2] A = 5 1 6 8 0 2 >> B = reshape(A,3,2) B - 5 0 8 6 1 2

Table 2-2: Built-in functions for handling arrays (Continued)

Function	Description	Example
diag(v)	When v is a vector, creates a square matrix with the elements of v in the diagonal.	>> V=[7 4 2]; >> A=diag(v) A = 7 0 0 0 4 0 0 0 2
diag(A)	When A is a matrix, creates a vector from the diagonal elements of A.	>> A=[1 2 3; 4 5 6; 7 8 9] A = 1 2 3 4 5 6 7 8 9 >> vec=diag(A) vee = 1 5 9

Additional built-in functions for manipulation of arrays are described in the Help Window. In this window, select "MATLAB," then in the Contents "Func- tions," and then "By Category."

Sample Problem 2-1: Create a matrix

Using the ones and zeros commands, create a 4 x 5 matrix in which the ftrst two rows are Os and the next two rows are 1s.

Solution

First, create a 2 x 5 matrix with Os.

```
>> A(1:2,:)=zeros(2,5)
A =
     0     0     0     0     0
     0     0     0     0     0
```

Add rows 3 and 4 with 1s.

```
>> A(3:4,:)=ones(2,5)
A =
     0     0     0     0     0
     0     0     0     0     0
     1     1     1     1     1
     1     1     1     1     1
```

A different solution to the problem is:

```
>> A=[zeros{2,5);ones{2,5)] A
=
     0     0     0     0     0
     0     0     0     0     0
     1     1     1     1     1
     1     1     1     1     1
```

Create a 4 x 5 matrix from two 2 x 5 matrices.

Sample Problem 2-2: Create a matrix

Create a 6 x 6 matrix in which the middle two rows and the middle two columns are 1s and the rest of the entries are Os.

Solution

```
>> AR=zeros{6,6)
AR =
     0     0     0     0     0     0
     0     0     0     0     0     0
     0     0     0     0     0     0
     0     0     0     0     0     0
     0     0     0     0     0     0
     0     0     0     0     0     0
```

First, create a 6 x 6 matrix with 0s.

```
>> AR{3:4,:)=ones{2,6)
AR =
     0     0     0     0     0     0
     0     0     0     0     0     0
     1     1     1     1     1     1
     1     1     1     1     1     1
     0     0     0     0     0     0
     0     0     0     0     0     0
```

Reassign the number 1 to the 3rd and 4th rows.

```
>> AR{:,3:4)=ones{6,2)
AR =
     0     0     1     1     0     0
     0     0     1     1     0     0
     1     1     1     1     1     1
     1     1     1     1     1     1
     0     0     1     1     0     0
     0     0     1     1     0     0
```

Reassign the number 1 to the 3rd and 4th columns.

Sample Problem 2-3: Matrix manipulation

Given are a 5 x 6 matrix A, a 3 x 6 matrix B, and a 9-element vector v.

$$A = \begin{bmatrix} 2 & 5 & 8 & 11 & 14 & 17 \\ 3 & 6 & 9 & 12 & 15 & 18 \\ 4 & 7 & 10 & 13 & 16 & 19 \\ 5 & 8 & 11 & 14 & 17 & 20 \\ 6 & 9 & 12 & 15 & 18 & 21 \end{bmatrix} \qquad B = \begin{bmatrix} 5 & 10 & 15 & 20 & 25 & 30 \\ 30 & 35 & 40 & 45 & 50 & 55 \\ 55 & 60 & 65 & 70 & 75 & 80 \end{bmatrix}$$

$$v = \begin{bmatrix} 99 & 98 & 97 & 96 & 95 & 94 & 93 & 92 \end{bmatrix}$$

Create the three arays in the Command Window, and then, by writing one command, replace the last four columns of the ftrst and third rows of A with the ftrst four columns of the ftrst two rows of B, the last four columns of the fourth row of A with the elements 5 through 8 of v, and the last four columns of the ftfth row of A with columns 3 through 5 of the third row of B.

Solution

```
>> A=[2:3:17; 3:3:18; 4:3:19; 5:3:20; 6:3:21]
A =
     2     5     8    11    14    17
     3     6     9    12    15    18
     4     7    10    13    16    19
     5     8    11    14    17    20
     6     9    12    15    18    21
>> B=[5:5:30; 30:5:55; 55:5:80]
B =
     5    10    15    20    25    30
    30    35    40    45    50    55
    55    60    65    70    75    80
>> V=[99:-1:91]
v =
    99    98    97    96    95    94    93    92    91
>> A([1 3 4 5],3:6)=[B([1 2],1:4); v(5:8); B(3,2:5)]
```

4 x 4 matrix made of columns 3 through 6 of rows 1, 3, 4, and 5.

4 x 4 matrix. The ftrst two rows are columns 1 through 4 of rows 1 and 2 of matrix B. The third row consists of elements 5 through 8 of vector v. The fourth row consists of columns 2 through 5 of row 3 of matrix B.

```
A =
     2     5     5    10    15    20
     3     6     9    12    15    18
     4     7    30    35    40    45
     5     8    95    94    93    92
     6     9    60    65    70    75
```

2.10 *STRINGS* AND *STRINGS* AS *VARIABLES*

- A string is an aray of characters. It is created by typing the characters within single quotes.
- Strings can include letters, digits, other symbols, and spaces.
- Examples of strings: 'ad ef', '3%fr2', '{edcba:21!','MATLAB'.
- A string that contains a single quote is created by typing two single quotes within the string.
- When a string is being typed in, the color of the text on the screen changes to maroon when the first single quote is typed. When the single quote at the end of the string is typed, the color of the string changes to purple.

Strings have several different uses in MATLAB. They are used in output commands to display text messages (Chapter 4), in formatting commands of plots (Chapter 5), and as input arguments of some functions (Chapter 7). More details are given in these chapters when strings are used for these purposes.

- When strings are being used in formatting plots (labels to axes, title, and text notes), characters within the string can be formatted to have a specified font, size, position (uppercase, lowercase), color, etc. See Chapter 5 for details.

Strings can also be assigned to variables by simply typing the string on the right side of the assignment operator, as shown in the examples below:

```
>> a='FRty 8'
a =
FRty 8
>> B='My name is John Smith'
B =
My name is John Smith
>>
```

When a variable is defmed as a string, the characters of the string are stored in an array just as numbers are. Each character, including a space, is an element in the array. This means that a one-line string is a row vector in which the number of elements is equal to the number of characters. The elements of the vectors are

addressed by position. For example, in the vector B that was defmed above the 4th element is the letter n, the 12th element is J, and so on.

```
>> B{4)
ans = n
>> B{12)
ans
J =
```

As with a vector that contains numbers, it is also possible to change specific elements by addressing them directly. For example, in the vector B above the name John can be changed to Bill by:

```
>> B{12:15)='Bill'
B =
My name is Bill Smith
>>
```

Using a colon to assign new characters to elements 12 through 15 in the vector B.

Strings can also be placed in a matrix. As with numbers, this is done by typ- ing a semicolon ; (or pressing the Enter key) at the end of each row. Each row must be typed as a string, which means that it must be enclosed in single quotes. In addition, as with a numerical matrix, all rows must have the same number of elements. This requirement can cause problems when the intention is to create rows with specific wording. Rows can be made to have the same number of ele- ments by adding spaces.

MATLAB has a built-in function named **char** that creates an aray with rows having the same number of characters from an input of rows not all of the same length. MATLAB makes the length of all the rows equal to that of the lon- gest row by adding spaces at the end of the short lines. In the **char** function, the rows are entered as strings separated by a comma according to the following for- mat:

```
variable_name =char (1string 11 , 1string 21 , 1string 31)
```

For example:

```
>> Info=char{'Student Name:','John Smith','Grade:','A+')
Info = Student
Name: John
Smith Grade:
A+
>>
```

A variable named **Info** is assigned four rows of strings, each with different length.

The function **char** creates an array with four rows with the same length as the longest row by adding empty spaces to the shorter lines.

A variable can be defined as either a number or a string made up of the same digits. For example, as shown below, x is defined to be the number 536, and y is defined to be a string made up of the digits 536.

```
>> X=536
X =
    536
>> y='536'
y =
536
>>
```

The two variables are not the same even though they appear identical on the screen. Note that the characters 536 in the line below the x= are indented, while the characters 536 in the line below the y= are not indented. The variable x can be used in mathematical expressions, whereas the variable y cannot.

2.11 PROBLEMS

1. Create a row vector that has the following elements: 8, 10/4, 12 x 1.4, 5 1, tan85°, Ji6, and 0.15 .

2. Create a row vector that has the following elements: $\sqrt{15}$ x 103, $\frac{25}{14\_62}$, ln35/0.43, $\frac{\sin 65°}{\cos 80°}$, 129, and cos2(1t/20) .

3. Create a column vector that has the following elements: 25 5 $\frac{(14\tan 5\ 8^{o})}{(2.12+11)}$, 6!, 2.74, 0.0375, and 1t/S.

4. Create a column vector that has the following elements: $\frac{32}{3\_22}$, sin235°, 6.1, ln292, 0.00552, In229, and 133.

5. Define the variables x – 0.85 , y – 12.5 , and then use them to create a col- umn vector that has the following elements: y , yx , In (yIx) , x x y , and x+y.

6. Define the variables a – 3.5 , b – -6.4, and then use them to create a row vector that has the following elements: a , a2, a/b , a . b, and JQ. .

7. Create a row vector in which the ftrst element is 1 and the last element is 43, with an increment of 6 between the elements (1, 7, 13, … , 43).

8. Create a row vector with 11 equally spaced elements in which the ftrst ele- ment is96 and the last element is 2.

9. Create a column vector in which the ftrst element is 26, the elements decrease with increments of -3. 6, and the last element is -10. (A column vector can be created by the transpose of a row vector.)

10. Create a column vector with9 equally spaced elements in which the ftrst ele- ment is -34 and the last element is -7. (A column vector can be created by the transpose of a row vector.)

11. Using the colon symbol, create a row vector (assign it to a variable named Fives) with ftve elements that are all 5.

12. Using the linspace command, create a row vector (assign it to a variable named Nines) with nine elements that are all9.

13. Use a single command to create a row vector (assign it to a variable named a) with 6 elements such that the last element is 4. 7 and the rest of the elements are Os. Do not type the vector elements explicitly.

14. Use a single command to create a row vector (assign it to a variable named b) with 8 elements such that the last three element are 3. 8 and the rest of the ele- ments are Os. Do not type the vector elements explicitly.

15. Use a single command to create a row vector (assign it to a variable named b) with 11 elements such that

b =

0 2 4 6 8 10 12 9 6 3 0

Do not type the vector explicitly.

16. Create two row vectors: a=2:3:17 andb=3:4:15. Then, by only using the name of the vectors (a and b), create a row vector c that is made from the ele- ments of a followed by the elements ofb.

17. Create two column vectors: a= [2:3:17] ' and b= [3:4:15] ' . Then, by only using the name of the vectors (a and b), create a column vector c that is made from the elements of a followed by the elements ofb.

18. Create a vector (name it vtA) that has 10 elements of which the ftrst is 8, the increment is 7, and the last element is 71. Then, assign elements ofvtA to a new vector (call it vtB) such that vtB has 7 elements. The ftrst 4 elements are the ftrst 4 elements of the vector vtA,and the last 3 are the last 3 elements of the vector vtA. Do not type the elements ofvtA vector explicitly.

19. Create a vector (name it vetC) that has 12 elements of which the ftrst is 5, the increment is 4 and the last element is 49. Then, by assigning elements of vctC to new vectors, create the following two vectors:
    (a) A vector (name it Codd) that contains all the elements with odd index of vctC; i.e., Codd – 5 13 21 ..... 45.
    (b) A vector (name it Ceven) that contains all the elements with even index ofvct;i.e.,Ceven – 9 17 25 ..... 49.
    In both parts use vectors of odd and even numbers for the index of Codd and Ceven,respectively. Do not type the elements of the vectors explicitly.

20. Create a vector (name it vctD) that has 9 elements of which the ftrst is 0, the increment is 3 and the last element is 27. Then create a vector (name it vet- Dop) that consist of the elements ofvctD in reverse order. Do it by assigning elements ofvctD to vctDop. (Do not type the elements ofvctDop vector explicitly.)

21. Create the following matrix by using vector notation for creating vectors with constant spacing and/or the linspace command. Do not type individual elements explicitly.

$$A = \begin{bmatrix} 130 & 110 & 90 & 70 & 50 & 30 & 10 \\ 1 & 2.8333 & 4.6667 & 6.5 & 8.3333 & 10.1667 & 12 \\ 12 & 22 & 32 & 42 & 52 & 62 & 72 \end{bmatrix}$$

22. Create the following matrix by using vector notation for creating vectors with the linspace command. Do not type individual elements explicitly.

$$B = \begin{bmatrix} 5 & 2 & 3 \\ 5 & 2 & 3 \\ 5 & 2 & 3 \\ 5 & 2 & 3 \end{bmatrix}$$

23. Create the following matrix by typing one command. Do not type individual elements explicitly.

$$C = \begin{bmatrix} 7 & 7 & 7 & 7 & 7 \\ 7 & 7 & 7 & 7 & 7 \end{bmatrix}$$

24. Create the following matrix by typing one command. Do not type individual elements explicitly.

$$D = \begin{bmatrix} & & & 8 \\ & & & \\ & & & \\ 0 & 0 & 0 & 6 \end{bmatrix}$$

25. Create the following matrix by typing one command. Do not type individual elements explicitly.

$$E = \begin{bmatrix} & & & & \\ & & & & \\ 0 & 0 & 2 & & \end{bmatrix}$$

26. Create the following matrix by typing one command. Do not type individual elements explicitly.

$$F = \begin{bmatrix} & & & \\ & & & \\ 0 & 0 & 3 & \end{bmatrix}$$

27. Create three row vectors:

$a = [3\ -1\ 5\ 11\ -4\ 2]$, $b = [7\ -9\ 2\ 13\ 1\ -2]$, $c = [-2\ 4\ -7\ 8\ 0\ 9]$

(a) Use the three vectors in a MATLAB command to create a 3×6 matrix in which the rows are the vectors a, b, c, and c.

(b) Use the three vectors in a MATLAB command to create a 6×3 matrix in which the columns are the vectors b, c, and a.

28. Create three row vectors:

$a = [3\ -1\ 5\ 11\ -4\ 2]$, $b = [7\ -9\ 2\ 13\ 1\ -2]$, $c = [-2\ 4\ -7\ 8\ 0\ 9]$

(a) Use the three vectors in a MATLAB command to create a 3×4 matrix such that the first, second, and third rows consist of the last four elements of the vectors a, b, and c, respectively.

(b) Use the three vectors in a MATLAB command to create a 3×3 matrix such that the first, second, and third columns consist of the first three elements of the vectors a, b, and c, respectively.

29. Create two row vectors:

$a = [3\ 9\ -0.5\ 3\ 6\ 15\ -0.8\ \pi]$, $b = [12\ -0.8\ 6\ 2\ 5\ 3\ -7.4]$

(a) Use the two vectors in a MATLAB command to create a 3×4 matrix such that the first row consists of elements 3 through 6 of vector a, the

second row consists of elements 4 through 7 of vector a, and the third row consists of elements 2 through 5 of vector b.

(b) Use the two vectors in a MATLAB command to create a 6×2 matrix such that the first column consists of elements 2 through 7 of vector a, and the second column consists of elements 1 through 3 and 5 through 7 of vector b.

30. By hand (pencil and paper) write what will be displayed if the following commands are executed by MATLAB. Check your answers by executing the commands with MATLAB. (Parts (b), (c), (d), and (e) use the vector that was defmed in part (a).)

(a) a=1:4:17 (b)b= [a(1 3)a] (c) c= [a;a]

(d)d=[a'a']' (e) e=[[a; a; a; a; a]a']

31. The following vector is defined in MATLAB:

$$v = [6\ 11\ -4_5\ 8\ 1\ -0.2_{-7}\ 19_5]$$

By hand (pencil and paper) write what will be displayed if the following commands are executed by MATLAB. Check your answers by executing the commands with MATLAB.

(a) a=v(3:8) (b)b=v([1,3,2: 7,4,6]) (c)c=v([9,1,5,4])'

32. The following vector is defined in MATLAB:

$$v = [6\ 11\ -4_5\ 8\ 1\ -0.2_{-7}\ 19_5]$$

By hand (pencil and paper) write what will be displayed if the following commands are executed by MATLAB. Check your answers by executing the commands with MATLAB.

(a) a=[v([1:3 7:-1:5]) v([10,1,4:6,2])]

(b) b=[v([9,2:4,1])'v([53102 7])'v([10:-2:4,10])']

33. Create the following matrix A.

$$A = \begin{bmatrix} 36 & 34 & 32 & 30 & 28 & 26 \\ 24 & 22 & 20 & 18 & 16 & 14 \\ 12 & 10 & 8 & 6 & 4 & 2 \end{bmatrix}$$

By writing one command and using the colon to address range of elements (do not type individual elements explicitly), use the matrix A to:

(a) Create a six-element row vector named ha that contains the elements of the second row of A.

(b) Create a three-element column vector named hb that contains the elements of the sixth column of A.

(c) Create a five-element row vector named he that contains the first two elements of the third row of A and the last three element of the first row of A.

34. Create the following vector A.

A = [1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18]

Then using the MATLAB's built-in reshape function create the following matrix B from the vector A:

$$B = \begin{bmatrix} 1 & 4 & 7 & 10 & 13 & 16 \\ 2 & 5 & 8 & 11 & 14 & 17 \\ 3 & 6 & 9 & 12 & 15 & 18 \end{bmatrix}$$

By writing one command and using the colon to address range of elements (do not type individual elements explicitly), use the matrix B to:

(a) Create a nine-element column vector named Ba that contains the ele- ments of the first, third, and fifth columns of B.

(b) Create a seven-element row vector named Bb that contains elements 2 through 5 of the second row of B and the elements of the third column of B.

(c) Create a six-element row vector named Be that contains elements 3 through 5 of the first row, and elements 2 through 4 of the third row of B.

35. Create the following vector C.

C = [1.5 2 2.5 3 3.5 4 4.5 5 9.6 9.1 8.6 8.1 7.6 7.1 6.6 6.1]

Then use MATLAB's built-in reshape function and the transpose operation to create the following matrix D from the vector C:

$$D = \begin{bmatrix} 1.5 & 2 & 2.5 & 3 \\ 3.5 & 4 & 4.5 & 5 \\ 9.6 & 9.1 & 8.6 & 8.1 \\ 7.6 & 7.1 & 6.6 & 6.1 \end{bmatrix}$$

By writing one command and using the colon to address a range of elements (do not type individual elements explicitly), use the matrix D to:

(a) Create a eight-element column vector named Da that contains the ele- ments of the first and third rows of D.

(b) Create an eight-element raw vector named Db that contains the elements of the second and the fourth columns of D.

(c) Create a eight-element row vector named De that contains the first two elements of the first row, the last three elements of the second column, and the first three elements of the fourth row of D.

36. Create the following matrix E:

$$E = \begin{bmatrix} 10 & 5 & 5 & 5 & 5 & 5 \\ 0.1 & 0.3 & 0.5 & 0.7 & 0.7 & 0.9 \\ 12 & 9 & 6 & 3 & 0 & -3 \\ 6 & 7 & 8 & 9 & 10 & 11 \end{bmatrix}$$

(a) Create a 2×3 matrix F from the second and third rows, and the third

through the fifth columns of matrixE.

(b) Create a 4 x 4 matrix G from all rows and the third through sixth columns of matrixE.

37. Create the following matrix H:

$$H = \begin{bmatrix} 1.25 & 1.5 & 1.75 & 2 & 2.25 & 2.5 & 2.75 \\ 1 & 2 & 3 & 1 & 2 & 3 & 4 \\ 45 & 40 & 35 & 30 & 25 & 20 & 15 \end{bmatrix}$$

(a) Create a 2 x 5 matrix G such that its first row includes the first three ele- ments and the last two elements of the first row of H, and the second row of G includes the last five elements of the third row of H.

(b) Create a 4 x 3 matrix K such that the first, second, third, and fourth rows are the second, third, fifth and seventh columns of matrix H.

38. The following matrix is defined in MATLAB:

$$A\{= \begin{bmatrix} 1 & 4 & 7 & 10 & 13 & 16 \\ 2 & 5 & 8 & 11 & 14 & 17 \\ 3 & 6 & 9 & 12 & 15 & 18 \end{bmatrix}$$

By hand (pencil and paper) write what will be displayed if the following com- mands are executed by MATLAB. Check your answers by executing the com- mands with MATLAB.

a) A=M([113]I [11516])
b) B=M(:1[414:6])
c) C=M([112]I:)
d) D=M([2I3] I[2I3])

39. The following matrix is defined in MATLAB:

N = [[illegible]
6 [illegible]4 23 35 4[illegible]
8 16 [illegible]]

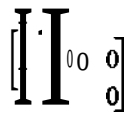
By hand (pencil and paper) write what will be displayed if the following com- mands are executed by MATLAB. Check your answers by executing the com- mands with MATLAB.

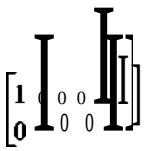
(a) A=[N(111: 4) 1IN(2I2:5) 1]
(b) B=[N(:I3) , N(31:)]
(c) C(3:415:6) =N(2:314:5)

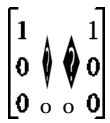
40. By hand (pencil and paper) write what will be displayed if the following com- mands are executed by MATLAB. Check your answers by executing the com- mands with MATLAB.

```
V=1:2:23
M=reshape(v,3,4)
M(2,:)=[]
M(:,3)=[]
N=ones(size(M))
```

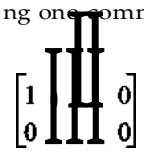
41. Using the zeros, ones, and eye commands, create the following arays by typing one command:

(a) 

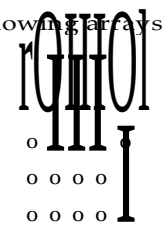
(b) 

(c) 

42. Using the zeros, ones, and eye commands create the following arrays by typing one command:

(a) 

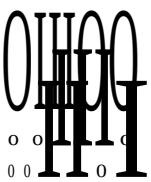
(b) 

(c) 

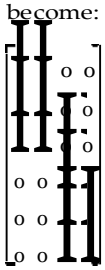
43. Use the eye, ones, and zeros commands to create the following arrays:

A=  B=  C= 

Using the variables A, B, and C, write a command that creates the following matrix D:

D= 

44. Create a 2 x 3 matrix A in which all the elements are 1. Then reassign A to itself(several times) such that Awill become:

A= 

Chapter3

Mathematical Operations with Arrays

Once variables are created in MATLAB they can be used in a wide variety of mathematical operations. In Chapter 1 the variables that were used in mathematical operations were all defined as scalars. This means that they were all 1 x 1 arrays(arays with one row and one column that have only one element) and the mathematical operations were done with single numbers. Arrays,however,can be one-dimensional (arrays with one row, or with one column), two-dimensional (arays with multiple rows and columns), and even of higher dimensions. In these cases the mathematical operations are more complex. MATLAB, as its name indicates,is designed to carry out advanced array operations that have many applications in science and engineering. This chapter presents the basic, most common mathematical operations that MATLAB performs using arrays.

Addition and subtraction are relatively simple operations and are covered first, in Section 3.1. The other basic operations-multiplication, division, and exponentiation--can be done in MATLAB in two different ways. One way, which uses the standard symbols(*,/, and ^), follows the rules of linear algebra and is presented in Sections 3.2 and 3.3. The second way, which is called element-by-element operations,is covered in Section 3.4. These operations use the symbols .*, ./,and .^(a period is typed in front of the standard operation symbol). In addition, in both types of calculations,MATLAB has left division operators(\ or\), which are also explained in Sections 3.3 and 3.4.

A Note to First-Time Users ofMATLAB:

Although matrix operations are presented first and element-by-element operations next, the order can be reversed since the two are independent of each other. It is expected that almost every MATLAB user has some knowledge of matrix operations and linear algebra, and thus will be able to follow the material covered in Sections 3.2 and 3.3 without any difficulty. Some readers, however, might prefer to read Section 3.4 first. MATLAB can be used with element-by-element operations in numerous applications that do not require linear algebra multiplication(or division) operations.

3.1 ADDITIONAND SUBTRACTION

The operations + (addition) and - (subtraction) can be used to add (subtract) arays of identical size (the same numbers of rows and columns) and to add (sub- tract) a scalar to an array. When two arrays are involved the sum, or the difference, of the arays is obtained by adding, or subtracting, their corresponding elements.

In general, if A and B are two arrays (for example, 2 x 3 matrices),

$$A = \begin{bmatrix} Au & A12 & A_{13} \\ A21 & A22 & A_{23} \end{bmatrix} \quad \text{and} \quad B = \begin{bmatrix} Bu & B12 & B_{13} \\ B21 & B22 & B_{23} \end{bmatrix}$$

then the matrix that is obtained by adding A and B is:

$$\begin{bmatrix} (Au + Bu) & (A12 + B12) & (A13 + B_{13}) \\ (A21 + B21) & (A22 + B22) & (A23 + B_{23}) \end{bmatrix}$$

Examples are:

```
>> VectA= [8 5 41; VectB=[10 2 7];
>>  VectC=VectA+VectB
VectC =
    18     7    11
>> A=[5   -3  8;  9  2  10]
A =
     5    -3     8
     9     2    10
>> B=[10   7  4;  -11   15   1]
B =
    10     7     4
   -11    15     1
>> A-B
ans =
     -5   -10     4
     20   -13     9
>> C=A+B
c =
    15     4    12
    -2    17    11
>> VectA+A
??? Error using  ==> plus
Matrix dimensions must agree.
>>
```

Defme two vectors.

Define a vector VectC that is equal to VectA + VectB.

Define two 2 x 3 matrices A and B.

Subtracting matrix B from matrix A.

Defme a matrix C that is equal to A + B.

Trying to add arrays of different size.

An error message is displayed.

When a scalar (number) is added to (or subtracted from) an array, the scalar is added to (or subtracted from) all the elements ofthe array. Examples are:

```
>> VectA=[1  5  8  -10  2]
VectA  =
     1     5     8   -10     2
>> VectA+4
ans  =
     5     9    12    -6     6
>> A=[6 21  -15;  0  -4  8]
A  =
     6    21   -15
     0    -4     8
>> A-5
ans  =
     1    16   -20
    -5    -9     3
```

Defme a vector named VectA.

Add the scalar 4 to VectA.

4 is added to each element ofVectA.

Defme a 2 x 3 matrix A.

Subtract the scalar 5 from A.

5 is subtracted from each element ofA.

3.2 ARRAYMULTIPLICATION

The multiplication operation * is executed by MATLAB according to the rules of linear algebra. This means that if A and B are two matrices, the operation A*B can be caried out only ifthe number ofcolumns in matrix A is equal to the number of rows in matrix B. The result is a matrix that has the same number ofrows as A and the same number of columns as B. For example, if A is a 4 x 3 matrix and B is a 3 x 2 matrix:

$$A = \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \\ A_{41} & A_{42} & A_{43} \end{bmatrix} \quad \text{and} \quad B = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \\ B_{31} & B_{32} \end{bmatrix}$$

then the matrix that is obtained with the operation A*B has dimensions 4 x 2 with the elements:

(A11B11 +A12B21 +A13B31) (A11B12 +A12B22 +A13B32)
(A21B11 +A22B21 +A23B3,) (A21B12 +A22B22 +A23B32)
(A31B11 +A32B21 +A33B3,) (A31B12 +A32B22 +A33B32)
(A41B11 +A42B21 +A43B3,) (A41B12 +A42B22 +A43B32)

A numerical example is:

$$\begin{bmatrix} 4 & 1 \\ 2 & 6 \\ 5 & 3 \\ 2 & 6 \end{bmatrix} \begin{bmatrix} 1 & 4 \\ 2 & 6 \\ 3 & 2 \end{bmatrix} = \begin{bmatrix} (4.1+1.2+3.3) & (4.4+1.6+3.2) \\ (2.1+6.2+8.3) & (2.4+6.6+8.2) \\ (5.1+3.2+2.3) & (5.4+3.6+2.2) \\ (2.1+6.2+8.3) & (2.4+6.6+8.2) \end{bmatrix} = \begin{bmatrix} 15 & 34 \\ 43 & 74 \\ 17 & 42 \\ 43 & 74 \end{bmatrix}$$

The product ofthe multiplication oftwo square matrices (they must be ofthe same size) is a square matrix ofthe same size. However, the multiplication of matrices is not commutative. This means that ifA and B are both n x n, then A*B t:. B*A. Also, the power operation can be executed only with a square matrix (since A*A can be carried out only ifthe number ofcolumns in the first matrix is equal to the number ofrows in the second matrix).

Two vectors can be multiplied only ifthey have the same number ofelements, and one is a row vector and the other is a column vector. The multiplication ofa row vector by a column vector gives a 1 x 1 matrix, which is a scalar. This is the dot product oftwo vectors. (MATLAB also has a built-in function, dot (a,b), that computes the dot product oftwo vectors.) When using the dot function, the vectors a and b can each be a row vector or a column vector (see Table3-1). The multiplication ofa column vector by a row vector, each with n elements, gives an n x n matrix. Multiplication ofarray is demonstrated in Tutorial3-1,

Tutorial 3-1: Multiplication of arrays.

```
>> A=[1 4 2; 5 7 3; 9 1 6; 4 2 8]
A =
     1     4     2
     5     7     3
     9     1     6
     4     2     8
>> B=[6 1; 2 5; 7 3]
B =
     6     1
     2     5
     7     3
>> C=A*B
c =
    28    27
    65    49
    98    32
    84    38
>> D=B*A
??? Error using ==> *
Inner matrix dimensions must agree.

>> F=[1 3; 5 7]
F =
     1     3
     5     7
>> G=[4 2; 1 6]
```

Defme a 4 x3 matrix A.

Define a 3 x 2 matrix B.

Multiply matrix A by matrix B and assign the result to variable C.

Trying to multiply B by A, B*A, gives an error since the number ofcolumns in B is 2 and the number of rows in A is 4.

Defme two 2 x 2 matrices F and G.

Tutorial3-1: Multiplication of arrays. (Continued)

```
G =
     4     2
     1     6
>> F*G                                   Multiply F*G
ans =
     7    20
    27    52
>> G*F                                   Multiply G*F
ans =
    14    26                            Note that the answer for G*F is not the
    31    45                            same as the answer for F*G
>> AV=[2  5  1]                         Defme a three-element row vector AV.
AV =
     2     5     1
>> BV=[3;  1;  4]                        Define a three-element column vector BV.
BV =
     3
     1
     4
>> AV*BV                                 Multiply AV by BV. The answer is a scalar.
ans =                                    (Dot product of two vectors.)
    15
>> BV*AV                                 Multiply BV by AV. The
ans =                                    answer is a 3 x 3 matrix.
     6    15     3
     2     5     1
     8    20     4
>>
```

When an array is multiplied by a number (actually a number is a 1 x 1 array), each element in the array is multiplied by the number. For example:

```
>> A=[2  5  7  0;  10  1  3  4;  6  2  11  5]   Defme a 3 x 4 matrix A.
A =
     2     5     7     0
    10     1     3     4
     6     2    11     5
>> b=3                                   Assign the number 3 to the variable b.
b =
     3
```

```
>> b*A

ans =
     6    15    21     0
    30     3     9    12
    18     6    33    15

>> C=A*S

C =
    10    25    35     0
    50     5    15    20
    30    10    55    25
```

Multiply the matrix A by b. This can be done by either typing b*A or A*b.

Multiply the matrix A by 5 and assign the result to a new variable C. (Typing C = 5*A gives the same result.)

Linear algebra rules of array multiplication provide a convenient way for writing a system of linear equations. For example, the system of three equations with three unknowns

$$A11x1 + A12x2 + A13x3 - B1$$

$$Azixi + AzzXz + Az3X3 - Bz\ A31x1 +$$

$$A32x2 + A33X3 - B3$$

can be written in a matrix form as

and in matrix notation as $\begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} B_1 \\ B_2 \\ B_3 \end{bmatrix}$

$$AX = B \quad \text{where } A = \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix}, X = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}, \text{and } B = \begin{bmatrix} B_1 \\ B_2 \\ B_3 \end{bmatrix}.$$

3.3 ARRAYDIVISION

The division operation is also associated with the rules of linear algebra. This operation is more complex, and only a brief explanation is given below. A full explanation can be found in books on linear algebra.

The division operation can be explained with the help of the identity matrix and the inverse operation.

Identity matrix:

The identity matrix is a square matrix in which the diagonal elements are 1s and the rest of the elements are Os. As was shown in Section 2.2.1, an identity matrix can be created in MATLAB with the eye command. When the identity matrix multiplies another matrix (or vector), that matrix (or vector) is unchanged (the

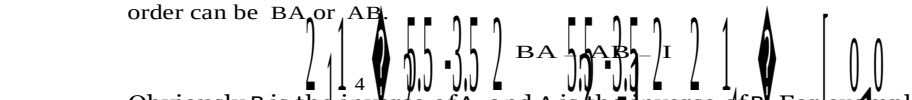
multiplication has to be done according to the rules of linear algebra). This is equivalent to multiplying a scalar by 1. For example:

If a matrix A is square, it can be multiplied by the identity matrix, I, from the left or from the right:

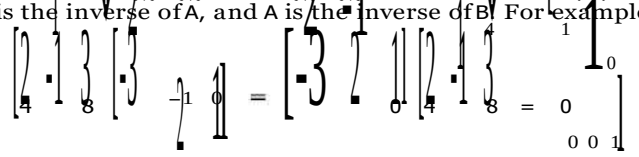
$$AI = IA = A$$

Inverse of a matrix:

The matrix B is the inverse of the matrix A if, when the two matrices are multi- plied, the product is the identity matrix. Both matrices must be square, and the multiplication order can be BA or AB.



Obviously B is the inverse of A, and A is the inverse of B. For example:



The inverse of a matrix A is typically written as A^{-1}. In MATLAB the inverse of a matrix can be obtained either by raising A to the power of -1, A^{-1}, or with the inv (A) function. Multiplying the matrices above with MATLAB is shown below.

```
>> A=[2  1  4;  4  1  8;  2  -1  3]
A =
     2     1     4
     4     1     8
     2    -1     3
```

Creating the matrix A.

```
>> B=inv(A)
B =
    5.5000   -3.5000    2.0000
    2.0000   -1.0000         0
   -3.0000    2.0000   -1.0000
```

Use the inv function to find the inverse of A and assign it to B.

```
>> A*B
ans =
     1     0     0
     0     1     0
     0     0     1
```

Multiplication of A and B gives the identity matrix.

```
>> A*A^-1
ans =
     1     0     0
     0     1     0
     0     0     1
```

Use the power -1 to fmd the inverse of A. Multiplying it by A gives the identity matrix.

Not every matrix has an inverse. A matrix has an inverse only if it is square and its determinant is not equal to zero.

Determinants:

A determinant is a function associated with square matrices. A short review on determinants is given below. For a more detailed coverage refer to books on linear algebra.

The determinant is a function that associates with each square matrix A a number, called the determinant of the matrix. The determinant is typically denoted by det(A) or !AI. The determinant is calculated according to specific rules. For a second-order 2 x 2 matrix, the rule is:

$$|A| = \begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix} = a_{11}a_{22}-a_{12}a_{21}, \quad \text{for example,} \quad \begin{vmatrix} 6 & 5 \\ 9 & 3 \end{vmatrix} = 39$$

The determinant of a square matrix can be calculated with the det command (see Table 3-1).

Array division:

MATLAB has two types of aray division, right division and left division.

Left division, \ :

Left division is used to solve the matrix equation AX= B. In this equation X and B are column vectors. This equation can be solved by multiplying, on the left, both sides by the inverse of A:

$$A^{-1}AX = I_1B$$

The left-hand side of this equation is X, since

$$I^{1}AX = IX = X$$

So the solution of AX= B is:

$$X = I^{1}B$$

In MATLAB the last equation can be written by using the left division character:

$$X = A\backslash B$$

It should be pointed out here that although the last two operations appear to give the same result, the method by which MATLAB calculates X is different. In the first, MATLAB calculates I^{1} and then uses it to multiply B. In the second (left division), the solution X is obtained numerically using a method that is based on Gauss elimination. The left division method is recommended for solving a set of

linear equations, because the calculation of the inverse may be less accurate than the Gauss elimination method when large matrices are involved.

Right division, I :

The right division is used to solve the matrix equation XC= D. In this equation X and D are row vectors. This equation can be solved by multiplying, on the right, both sides by the inverse of C:

$$x.\ cc\text{-}^{1} \_ n\ .\ c\text{-}^{1}$$

which gives

$$X = D\cdot\ c\text{-}^{1}$$

In MATLAB the last equation can be written using the right division character:

$$X=DIC$$

The following example demonstrates the use of the left and right division, and the inv function to solve a set of linear equations.

Sample Problem 3-1: Solving three linear equations (array division)

Use matrix operations to solve the following system of linear equations.

$$4x-2y+6z=8$$
$$2x+8y+2z-4$$
$$6x+10y+3z\_0$$

Solution

Using the rules of linear algebra demonstrated earlier, the above system of equa- tions can be written in the matrix form AX = B or in the form XC= D:

$$\begin{bmatrix} 4 & -2 & 6 \\ 2 & 8 & 2 \\ 6 & 10 & 3 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 8 \\ 4 \\ 0 \end{bmatrix} \quad \text{or} \quad \begin{bmatrix} x\ y\ z \end{bmatrix} \begin{bmatrix} 4 & 2 & 6 \\ -2 & 8 & 10 \\ 6 & 2 & 3 \end{bmatrix} = \begin{bmatrix} 8\ 4\ 0 \end{bmatrix}$$

Solutions for both forms are shown below:

```
>> A=[4 -2 6; 2 8 2; 6 10 3];
>> B=[8; 4; 0];
>> X=A\B
X =
    -1.8049
     0.2927
     2.6341
>> Xb=inv(A)*B
Xb =
    -1.8049
     0.2927
     2.6341
```

Solving the form AX=B.

Solving by using left division: X=A\B.

Solving by using the inverse of A: X = £1 *B.*

```
>> C=[4  2  6;  -2  8  10;  6  2  3];
>> D=[8  4  0];
>> Xc=D/C
    1.8049      0.2927      2.6341
>> Xd=D*inv(C)
   -1.8049      0.2927      2.6341
```

Solving the form XC= D.

Solving by using right division: X= Xc D/C.

Solving by using the inverse of C: X = D Xd C⁻¹.

3.4 ELEMENT-BY-ELEMENT OPERATIONS

In Sections 3.2 and 3.3 it was shown that when the regular symbols for multiplica- tion and division(* and /) are used with arrays, the mathematical operations fol- low the rules of linear algebra. There are, however, many situations that require element-by-element operations. These operations are caried out on each of the elements of the aray (or arrays). Addition and subtraction are by definition already element-by-element operations, since when two arrays are added(or sub- tracted) the operation is executed with the elements that are in the same position in the arrays. Element-by-element operations can be done only with arays of the same siZe.

Element-by-element multiplication, division, or exponentiation of two vectors or matrices is entered in MATLAB by typing a period in front of the arithmetic operator.

Symbol	Description	Symbol	Description
.*	Multiplication	./	Right division
.^	Exponentiation	.\	Left Division

If two vectors a and b are a= [a, az a 3a] and b= [b, hz b3 b], then element-by-element multiplication, division, and exponentiation of the two vec- tors gives:

$$a.*b = [a_1b_1\ a_2b_2\ a_3b_3\ a_4b]$$

$$a./b = [a_1/b_1\ a_2/b_2\ a_3/b_3\ a_4/b]$$

$$a.^{\wedge}b = [(a_1)^{b_1}\ (a_2)^{b_2}\ (a_3)^{b_3}\ (a_4)^{b}]$$

If two matrices A and B are

$$A = \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix} \quad \text{and} \quad B = \begin{bmatrix} B_{11} & B_{12} & B_{13} \\ B_{21} & B_{22} & B_{23} \\ B_{31} & B_{32} & B_{33} \end{bmatrix}$$

then element-by-element multiplication and division of the two matrices give:

$$A .* B = \begin{bmatrix} A_{11}B_{11} & A_{12}B_{12} & A_{13}B_{13} \\ A_{21}B_{21} & A_{22}B_{22} & A_{23}B_{23} \\ A_{31}B_{31} & A_{32}B_{32} & A_{33}B_{33} \end{bmatrix} \qquad A ./ B = \begin{bmatrix} A_{11}/B_{11} & A12/B12 & A13/B13 \\ A_{21}/B_{21} & AzzlBzz & Az31Bz3 \\ A_{31}/B_{31} & A32IB32 & A33IB33 \end{bmatrix}$$

Element-by-element exponentiation of matrix A gives:

$$A .\hat{} n = \begin{bmatrix} (A_{11})^n & (A_{12})^n & (A_{13})^n \\ (A_{21})^n & (A_{22})^n & (A_{23})^n \\ (A_{31})^n & (A_{32})^n & (A_{33})^n \end{bmatrix}$$

Element-by-element multiplication, division, and exponentiation are dem- onstrated in Tutorial3-2.

Tutorial3-2: Element-by-element operations.

```
>> A=[2  6  3;  5  8  4]
A =
     2     6     3
     5     8     4
>> B=[1  4  10;  3  2  7]
B =
     1     4    10
     3     2     7
>> A.*B
ans =
     2    24    30
    15    16    28
>> C=A./B
c =
    2.0000    1.5000    0.3000
    1.6667    4.0000    0.571 4
```

Defme a 2 x3 array A.

Defme a 2 x3 array B.

Element-by-element multiplication of array A by B.

Element-by-element division of array A by B. The result is assigned to variable C.

Tutorial3-2: Element-by-element operations. (Continued)

```
>> B.^3

ans =
     1    64  1000
    27     8   343

>> A*B

??? Error using ==> *
Inner matrix dimensions must agree
```

Element-by-element exponentiation of array B. The result is an array in which each term is the corresponding term in B raised to the power of 3.

Trying to multiply A*B gives an error, since A and B cannot be multiplied according to linear algebra rules. (The number of columns in A is not equal to the number of rows in B.)

Element-by-element calculations are very useful for calculating the value of a function at many values of its argument. This is done by first defming a vector that contains values of the independent variable, and then using this vector in element-by-element computations to create a vector in which each element is the corresponding value of the function. One example is:

```
>> X=[1:8]
X =
     1     2     3     4     5     6     7     8
>> y=x.^2-4*x
y =
    -3    -4    -3     0     5    12    21    32
>>
```

Create a vector x with eight elements.

Vector x is used in element-by-element calculations of the elements of vector y.

In the example above $y = x^2 - 4x$. Element-by-element operation is needed when x is squared. Each element in the vector y is the value of y that is obtained when the value of the corresponding element of the vector x is substituted in the equation. Another example is:

```
>> Z=(1:2:11]
z =
     1     3     5     7     9    11
>> y=(z.^3 + 5*z)./(4*z.^2 - 10)

y
   -1.0000    1.6154    1.6667    2.0323    2.4650    2.9241
```

Create a vector z with six elements.

Vector z is used in element-by-element calculations of the elements of vector y.

In the last example $y = \frac{z^3 + 5z}{4z^2 - 10}$. Element-by-element operations are used in this example three times: to calculate z^3 and z^2, and to divide the numerator by the denominator.

3.5 USING ARRAYS IN MATLAB BUILT-IN MATH FUNCTIONS

The built-in functions in MATLAB are written such that when the argument (input) is an array, the operation that is defmed by the function is executed on each element of the array. (One can think of the operation as element-by-element appli- cation of the function.) The result (output) from such an operation is an array in which each element is calculated by entering the corresponding element of the argument (input) array into the function. For example, if a vector with seven ele- ments is substituted in the function cos (x), the result is a vector with seven ele- ments in which each element is the cosine of the corresponding element in x. This is shown below.

```
>> X= (0 :pi/6 :pi)
X =
     0    0.5236    1.0472    1.5708    2.0944    2.6180    3.1416
>>y=cos(x)

   1.0000    0.8660    0.5000    0.0000   -0.5000   -0.8660   -1.0000
>>
```

```
d =
     1     4     9
    16    25    36
    49    64    81
>> h=sqrt(d)
h =  1     2     3
     4     5     6
     7     8     9
```

h is a 3 x 3 aray in which each element is the square root of the corresponding element in array d.

The feature ofMATLAB in which arrays can be used as arguments in functions is called vectorization.

3.6 BUILT-IN FUNCTIONS FOR ANAL¥ZING ARRAYS

MATLAB has many built-in functions for analyzing arrays. Table 3-1 lists some of these functions.

Table 3-1: Built-in array functions

Function	Description	Example
mean(A)	If A is a vector, returns the mean value of the elements of the vector.	>> A=[5 9 2 41; >> mean(A) ans = 5
C=max(A) [d, n] =max(A)	If A is a vector, cis the larg- est element in A. If A is a matrix, C is a row vector containing the largest ele- ment of each column of A. If A is a vector, dis the larg- est element in A, and n is the position of the element (the first if several have the max value).	>> A=[5 9 2 4 11 6 11 11; >> C=max(A) C = 11 >> [d,n]=max(A) d = 11 n = 5
min(A) [d, n] =min(A)	The same as max(A), but for the smallest element. The same as [d, n] = max(A), but for the smallest element.	>> A=[5 9 2 4]; >> min(A) ans = 2
sum(A)	If A is a vector, returns the sum of the elements of the vector.	>> A=[5 9 2 4]; >> sum(A) ans = 20
sort(A)	If A is a vector, arranges the elements of the vector in ascending order.	>> A=[5 9 2 41; >> sort(A) ans = 2 4 5 9
median(A)	If A is a vector, returns the median value of the elements of the vector.	>> A= [5 9 2 41; >> median(A) ans = 4.5000

Table 3-1: Built-in array functions (Continued)

Function	Description	Example
std(A)	If A is a vector, returns the standard deviation of the elements of the vector.	>> A=[5 9 2 4]; >> std(A) ans = 2.9439
det(A)	Returns the determinant of a square matrix A.	>> A=[2 4; 3 5]; >> det(A) ans = -2
dot(a,b)	Calculates the scalar (dot) product of two vectors a and b. The vectors can each be row or column vectors.	>> a=[1 2 3]; >> b=[3 4 5]; >> dot(a,b) ans = 26
cross(a,b)	Calculates the cross product of two vectors a and b, (axb). The two vectors must have each three elements.	>> a=[1 3 2]; >> b=[2 4 1]; >> cross(a,b) ans = -5 3 -2
inv(A)	Returns the inverse of a square matrix A.	>> A=[2 -2 1; 3 2 -1; 2 -3 2]; >> inv(A) ans = 0.2000 0.2000 0 -1.6000 0.4000 1.0000 -2.6000 0.4000 2.0000

3.7 GENERATION OF RANDOM NUMBERS

Simulations of many physical processes and engineering applications frequently require using a number (or a set of numbers) with a random value. MATLAB has three commands-rand, randn, and randi-that can be used to assign ran- dom numbers to variables.

The rand command:

The rand command generates uniformly distributed random numbers with val- ues between 0 and 1. The command can be used to assign these numbers to a sca- lar, a vector, or a matrix, as shown in Table 3-2.

Table 3-2: The rand command

Command	Description	Example
rand	Generates a single random number between 0 and 1.	>> rand ans = 0.2311
rand(l,n)	Generates an n-element row vector of random numbers between 0 and 1.	>> a=rand(1,4) a = 0.6068 0.4860 0.8913 0.7621
rand(n)	Generates an n x n matrix with random numbers between 0 and 1.	>> b=rand(3) b = 0.4565 0.4447 0.9218 0.0185 0.6154 0.7382 0.8214 0.7919 0.1763
rand(m,n)	Generates an m x n matrix with random numbers between 0 and 1.	>> c=rand(2,4) c = 0.4057 0.9169 0.8936 0.3529 0.9355 0.4103 0.0579 0.8132
randperm(n)	Generates a row vector with n elements that are random permutation of integers 1 through n.	>> randperm(8) ans = 8 2 7 4 3 6 5 1

Sometimes there is a need for random numbers that are distributed in an inter- val other than (0, 1), or for numbers that are integers only. This can be done using mathematical operations with the rand function. Random numbers that are dis- tributed in a range (a,b) can be obtained by multiplying rand by (b- a) and add- ing the product to a:

$$(b- a)*rand +a$$

For example, a vector of 10 elements with random values between -5 and 10 can be created by (a= -5, b = 10):

```
>> v=15*rand(1,10)-5 v
=
    -1.8640       0.6973       6.7499       5.2127       1.9164       3.5174
 6.9132      -4.1123       4.0430      -4.2460
```

The randi command:

The randi command generates uniformly distributed random integer. The com- mand can be used to assign these numbers to a scalar, a vector, or a matrix, as shown in Table 3-3.

Table 3-3: The randi command

Command	Description	Example
randi(imax) (imax is an integer)	Generates a single random number between 1 and 1max.	>> a=randi(15) a = 9
randi(imax, n)	Generates an n x n matrix with random integers between 1 and imax.	>> b=randi(15,3) b = 4 8 11 14 3 8 1 15 8
randi(imax, m,n)	Generates an m x n matrix with random integers between 1 and imax.	>> c=randi(15,2,4) c = 1 1 8 13 11 2 2 13

The range of the random integers can be set to be between any two integers by typing [imin imax] instead of imax. For example, a 3 x 4 matrix with ran- dom integers between 50 and 90 is created by:

```
>> d=randi( [50  90],3,4)
d =
     82    71    75
57
66   52    67    61
84   66    76    67
```

The randn command:

The randn command generates normally distributed numbers with mean 0 and standard deviation of 1. The command can be used to generate a single number, a vector, or a matrix in the same way as the rand command. For example, a 3 x 4 matrix is created by:

```
>> d=randn(3,4)
d
    -0.4326    0.2877    1.1892    0.1746
= -1.6656   -1.1465   -0.0376   -0.1867
     0.1253    1.1909    0.3273    0.7258
```

The mean and standard deviation of the numbers can be changed by mathematical operations to have any values. This is done by multiplying the number generated by the randn function by the desired standard deviation, and adding the desired mean. For example, a vector of six numbers with a mean of 50 and standard devi-

ation of 6 is generated by:

```
>> v=4*randn(1,6)+50
v =
   42.7785    57.4344    47.5819    50.4134    52.2527    50.4544
```

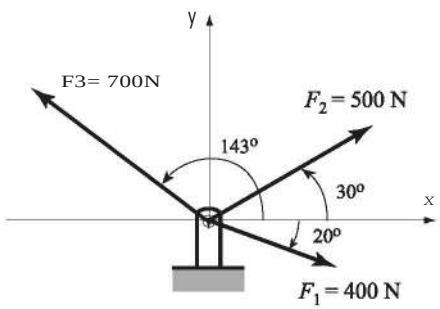
Integers of normally distributed numbers can be obtained by using the round function.

```
>> w=round(4*randn(1,6)+50)
w =
    51     49     46     49     50     44
```

3.8 EXAMPLES OF MATLAB APPLICATIONS

Sample Problem 3-2: Equivalent force system (addition of vectors)

Three forces are applied to a bracket as shown. Determine the total (equivalent) force applied to the bracket.



Solution

A force is a vector (a physical quantity that has a magnitude and direction). In a Cartesian coordinate system a two-dimensional vector F can be written as:

$$F = F_x i + F_Y j = F\cos\theta i + F\sin\theta j = F(\cos\theta i + \sin\theta j)$$

where F is the magnitude of the force and e is its angle relative to the x axis, Fx and F_y are the components ofF in the directions of the x andy axes, respectively, and i and j are unit vectors in these directions. If Fx and F_y are known, then F and e can be determined by:

$$F = \sqrt{F_x^2 + F_y^2} \quad \text{and} \quad \tan\theta = \frac{F_y}{F_x}$$

The total (equivalent) force applied on the bracket is obtained by adding the forces that are acting on the bracket. The MATLAB solution below follows three steps:

- Write each force as a vector with two elements, where the first element is the x component of the vector and the second element is they component.
- Determine the vector form of the equivalent force by adding the vectors.
- Determine the magnitude and direction of the equivalent force.

The problem is solved in the following script file.

```
% Sample Problem 3-2 solution {script file)
clear
FIM=400;  F2M=500;  F3M=700;

Th1=-20;  Th2=30;  Th3=143;

F1=F1M*[cosd{Th1)  sind{Th1)]
F2=F2M*[cosd{Th2)   sind{Th2)
F3=F3M*[cosd{Th3)  sind{Th3)]
Ftot=F1+F2+F3
FtotM=sqrt{Ftot{1)A2+Ftot{2)A2)
Th=atand{Ftot{2)/Ftot{1))
```

Defmevariables with the magnitudeof ea ch vector.

Define variables with the angle of each vector.

Defmethethree vectors.

Calculate the total force vector.

Cal culatethemagnitu deof the total for cevector.

Calculate the angleof thetotal for ce vector.

When the program is executed, the following is displayed in the Command Window:

```
F1  = 375.8770
F2  = 433.0127   -136.8081
F3  =
  -559.0449       250.0000
Ftot  =
    249.8449      534.4625
FtotM  = 589.9768
Th
     64.9453
```

The components ofF1.

The components ofF2.

The components ofF3.

The components of thetotal

Themagnitu deof thetotal for ce

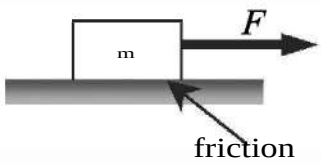
direction of thetotal for ce in degrees.

The equivalent for ce has a magnitude of 589.98 N, and is directed 64.95° (c cw) relativeto thex a xis. In vector notation, thefor ceis F = 249.84i + 534.46j N.

Sample Problem 3-3: Friction experiment (element-by-element calculations)

The coefficient of friction, Jl, can be determined in an experiment by measuring the force F required to move a mass m. When F is measured and m is known, the coefficient of friction can be calculated by:

ll = F/(mg) (g=9.81m/s2).



Results from measuring F in six tests are given in the table below. Determine the coefficient of friction in each test, and the average from all tests.

Test	1	2	3	4	5	6
Mass m (kg)	2	4	5	10	20	50
ForceF(N)	12.5	23.5	30	61	117	294

Solution

A solution using MATLAB commands in the Command Window is shown below.

```
>> m=[2 4 5 10 20 50];
>> F=[12.5 23.5 30 61 117 294];
>> mu=F./(m*9.81)

mu

    0.6371    0.5989    0.6116    0.6218    0.5963    0.5994

>> mu_ave=mean(mu)

mu ave =
    0.6109
```

the values ofm in a vector.

Enter

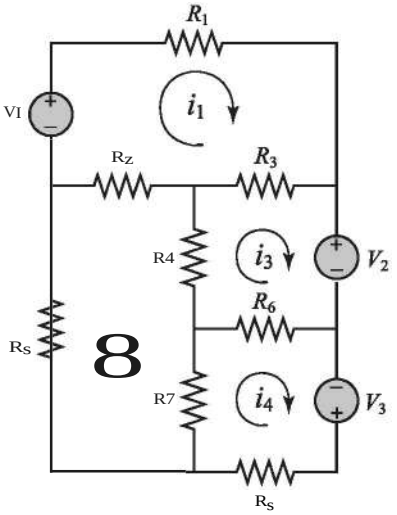
Enter the values of F in a vector.

A value for mu is calculated for each test, using element-by-element calculations.

The average of the elements in the vectormu is determined by using the functionmean.

Sample Problem 3-4: Electrical resistive network analysis (solving a system of linear equations)

The electrical circuit shown consists of resistors and voltage sources. Determine the curent in each resistor using the mesh current method, which is based on Kirchhoff's voltage law.



$V1 = 20\,V,\ V2 = 12\,V,\ V3 = 40\,V$
$R1 = 18Q,\ R2 = 10Q,\ R3 = 16Q$
$R4 = 6Q,\ R5 = 15Q,\ R6 = 8Q$
$R7 = 12Q,\ R8 = 14Q$

Solution

Kirchhoff's voltage law states that the sum of the voltage around a closed circuit is zero. In the mesh current method a current is ftrst assigned for each mesh (ii. iz, i3, i4 in the figure). Then Kirchhoff's voltage law is applied for each mesh. This results in a system of linear equations for the currents (in this case four equations). The solution gives the values of the mesh curents. The current in a resistor that belongs to two meshes is the sum of the currents in the corresponding meshes. It is convenient to assume that all the currents are in the same direction (clockwise in this case). In the equation for each mesh, the voltage source is positive if the cur- rent flows to the – pole, and the voltage of a resistor is negative for curent in the direction of the mesh curent.

The equations for the four meshes in the curent problem are:

$$V1-R1i1-R3(i1-i3)-R2(i1-i2) = 0$$
$$-R5i2-R2(i2-i1)-R4(i2-i3)-R7(i2-i4) = 0$$
$$-V2-R6(i3-i4)-R4(i3-i2)-R3(i3-i1) = 0$$
$$V3-R8i4-R7(i4-i2)-R6(i4-i3) = 0$$

The four equations can be rewritten in matrix form [A][x] = [B]:

il

$$\begin{bmatrix} -(R1+Rz+R3) & Rz & R3 & 0 \\ Rz & -(Rz+R4+Rs+R7) & R4 & R7 \\ R3 & R4 & -(R3+R4+R6) & R6 \\ 0 & R7 & R6 & -(R6+R7+R8) \end{bmatrix} \begin{bmatrix} iz \\ i3 \\ i4 \end{bmatrix} = \begin{bmatrix} \vdots \\ Vz \\ -V3 \end{bmatrix}$$

The problem is solved in the following program, written in a script file:

```
V1=20;  V2=12;  V3=40;
R1=18;  R2=10;  R3=16;  R4=6;
R5=15;  R6=8;  R7=12;  R8=14;
A=[-{R1+R2+R3)  R2  R3  0 R2
-{R2+R4+R5+R7)  R4  R7 R3  R4
-{R3+R4+R6)  R6
0  R7  R6  -{R6+R7+R8)]
>> B=[-V1;  0;  V2;  -V3]
>> I=A\B
```

Defme variables with the values of the V's and R's.

Create the matrix A.

Create the vector B.

Solve for the currents by using left division.

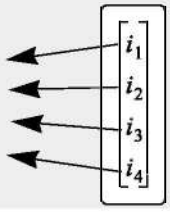
When the script file is executed, the following is displayed in the Command Win- dow:

```
A
     -44      10      16       0
=     10     -43       6      12
      16       6     -30       8
       0      12       8     -34
B =
     -20
       0
      12
     -40
I =
      0.8411
      0.7206
      0.6127
      1.5750
>>
```

The numerical value of the matrix A.

The numerical value of the vector B.

The solution.

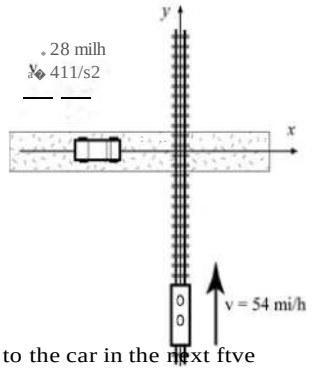


The last column vector gives the current in each mesh. The curents in the resis- tors R1, R5, and R8 are i1 – 0.8411 A, i2 – 0.7206 A, and i4 – 1.5750 A, respec- tively. The other resistors belong to two meshes and their current is the sum of the currents in the meshes.

The current in resistor R2 is i1- i2 – 0.1205 A. The current in resistor R3 is i1- i3 – 0.2284 A. The current in resistor R4 is i2- i3 – 0.1079 A. The current in resistor R6 is i4- i3 – 0.9623 A. The current in resistor R1 is i4- i2 – 0.8544 A.

Sample Problem 3-5: Motion of two particles

A train and a car are approaching a road crossing. At time t = 0 the train is 400 ft south of the crossing traveling north at a constant speed of 54 mi/h. At the same time the car is 200ft west of the crossing trav- eling east at a speed of 28 milh and accelerating at 4 ftls2• Determine the positions of the train and the car, the distance between them, and the speed of the train relative to the car every second for the next 10 sec- onds.



To show the results, create an 11 x 6 matrix in which each row has the time in the ftrst column and the train position, car position, distance between the train and the car, car speed, and the speed of the train relative to the car in the next ftve columns, respectively.

Solution

The position of an object that moves along a straight line at a constant acceleration is given by $s = s0+v0t+\frac{1}{2}at^2$ where s0 and v0 are the position and velocity at t = 0, and a is the acceleration. Applying this equation to the train and the car gives:

$$y = -400+v_{train}t \qquad \text{(train)}$$

$$x = -200+v_{car}t+\frac{1}{2}a_{cart}t^2 \qquad \text{(car)}$$

The distance between the car and the train is: $d = \sqrt{x^2+y^2}$ • The velocity of the train is constant and in vector notation is given by $v_{train} = v_{train}i$. The car is accelerating and its velocity at time tis given by $v_{car} = (v_{car} + a_{cart})i$. The veloc- ity of the train relative to the car, $v_{t/c}$, is given by $v_{t/c} = v_{train}-v_{car} = -(v_{car}+a_{cart})i+v_{train}j$. The magnitude (speed) of this velocity is the length of the vector.

The problem is solved in the following program, written in a script ftle. First a vector t with 11 elements for the time from 0 to 10 s is created, then the positions of the train and the car, the distance between them, and the speed of the train rela- tive to the car at each time element are calculated.

```
v0train=54*5280/3600;   v0car=28*5280/3600;   acar=4;
```

Create variables for the initial velocities (in ftls) and the acceleration.

```
t=O: 10;
```

the vector t.

```
Y=-400+v0train*t;
```

Calculate the train and car positions.

```
x=-200+v0car*t+O. S*acar*t. "2;
```

the between the train and car.

Create

```
d=sqrt(x.^2+y.^2);
```

Calculate distance

```
vcar=vOcar+acar*t;                                  Calculatethe car's velocity.
speed_trainRcar=sqrt(vcar.A2+v0trainA2);
                                  Calculate the speed of the train relative to the car.
table=[t' y' x• d' vcar' speed _trainRcar']
                                        Create a table (see note below).
```

Note: Inthe commands above, table isthe nru.ne of the variablethat is a ma ix containingthe data to be displayed.

When the script file is executed, the following is displayed inthe Command Window:

```
table =
         0   -400.0000   -200.0000   447.2136   41.0667   89.2139
    1.0000   -320.8000   -156.9333   357.1284   45.0667   91.1243
    2.0000   -241.6000   -109.8667   265.4077   49.0667   93.1675
    3.0000   -162.4000    -58.8000   172.7171   53.0667   95.3347
    4.0000    -83.2000     -3.7333    83.2837   57.0667   97.6178
    5.0000     -4.0000     55.3333    55.4777   61.0667  100.0089
    6.0000     75.2000    118.4000   140.2626   65.0667  102.5003
    7.0000    154.4000    185.4667   241.3239   69.0667  105.0849
    8.0000    233.6000    256.5333   346.9558   73.0667  107.7561
    9.0000    312.8000    331.6000   455.8535   77.0667  110.5075
   10.0000    392.0000    410.6667   567.7245   81.0667  113.3333
```

Time (s) | Train position (ft) | Car position (ft) | Car-train distance (ft) | Car speed (ftJs) | Train speed relative to the car (ftJs)

In this problemthe results(numbers) are displayed by MATLAB without any text. Instructions on how to add text to output generated by MATLAB are pre- sented in Chapter 4.

3.9 PROBLEMS

Note: *Additional problems for practicing mathematical operations with arrays are provided at the end of Chapter* 4.

1. For the function $y = x^2 - e^{0.5x} + x$, calculatethe value of y for the following values ofx using element-by-element operations: -3, -2, -1, 0, 1, 2, 3 .

2. For the function $y = \frac{(x + 5)^3}{x^2}$, calculate the value of y for the following values ofx using element-by-element operations: 1, 2, 3, 4, 5, 6.

3. For the function $y = \frac{(x+7)^4}{(x+1)\sqrt{x}}$, calculate the value of y for the following values of x using element-by-element operations: 1.5, 2.5, 3.5, 4.5, 5.5, 6.6.

4. For the function $y = \frac{2\sin x + \cos 2x}{\sin x}$, calculate the value of y for the following values of x using element-by-element operations: 20°, 30°, 40°, 50°, 60°, 70°.

5. The radius, r, of a sphere can be calculated from its surface area, s, by:

$$r = \frac{\sqrt{s}}{2}$$

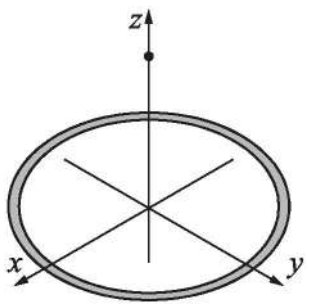
The volume, V, is given by:

$$V = \frac{4\pi r^3}{3}$$

Determine the volume of spheres with surface area of 50, 100, 150, 200, 250, and 300 π. Display the results in a two-column table where the values of s and V are displayed in the first and second columns, respectively.

6. The electric field intensity, E(z), due to a ring of radius R at any point z along the axis of the ring is given by:

$$E(z) = \frac{\rho_l}{2\varepsilon_0}\frac{Rz}{(z^2+R^2)^{3/2}}$$



where ρ_l is the charge density, $\varepsilon_0 = 8.85 \times 10^{-12}$ is the electric constant, and R is the radius of the ring. Consider the case where $\rho_l = 1.7 \times 10^{-7}$ C/m and R = 6 cm.

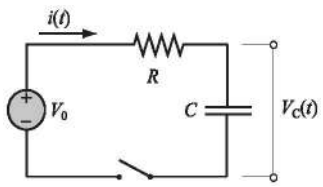
(a) Determine E(z) at z = 0, 2, 4, 6, 8, and 10 cm.

(b) Determine the distance z where E is maximum. Do it by creating a vector z with elements ranging from 2 cm to 6 cm and spacing of 0.01 cm. Calculate E for each value of z and then find the maximum E and associated z with MATLAB's built-in function max.

7. The voltage $v_c(t)$ (in V) and the current i(t) (in Amp) t seconds after closing the switch in the circuit shown are given by:

$$v_c(t) = V_0(1 - e^{-t/\tau_0})$$

$$i(t) = \frac{V_0}{R}e^{-t/\tau_0}$$



where $\tau_0 = RC$ is the time constant. Consider the case where $V_0 = 24$ V, R = 3800 Ω and C = 4000 x 10-6 F. Determine the voltage and the current during the first 20 s after the switch is closed. Create a vector with values of

times from 0 to20 s with spacing of2 s, and use it for calculating Vc(t) and i(t). Display the results in a three-column table where the values of time, voltage and curent are displayed in the first, second, and third columns, respectively.

8. The length lui (magnitude) of a vector $\mathbf{u} = xi+yj+zk$ is given by $|u| = \sqrt{x^2 + \ldots}$ Given the vector $\mathbf{u} = 23.5i - 17j + 6k$, determine its length in the following two ways:
   (a) Define the vector in MATLAB, and then write a mathematical expression that uses the components of the vector.
   (b) Define the vector in MATLAB, then determine the length by writing one command that uses element-by-element operation and MATLAB built-in functions sum and sqrt.

9. A vector w_L of length L in the direction of a vector $\mathbf{u} = xi+yj+zk$ can determined by $w_L = Lu_n$ (multiplying a unit vector in the direction of u by L). The unit vector u_n in the direction of the vector $\mathbf{u}$ is given by $u_n = \frac{xi+yj+zk}{\sqrt{x^2+y^2+z2}}$. By writing one MATLAB comman , determine a vector of length 1 8 in the direction of the vector $u = 7i - 4j - 11k$.

10. The following two vectors are defmed in MATLAB:

$$v = [15,8,-6] \quad u = [3,-2,6]$$

By hand (pencil and paper) write what will be displayed if the following com- mands are executed by MATLAB. Check your answers by executing the com- mands with MATLAB.

(a) v. /u (b) u' *v (c) u*v'

11. Two vectors are given:

$\mathbf{u} = 5i - 6j + 9k$ and $v = 11i + 7j - 4k$

Use MATLAB to calculate the dot product $u \cdot v$ of the vectors in three ways:

(a) Write an expression using element-by-element calculation and the MAT- LAB built-in function sum.
(b) Define u as a row vector and vas a column vector, and then use matrix multiplication.
(c) Use the MATLAB built-in function dot.

12. Defme the vector v = [2 3 4 5 6]. Then use the vector in a mathematical expression to create the following vectors:

(a) a = [4 6 8 10 12] (b) b = [8 27 64 125 216]
(c) c = [22 33 44 5s 66] (d) d = [1 1.5 2 2.5 3]

13. Define the vector v = [8 6 4 2]. Then use the vector in a mathematical expression to create the following vectors:

(a) a = [1 1 1 1; 1 1 1 1]

(b) b = [1 1 1 1] / [√82 √62 √42 √22]

(c) c = [√8 √6 √4 √2]

(d) d = [3 1 -1 -3]

14. Define x and y as the vectors x = [1, 2, 3, 4, 5] and y = [2, 4, 6, 8, 10]. Then use them in the following expressions to calculate z using element-by-element calculations.

(a) $z = \frac{(x+y)^2}{x-y}$

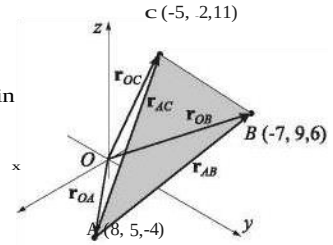
(b) w = xln(x2 + y2) + $\sqrt{\frac{y^3}{(y-x)2}}$

15. Define r and s as scalars r = 1.6 x 10_3 and s = 14.2, and, t, x, and y as vectors t= [1, 2, 3, 4, 5], x= [0, 2, 4, 6, 8], and y= [3, 6, 9, 12, 15]. Then use these variables to calculate the following expressions using element-by-element calculations for the vectors.

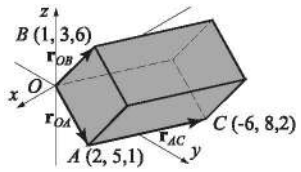
(a) G = xt+ $\frac{r}{2}$(y2-x)t

(b) R = $\frac{r(-xt+yt^2)}{15}$-s2(y-0.5x2)t

16. The area of a triangle ABC can be calculated by |rAB x rAd/2, where rAB and rAe are vectors connecting the vertices A and B and A and C, respectively. Determine the area of the triangle shown in the figure. Use the following steps in a script file to calculate the area. First, defme the vectors roA, rOB and rae from knowing the coordinates of points A, B, and C. Then determine the vectors rAB and rAe from roA, rOB and rae. Finally, determine the area by using MATLAB's built-in functions **cross**, sum. and **sqrt**.



17. The volume of the parallelepiped shown can be calculated by r0B (r0A x rAe). Use the following steps in a script file to calculate the area. Defme the vectors roA, rAe, and rOB from knowing position of points A, B, and C.
Determine the volume by using MATLAB's built-in functions **dot** and **cross**.



18. Define the vectors:

$$u = 5i - 2j + 4k, \quad v = -2i + 7j + 3k, \text{ and } w = 8i + 1j - 3k$$

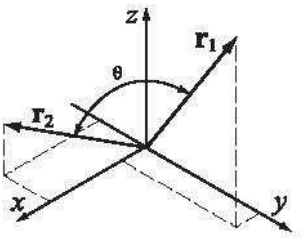
Use the vectors to verify the identity:

$$(u+v) \cdot [(v+w) \times (w+u)] = 2u \cdot (v \times w)$$

Use MATLAB's built-in functions cross and dot, calculate the value of the leftand right sides of the identity.

19. The dot product can be used for determin the angle between two vecton:

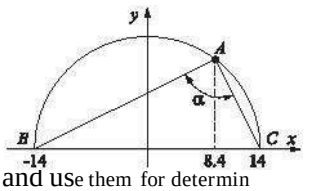
$$\theta = \cos^{-1}\left(\frac{r_1 \cdot r_2}{|r_1||r_2|}\right)$$



Use MATLAB's built-in functions acosd, sqrt, and dot to find the angle (in degres) between $r_1 = 2i + 9j + 10k$ and $r_2 = 6i - 3j + 2k$.

Show that $|r_1| = |r_2|$.

20 Use MAILAB to show that the angle inscribed in a semi-circle is a right angle. Use the follow- ing steps in a script file to calculate the angle. Define a variable with the value of the x coordi- nate of point A. Determine the y coordinate of point A using the equation $x^2 + y^2 = R^2$. Define vectors that con-espond to the position of points A, B, and C and use them for determin position vectors r_{AB} and r_{AC}. Calculate theangle θ in two ways.



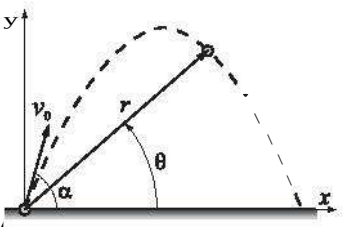
First by using the equation $\theta = \cos^{-1}\left(\frac{r_{AB} \cdot r_{AC}}{|r_{AB}||r_{AC}|}\right)$ and then by using the equation $\theta = \sin^{-1}\left(\frac{|r_{AB} \times r_{AC}|}{|r_{AB}||r_{AC}|}\right)$. Both should give 90°.

21. The position as a function of time (x(t),y(t)) of a projectile fired with a speed of v0 at an angle θ is given by

$$x(t) = v_0 \cos\theta \; t \qquad y(t) = v_0 \sin\theta \; t - \frac{1}{2} g t^2$$

where $g = 9.81 \text{ m/s}^2$. The polar coordinates of the projectile at time t are $(r(t), \theta(t))$, where

$$r(t) = \sqrt{x(t)^2 + y(t)^2} \quad \text{and} \quad \tan\theta(t) = \frac{y(t)}{x(t)}$$



Consider the case where $v_0 = 162 \text{ m/s}$ and $\theta = 70°$. Determine r(t) and $\theta(t)$ for $t = 1, 6, 11, \ldots, 31$s.

22. Use MATLAB to show that the sum of the infinite series $\sum_{n=0}^{\infty} \frac{2^n}{n!}$ converges to e^2. Do this by computing the sum for:

(a) n = 5, (b) n = 10, (c) n = 50

For each part create a vector n in which the first element is 0, the increment is 1 and the last term is 5, 10, or 50. Then use element-by-element calculations to create a vector in which the elements are $\frac{2^n}{n!}$. Finally, use MATLAB's built-in function sum to sum the series. Compare the values to e^2 (use format long to display the numbers).

23. Use MATLAB to show that the sum of the infmite series $\sum_{n=1}^{\infty} \frac{(9/10)^n}{n}$ converges to ln10. Do this by computing the sum for

(a) n = 10, (b) n = 50, (c) n = 100

For each part, create a vector n in which the frrst element is 1, the increment is 1 and the last term is 10, 50 or 100. Then use element-by-element calculations to create a vector in which the elements are $\frac{(9/10)^n}{n}$. Finally, use MATLAB's built-in function sum to sum the series. Compare the values to ln10 (use format long to display the numbers).

24. According to Zeno's paradox any object in motion must arrive at the halfway point before it can arrive at its destination. Once arriving at the halfway point, the remaining distance is once again divided in half and so on to infmity. Since it is impossible to complete this process, Zeno concluded all motion must be an illusion. Letting the length be unity, Zeno's paradox can be written in terms of the infinite sum $\sum_{n=1}^{\infty} \frac{1}{2^n}$. To see how quickly this series converges to 1, compute the sum for:

(a) n = 5, (b) n = 10, (c) n = 40

For each part create a vector n in which the first element is 1, the increment is 1, and the last term is 5, 10, or 40. Then use element-by-element calculations to create a vector in which the elements are $\frac{1}{2^n}$. Finally, use the MATLAB built-in function sum to add the terms of the series. Compare the values obtained in parts (a), (b), and (c) with the value of 1.

25. Show that $\lim_{x \to 0} \frac{\cos(2x)-1}{\cos x - 1} = 4$.
Do this by first creating a vector x that has the elements 1.0, 0.5, 0.1, 0.01, 0.001, and 0.0001. Then, create a new vectoryin which each element is determined from the elements of x by $\frac{\cos(2x)-1}{\cos x - 1}$. Compare the elements of y with the value 4 (use format long to display the numbers).

26. Show that $\lim_{x \to 1} \frac{x^{113}-1}{x^{114}-1} = \frac{4}{3}$.
Do this by first creating a vector x that has the elements 2.0, 1.5, 1.1, 1.01, 1.001, 1.00001, and 1.0000001. Then, create a new vectoryin which each element is determined from the elements of x by $\frac{x^{113}-1}{x^{114}-1}$ Compare the elements ofywith the value 4/3 (use format long to display the numbers).

27. The demand for water during a fire is often the most important factor in the design of distribution storage tanks and pumps. For communities with populations less than 200,000, the demand Q (in gallons/min) can be calculated by:

$$Q = 1020P^{1/2}(1-0.01P^{1/2})$$

where P is the population in thousands. Set up a vector for P that starts at 10 and increments by 10 up to 200. Use element-by -element computations to determine the demand Q for each population in P.

28. The ideal gas equation states that $P = \frac{nRT}{V}$, where P is the pressure, V is the volume, T is the temperature, R = 0.08206 (L atm)/(mol K) is the gas constant, and n is the number of moles. Real gases, especially at high pressure, deviate from this behavior. Their response can be modeled with the van der Waals equation

$$P = \frac{nRT}{V-nb} - \frac{n^2 a}{V^2}$$

where a and b are material constants. Consider 1 mole (n = 1) of nitrogen gas at T = 300K. (For nitrogen gas a = 1.39 $(L^2 atm)/mol^2$, and b = 0.0391 L/mol.) Create a vector with values of Vs for $0.1 \le V \le 1$ L, using increments of 0.02 L. Using this vector calculate P twice for each value of V, once using the ideal gas equation and once with the van der Waals equation. Using the two sets of values for P, calculate the percent of error $\left(\frac{P_{ideal}-P_{waals}}{P_{waals}} 100\right)$ for each value of V. Finally, by using MATLAB's built-in function max, determine the maximum error and the corresponding volume.

29. Create the following three matrices:

$$A=\begin{bmatrix} 1 & -3 & 5 \\ 2 & 2 & 4 \\ -2 & & 6 \end{bmatrix} \qquad B=\begin{bmatrix} 2 & 1 & \\ 5 & 1 & -6 \\ 2 & 7 & -1 \end{bmatrix} \qquad C=\begin{bmatrix} & 4 & -1 \\ 8 & 2 & \\ -3 & 5 & 3 \end{bmatrix}$$

(a) Calculate A+B and B+A to show that addition of matrices is commuta- tive.

(b) Calculate A+(B+C) and (A+B)+C to show that addition of matrices is associative.

(c) Calculate 3(A+C) and 3A+5C to show that, when matrices are multi- plied by a scalar, the multiplication is distributive.

(d) Calculate A*(B+C) and A*B+A*C to show that matrix multiplication is distributive.

30. Use the matrices A, B, and C from the previous problem to answer the follow- ing:

(a) Does A*B= B*A? (b) Does A*(B*C) = (A*B)*C?

(c) Does $(A*B)^t = A^t*B^t$? (^{t} means transpose) (d) Does $(A+B)^t = A^t + B^t$?

31. Create a 4 x 4 matrix A having random integer values between 1 and 10. Call the matrix A and, using MATLAB, perform the following operations. For each part explain the operation.

(a) A* A (b) A.*A (c) A\A

(d) A.IA (e) det (A) (e) inv (A)

32. The magic square is an arrangement of numbers in a square grid in such a way that the sum of the numbers in each row, and in each column, and in each diagonal is the same. MATLAB has a built-in function magic (n) that returns an n x n magic square. In a script file create a (6 x 6) magic square, and then test the properties of the resulting matrix by fmding the sum of the elements in each row, in each column and in both diagonals. In each case, use MATLAB's built-in function sum. (Other functions that can be useful are diag and fliplr.)

33. Solve the following system of three linear equations:

$$-4x+3y+z = -18.2$$
$$5x+6y-2z = -48.8$$
$$2x-5y+4.5z = 92.5$$

34. Solve the following system of five linear equations:

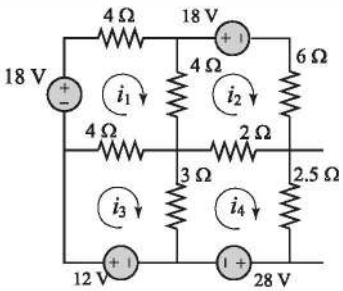
$$2.5a-b+3c+1.5d-2e = 57.1$$
$$3a+4b-2c+2.5d-e = 27.6$$
$$-4a+3b+c-6d+2e = -81.2$$
$$2a+3b+c-2.5d+4e = -22.2$$
$$a+2b+5c-3d+4e = -12.2$$

35. A food company manufactures five types of 8 oz Trail mix packages using different mixtures of peanuts, almonds, walnuts, raisins, and M&Ms. The mixtures have the following compositions:

	Peanuts(oz)	Almonds(oz)	Walnuts(oz)	Raisins(oz)	M&Ms(oz)
Mix 1	3	1	1	2	1
Mix2	1	2	1	3	1
Mix3	1	1	0	3	3
Mix4	2	0	3	1	2
Mix5	1	2	3	0	2

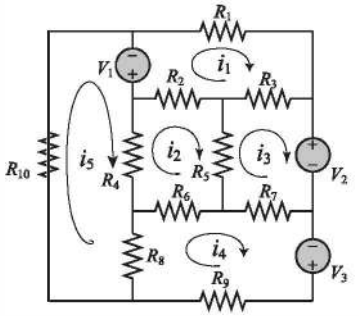
How many packages of each mix can be manufactured if 128 lb of peanuts, 118 lb of almonds, 112 lb of walnuts, 112 lb of raisins, and 104 lb of M&Ms are available? Write a system of linear equations and solve.

36. The electrical circuit shown consists of resistors and voltage sources. Determine i1, i2, i3, and i4, using the mesh current method based on Kirchhoff's voltage law (see Sample Problem 3-4).



37. The electrical circuit shown consists of resistors and voltage sources. Determine i1, i2, i3, i4 and i5, using the mesh curent method based on Kirchhoff's voltage law (see Sample Problem 3-4).

V1 = 40V, V2 = 30V, V3 = 36V,
R1 = 160, R2 = 200, R3 = 100
R4 = 140, R5 = 8 Q, R6 = 16 Q, R1 = 10 Q, R8 = 15 Q, R9 = 6 Q, R_{10} = 4 Q.



Unit 4

Using Script Files and Managing Data

A script file (see Section 1.8) is a list ofMATLAB commands, called a program, that is saved in a file. When the script file is executed (run), MATLAB executes the commands. Section 1.8 describes how to create, save, and run a simple script file in which the commands are executed in the order in which they are listed, and in which all the variables are defmed within the script file. This chapter gives more details about how to input data to a script file, how data is stored in MAT- LAB, various ways to display and save data that is created in script files, and how to exchange data between MATLAB and other applications.

In general, variables can be defmed (created) in several ways. As shown in Chapter 2, variables can be defined implicitly by assigning values to a variable name. Variables can also be assigned values by the output of a function. In addi- tion, variables can be defined with data that is imported from files outside MAT- LAB. Once defmed (either in the Command Window or when a script file is executed), the variables are stored in MATLAB's Workspace.

Variables that reside in the workspace can be displayed in various ways, saved, or exported to applications outside MATLAB. Similarly, data from files outside MATLAB can be imported to the workspace and then used in MATLAB.

Section 4.1 explains how MATLAB stores data in the workspace and how the user can see the data that is stored. Section 4.2 shows how variables that are used in script files can be defined in the Command Window and/or in script files. Section 4.3 shows how to output data that is generated when script files are exe- cuted. Section 4.4 explains how the variables in the workspace can be saved and then retrieved, and Section 4.5 shows how to import and export data from and to applications outside MATLAB.

4.1 THE MATLAB WORKSPACE AND THE WORKSPACE WINDOW

The MATLAB workspace consists of the set of variables (named arrays) that are defmed and stored during a MATLAB session. It includes variables that have been defined in the Command Window and variables defined when script files are executed. This means that the Command Window and script files share the same memory zone within the computer. This implies that once a variable is in the workspace, it is recognized and can be used, and it can be reassigned new values, in both the Command Window and script files. there is another type of file in MATLAB, called a function file, where variables can also be defined. These variables, however, are normally not shared with other parts of the program since they use a separate workspace.

Recall from Chapter 1 that the who command displays a list of the variables currently in the workspace. The whos command displays a list of the variables currently in the workspace and information about their size, bytes, and class. An example is shown below.

```
>> 'variables in memory'
ans =
Variables in memory
>> a = 7;
>> E = 3;
>> d = [5, a+E, 4, E^2]
d=
     5    10     4     9
>> g = [a, a..2, 13; a*E, 1, a..E]
g =
     7    49    13
    21     1   343
>> who
Your variables are:
E   a   ans   d   g
>> whos
  Name        Size              Bytes  Class     Attributes
  E           1x1                   8  double
  a           1x1                   8  double
  ans         1x19                 38  char
  d           1x4                  32  double
  g           2x3                  48  double
>
```

Typing a string.

The string is assigned to ans.

Creating the variables a, E, d, and g.

The who command displays the variables currently in the workspace.

The whos command displays the variables curently in the workspace and information about their size and other information.

The variables curtly in memory can also be viewed in the Workspace Window. This window can be opened by selecting Workspace in the Desktop menu. Figure 4-1 shows the Workspace Window that corresponds to the variables defined above. The variable that are displayed in the Workspace Window can

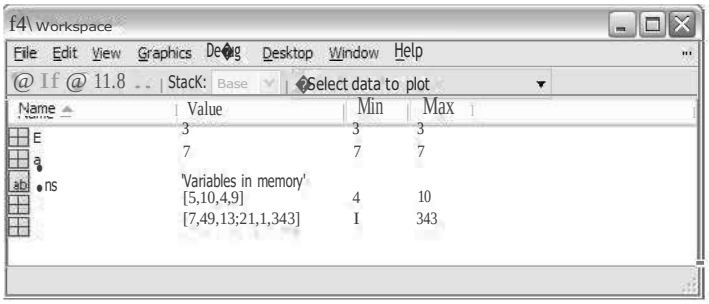


Figure 4-1: The Workspace Window.

also be edited (changed). Double-clicking on a variable opens the Variable Editor Window, where the content of the variable is displayed in a table. For example, Figure 4-2 shows the Variable Editor Window that opens when the variable g in Figure 4-1 is double-clicked.

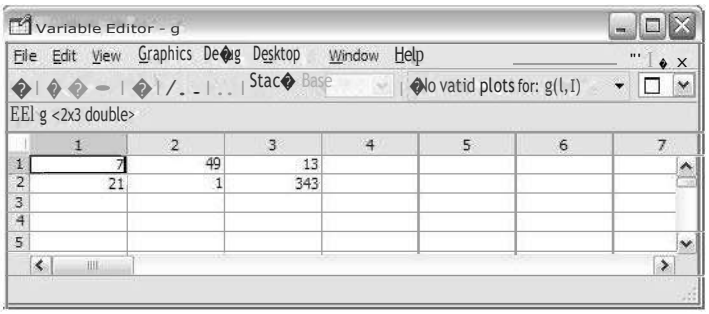


Figure 4-2: The Variable Editor Window.

The elements in the Variable Editor Window can be edited. The variables in the Workspace Window can be deleted by selecting them, and then either pressing the delete key on the keyboard or selecting delete from the edit menu. This has the same effect as entering the command `clear variable_name` in the command Window.

4.2 INPUT TO A SCRIPTFILE

When a script file is executed, the variables that are used in the calculations within the file must have assigned values. In other words, the variables must be in the workspace. The assignment of a value to a variable can be done in three ways, depending on where and how the variable is defined.

1. The variable is defined and assigned a value in the script file.

In this case the assignment of a value to the variable is part of the script file. If the user wants to run the file with a different variable value, the file must be edited and the assignment of the variable changed. Then, after the file is saved, it can be executed again.

The following is an example of such a case. The script file (saved as Chapter4Example2) calculates the average points scored in three games.

```
% This script file calculates the average points scored in three games.
% The assignment of the values of the points is part of the script file. gamel=75;
game2=93;
game3=68;                          ave_point
s=(gamel+game2+game3)/3
```

The variables are assigned values within the script file.

The display in the Command Window when the script file is executed is:

```
>> Chapter4Example2

ave_points 78
    .6667

>>
```

The variable ave_points with its value is displayed in the Command Window.

The script file is executed by typing the name of the file.

2. The variable is defined and assigned a value in the Command Window.

In this case the assignment of a value to the variable is done in the Command Window. (Recall that the variable is recognized in the script file.) If the user wants to run the script file with a different value for the variable, the new value is assigned in the Command Window and the file is executed again.

For the previous example in which the script file has a program that calcu- lates the average of points scored in three games, the script file (saved as Chapter4Example3) is:

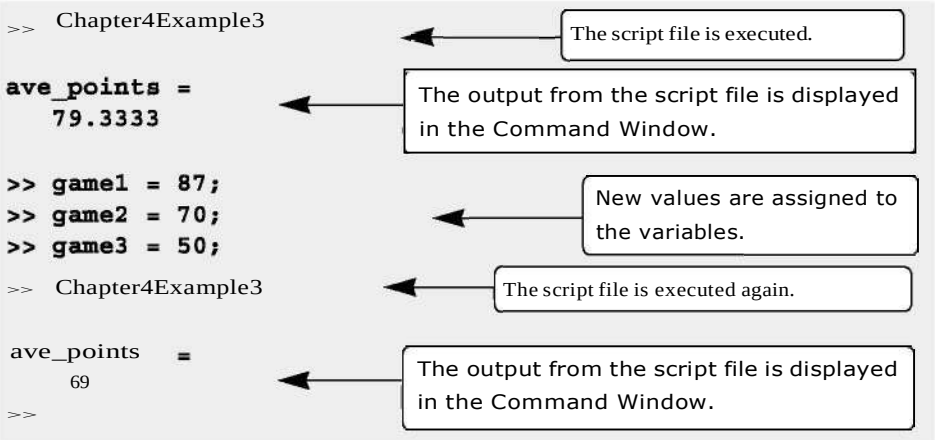
```
% This script file calculates the average points scored in three games.
% The assignment of the values of the points to the variables
% gamel, game2, and game3 is done in the Command Window.

ave_points=(gamel+game2+game3)/3
```

The Command Window for running this file is:

```
>> gamel    67;
>> game2    90;
>> game3    81;
```

The variables are assigned values in the Command Window.

```
=
=
=
```



3. The variable is defined in the script file, but a specific value is entered in the Command Window when the script file is executed.

In this case the variable is defined in the script file, and when the file is executed, the user is prompted to assign a value to the variable in the Command Window. This is done by using the **input** command for creating the variable.

The form of the **input** command is:

```
variable_name =input ('string with a message that
                      is displayed in the Command Window')
```

When the **input** command is executed as the script file runs, the string is dis- played in the Command Window. The string is a message prompting the user to enter a value that is assigned to the variable. The user types the value and presses the Enter key. This assigns the value to the variable. As with any variable, the variable and its assigned value will be displayed in the Command Window unless a semicolon is typed at the very end of the **input** command. A script file that uses the **input** command to enter the points scored in each game to the program that calculates the average of the scores is shown below.

```
%This script file calculates the average of points scored in thre games.
%The points fran each game are assigned to the variables by
%using the input ccmnand.
gamel=iDput(1Enter the points scored in the fir s t game 1);
game2=iDput(1Enter the points scored in the second game 1);
game3=iDput(1Enter the points scored in the third game 1);
ave_point s=(gamel+game2+game3)/3
```

The following shows the Command Window when this script file (saved as

Chapter4Example4) is executed.

```
>> Chapter4Example4
Enter the points scored in the first game    67
Enter the points scored in the second game   91
Enter the points scored in the third game    70

ave_points =
   76
>>
```

The computer displays the message. Then the value of the score is typed by the user and the Enter key is pressed.

In this example scalars are assigned to the variables. In general, however, vectors and arrays can also be assigned. This is done by typing the array in the same way that it is usually assigned to a variable (left bracket, then typing row by row, and a right bracket).

The **input** command can also be used to assign a string to a variable. This can be done in one of two ways. One way is to use the command in the same form as shown above, and when the prompt message appears the string is typed between two single quotes in the same way that a string is assigned to a variable without the **input** command. The second way is to use an option in the **input** command that defines the characters that are entered as a string. The form of the command is:

```
variable_name =input ('prompt message', 's')
```

where the **'s'** inside the command defines the characters that will be entered as a string. In this case when the prompt message appears, the text is typed in without the single quotes, but it is assigned to the variable as a string. An example where the input command is used with this option is included in Sample Problem 6-4.

4.3 OUTPUT COMMANDS

As discussed before, MATLAB automatically generates a display when some commands are executed. For example, when a variable is assigned a value, or the name of a previously assigned variable is typed and the Enter key is pressed, MATLAB displays the variable and its value. This type of output is not displayed if a semicolon is typed at the end of the command. In addition to this automatic display, MATLAB has several commands that can be used to generate displays. The displays can be messages that provide information, numerical data, and plots. Two commands that are frequently used to generate output are **disp** and **fprintf.** The **disp** command displays the output on the screen, while the **fprintf** command can be used to display the output on the screen or to save the output to a file. The commands can be used in the Command Window, in a script file, and, as will be shown later, in a function file. When these commands are used

in a script file, the display output that they generate is displayed in the Command Window.

4.3.1 The disp Command

The disp command is used to display the elements of a variable without display- ing the name of the variable, and to display text. The format of the disp com- mand is:

disp(name of a variable) or disp('text as string')

- Every time the disp command is executed, the display it generates appears in a new line. One example is:

```
>> abc = [5          2  41;   A 2 x 3 array is assigned to variable abc.
>> disp(abc) 9  1;  7      disp command is used to display the abc array.
                    The      The array is displayed without its name.
     5      9      1
     7      2      4
>> disp('The problem has no solution.')
                                  The disp command is used
The problem has no solution.      to display a message.
>>
```

The next example shows the use of the disp command in the script file that cal- culates the average points scored in three games.

```
% This script file calculates the average points scored in thre games.
% The points  from each game are assigned to the variables by
% using the input command.
% The disp command is used to display the output.

game1=input('Enter the points scored in the first game ');
game2=input ('Enter the points scored in the second game ' );
game3=input('Enter the points scored in the third game ');
ave_points=(game1+game2+game3)/3;
disp (' ')                                              empty
disp('The average of points scored in a game is:')      Display text.
                                                  Display      line.
disp (' ')                                        Display empty line.
disp(ave_points)                  the value of the variable ave
                                  Display          _points.
```

When this file (saved as Chapter4Example5) is executed, the display in the Command Window is:

```
>> Chapter4Example5
Enter the points scored in the first game      89
Enter the points scored in the second game     60
Enter the points scored in the third game      82

The average of points scored in a game is:

   77
```

An empty line is displayed.
The text line is displayed.
An empty line is displayed.
The value of the variable ave_points is displayed.

- Only one variable can be displayed in a disp command. If elements of two variables need to be displayed together, a new variable (that contains the ele- ments to be displayed) must first be defined and then displayed.

In many situations it is nice to display output (numbers) in a table. This can be done by first defming a variable that is an array with the numbers and then using the disp command to display the array. Headings to the columns can also be created with the disp command. Since in the disp command the user cannot control the format (the width of the columns and the distance between the col- umns) of the display of the array, the position of the headings has to be aligned with the columns by adding spaces. As an example, the script file below shows how to display the population data from Chapter 2 in a table.

```
yr= [1984 1986 1988 1990 1992 1994 1996] ;
pop= [127 130 136 145 158 178 211];
tableYP (:,1) ::yr'; yr is entered as the frrst column in the array tableYP
(:,2)=pop'          is entered as the second column in the array tableYP.
;
disp (' YEAR    POPULATION')
disp ('       (MILLIONS) ')
disp (' ')
disp (tableYP)
```

The population data is entered in two row vectors.
tableYP.
Display heading (first line).
Display heading (second line).
Display an empty line.
Display the array tableYP.

When this script file (saved as PopTable) is executed, the display in the Command Window is:

```
>> PopTable
      YEAR      POPULATION
                (MILLIONS)

      1984          127
```

Headings are displayed.
An empty line is displayed.

1986	130
1988	136
1990	145
1992	158
1994	178
1996	211

The tableYP array is displayed.

Another example of displaying a table is shown in Sample Problem 4-3. Tables can also be created and displayed with the **fprintf** command, which is explained in the next section.

4.3.2 The fprintf Command

The **fprintf** command can be used to display output (text and data) on the screen or to save it to a file. With this command (unlike with the **disp** command) the output can be formatted. For example, text and numerical values of variables can be intermixed and displayed in the same line. In addition, the format of the numbers can be controlled.

With many available options, the **fprintf** command can be long and complicated. To avoid confusion, the command is presented gradually. First, this section shows how to use the command to display text messages, then how to mix numerical data and text, next how to format the display of numbers, and finally how to save the output to a file.

Using the fprintf command to display text:

To display text, the fprintf command has the form:

```
fprintf('text typed in as a string')
```

For example:

```
fprintf('The problem, as entered, has no solution. Please check the
input data, ')
```

If this line is part of a script file, then when the line is executed, the following is displayed in the Command Window:

```
The problem, as entered, has no solution. Please check the input
data.
```

With the **fprintf** command it is possible to start a new line in the middle of the string. This is done by inserting \n before the character that will start the new line. For example, inserting \n after the first sentence in the previous example gives:

```
fprintf('The problem, as entered, has no solution.\nPlease check the input
data.')
```

When this line executes, the display in the Command Window is:

```
The problem, as entered, has no solution. Please check
the input data.
```

The \n is called an escape character. It is used to control the display. Other escape characters that can be inserted within the string are:

\b Backspace.
\t Horizontal tab.

When a program has more than one **fprintf** command, the display gener- ated is continuous (the **fprintf** command does not automatically start a new line). This is true even if there are other commands between the **fprintf** com- mands. An example is the following script file:

```
fprintf('The problem, as entered, has no solution. Please check the input data.')
x = 6; d = 19 + S*x;
fprintf('Try to run the program later.') y = d + x;
fprintf('Use different input values.')
```

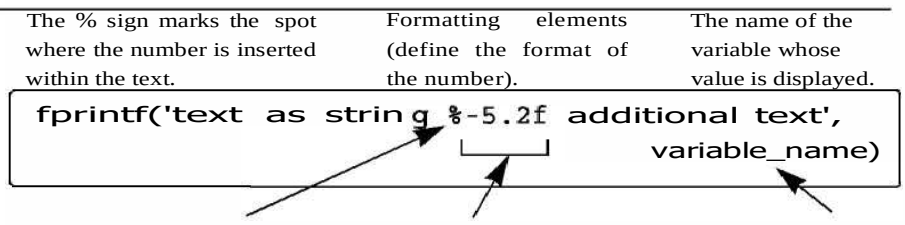
When this file is executed the display in the Command Window is:

```
The problem, as entered, has no solution. Please check the input data.Try to
run the program later.Use different input values.
```

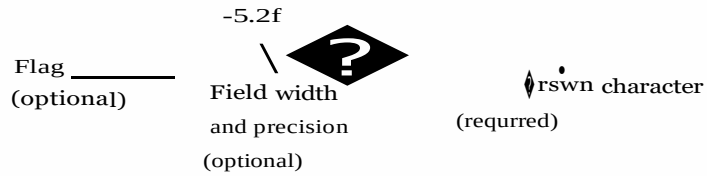
To start a new line with the **fprintf** command, \n must be typed at the start of the string.

Using the **fprintf** command to display a mix of text and numerical data:

To display a mix of text and a number (value of a variable), the **fprintf** com- mand has the form:



The formatting elements are:



The flag, which is optional, can be one of the following three characters:

Character used for flag	Description
–(minus sign)	Left-justifies the number within the field.
+(plus sign)	Prints a sign character(+ or-) in front of the number.
0(zero)	Adds zeros if the number is shorter than the field.

The field width and precision (5.2 in the previous example) are optional. The frrst number (5 in the example) is the field width, which specifies the mini- mum number of digits in the display. If the number to be displayed is shorter than the field width, spaces or zeros are added in front of the number. The precision is the second number(2 in the example). It specifies the number of digits to be dis- played to the right of the decimal point.

The last element in the formatting elements, which is required, is the con- version character, which specifies the notation in which the number is displayed. Some of the common notations are:

e	Exponential notation using lowercase e(e.g., 1.709098e+001). E Exponential notation using uppercase E(e.g., 1.709098E+001).
f	Fixed-point notation(e.g., 17.090980).
g	The shorter of e or f notations.
G	The shorter of E or f notations.
i	Integer.

Information about additional notation is available in the help menu ofMATLAB. As an example, the `fprintf` command with a mix of text and a number is used in the script file that calculates the average points scored in three games.

```
% This script file calculates the average points scored in thre games.
% The values are assigned to the variables by using the input ccmnand.
% The fprintf ccmnand is used to display the output.
game (1) - input(1Enter the points scored in the first game  1); game
(2) = input(1Enter the points scored in the second game  1); game(3) =
input(1Enter the points scored in the third game  1); ave_points =
mean(game);
```

```
fprintf('An average of %f points was scored in the thre games.' ,ave_points)
```

Text

% marks the position of the number.

Additional text.

The name of the variable whose value is displayed.

Notice that, besides using the **fprintf** command, this file differs from the ones shown earlier in the chapter in that the scores are stored in the ftrst three elements of a vector named garne, and the average of the scores is calculated by using the mean function. The Command Window where the script file above (saved as Chapter4Example6) was run is shown below.

```
>> Chapter4Example 6
Enter the points scored in the first game          75
Enter the points scored in the second game         60
Enter the points scored in the third game          81
An average of 72.000000 points was scored in the thre games.
>>
```

The display generated by the **fprintf** command combines text and a number (value of a variable).

With the **fprintf** command it is possible to insert more than one number (value of a variable) within the text. This is done by typing%g (or% followed by any formatting elements) at the places in the text where the numbers are to be inserted. Then, after the string argument of the command (following the comma), the names of the variables are typed in the order in which they are inserted in the text. In general the command looks like:

```
fprintf(' ..text...%g...%g...%f.....' ,variable1,variable2,variable3)
```

An example is shown in the following script file:

```
% This progra calculates the distance a projectile flies,
% given its initial velocity and the angle at which it is shot.
% the fprintf c:cmnand is used to display a mix of text and numbers.

v=1584;    % Initial velocity (km/h)
theta=30;      % Angle (degrees)
vms=v*l000/3600;                    Changing velocity units to m/s.
t=vms*sind(30)/9.81;                Calculating the time to highest point.
d=vms*cosd(30)*2*t/1000;            Calculating max distance.
```

```
fprintf('A projectile shot at %3.2f degrees with a velocity of %
4.2f km/h will travel a distance of %g km.\n',theta,v,d)
```

When this script file (saved as Chapter4Example7) is executed, the display in the Command Window is:

```
>> Chapter4Example7
A projectile shot at 30.00 degrees with a velocity of 15
84.00 km/h will travel a distance of 17.091 km.
>>
```

Additional remarks about the fprintf command:

- To place a single quotation mark in the displayed text, type two single quota- tion marks in the string inside the command.
- The **fprintf** command is vectorized. This means that when a variable that is a vector or a matrix is included in the command, the command repeats itself until all the elements are displayed. If the variable is a matrix, the data is used column by column.

For example, the script file below creates a 2 x 5 matrix T in which the first row contains the numbers 1 through 5, and the second row shows the correspond- ing square roots.

```
X=1:5;
y=sqrt(x);
T= [x; y]
fprintf('If the number is: %i, its square root is: %f\n',T)
```

Create a vector x.

Create a vector y.

Create 2 x 5 matrix T, frrst row is x, second row is y.

fprintf command displays two numbers from T in every

When this script file is executed, the display in the Command Window is:

The line.

```
T =
    1.0000    2.0000    3.0000    4.0000    5.0000
    1.0000    1.4142    1.7321    2.0000    2.2361
If the nmber is: 1, its square rot is: 1.000000 If the
number is: 2, its square rot is: 1.4142 14 If the
number is: 3, its square rot is: 1.73205 1 If
thenumber is: 4, its square rot is: 2.000000
If thenumber is:    its square rot is: 2.236068
```

The 2 x 5 matrix T

The **fprintf** command repeats five times, using the numbers from the matrix T col- umn after column.

Using the fprintf command to save output to a me:

In addition to displaying output in the Command Window, the **fprintf** com- mand can be used for writing the output to a file when it is necessary to save the output. The data that is saved can subsequently be displayed or used in MATLAB and in other applications.

Writing output to a file requires three steps:

a) Opening a file using the fopen command.
b) Writing the output to the open file using the **fprintf** command.
c) Closing the file using the **fclose** command.

Step a:

Before data can be written to a file, the file must be opened. This is done with the **fopen** command, which creates a new file or opens an existing file. The **fopen** command has the form:

```
fid= fopen('file_name', 'permission')
```

fid is a variable called the file identifier. A scalar value is assigned to **fid** when **fopen** is executed. The file name is written (including its extension) within sin- gle quotes as a string. The permission is a code (also written as a string) that tells how the file is opened. Some of the more common permission codes are:

- **'r'** Open file for reading (default).
- **'w'** Open file for writing. If the file already exists, its content is deleted. If the file does not exist, a new file is created.
- **'a'** Same as **'w'**, except that if the file exists the written data is appended to the end of the file.
- **'r+'** Open (do not create) file for reading and writing.
- **'w+'** Open file for reading and writing. If the file already exists, its con- tent is deleted. If the file does not exist, a new file is created.
- **'a+'** Same as **'w+'** except that if the file exists the written data is appended to the end of the file.

If a permission code is not included in the command, the file opens with the default code **'r'**. Additional permission codes are described in the help menu.

Step b:

Once the file is open, the **fprintf** command can be used to write output to the file. The **fprintf** command is used in exactly the same way as it is used to dis- play output in the Command Window, except that the variable **fid** is inserted inside the command. The **fprintf** command then has the form:

```
fprintf(fid, 'text %-5.2f additional text',vari
                                        able_name)
```

fid is added to the **fprintf** command.

Step c:

When the writing of data to the file is complete, the file is closed using the fclose command. The fclose command has the form:

```
fclose (fid)
```

Additional notes on using the **fprintf** command for saving output to a file:

- The created file is saved in the current directory.
- It is possible to use the **fprintf** command to write to several different files. This is done by first opening the files, assigning a different **fid** to each (e.g. **fidl,** fid2, fid3, etc.), and then using the **fid** of a specific file in the **fprintf** command to write to that file.

An example of using **fprintf** commands for saving output to two files is shown in the following script file. The program in the file generates two unit con- version tables. One table converts velocity units from miles per hour to kilometers per hour, and the other table converts force units from pounds to newtons. Each conversion table is saved to a different text file (extension .txt).

```
% Script file in which fprintf is used to write output to files.
% Tt;o conversion tables are created and saved to two different files.
% One converts mi/h to km/h, the other converts lb to N. clear al l
Vmph=10:10:100;                          Creating a vector of velocities in milh.
Vkmh= Vmph. *1 . 6 o 9;                  Converting mph to km/h.
TBLl= [Vmph; Vkmh];                      Creating a table (matrix) with two rows.
Flb=200:200:2000;                        a vector of forces in lb. FN=
Flb. *4. 44 8;                           Converting lb to N.
TBL2= [Flb; FN];                         a table (matrix) with two
fid1=fopen( 1 Vmph toVkm. txt 1 1 1W1l;  Creating file named fid2=fopen( 1
Flb toFN. txt 1 1 1W1) ;                 Open a .txt file named FlbtoFN.
fprintf(fid1,1Velocity Conversion Tabl e\n \n1);
                                         Creating rows.
fprintf(fid1,1 mi/h km/h \n1);           Open VmphtoVkm.
fprintf(fid1,1 %8.2f %8.2f\n11TBL1);
                                         Writing the data from the variable TBL1 to the file
                                         Writing a title and an empty line to the file fid1.
                                         Writing two column headings to the file fid1.
                                         fid1.
```

```
fprintf(fid2, 'Force Conversion  Table\n \n');
fprintf(fid2,•        Pounds          Newtons        \n');
fprintf(fid2,'        %8.2£           %8.2f\n',TBL2);
fclose( fidl);
fclose( fid2 );
```

Writing the force conversion table (data in variable TBL2) to the file fid2.

Closing the files fid1 and fid2.

When the script file above is executed two new .txt files, named VmphtoVkm and FlbtoFN, are created and saved in the current directory. These files can be opened with any application that can read .txt files. Figures 4-3 and 4- 4 show how the two files appear when they are opened with Microsoft Word.

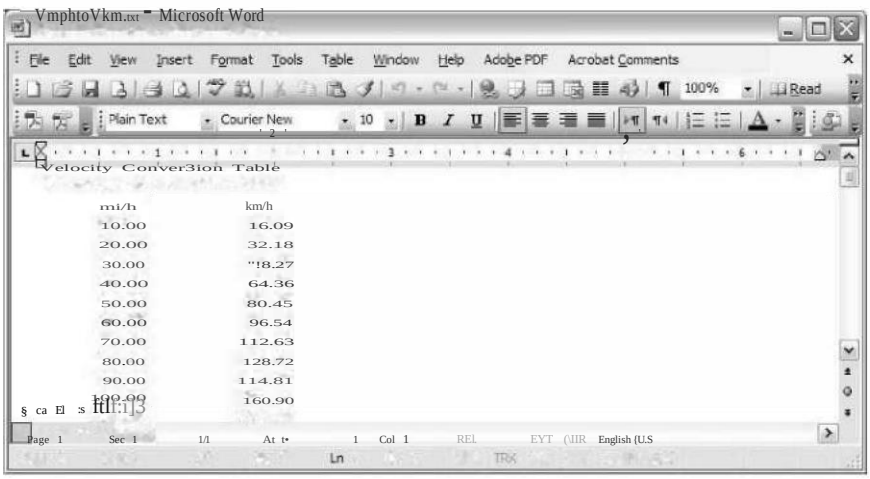


Figure 4-3: The VmphtoVkm.txt me opened in Word.

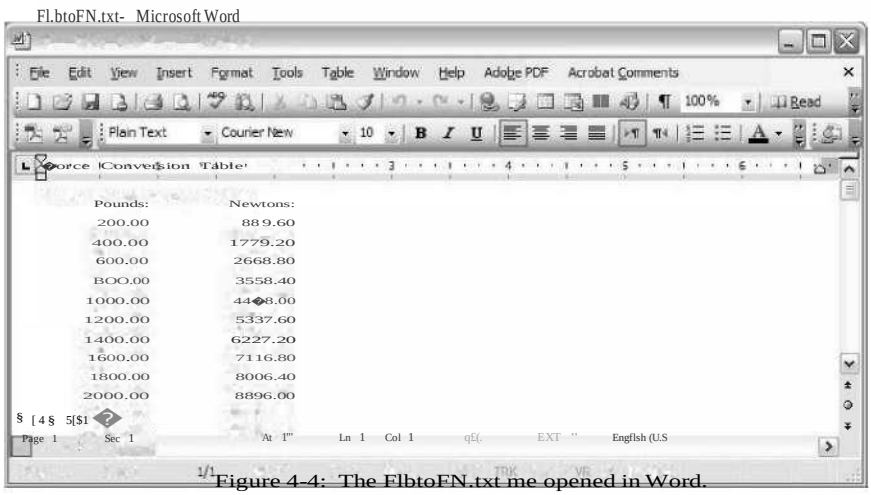


Figure 4-4: The FlbtoFN.txt me opened in Word.

4.4 THE save AND load COMMANDS

The save and load commands are most useful for saving and retrieving data for use in MATLAB. The save command can be used for saving the variables that are currently in the workspace, and the load command is used for retrieving vari- ables that have been previously saved, to the workspace. The workspace can be saved when MATLAB is used in one type of platform(e.g., PC), and retrieved for use in MATLAB in another platform (e.g., Mac). The save and load com- mands can also be used for exchanging data with applications outside MATLAB. Additional commands that can be used for this purpose are presented in Section 4.5.

4.4.1 The save Command

The save command is used for saving the variables(all or some of them) that are stored in the workspace. The two simplest forms of the save command are:

```
save file_name
```

and

```
save('file_name1)
```

When either one of these commands is executed, all of the variables currently in the workspace are saved in a file named file_name. mat that is created in the current directory. In rnat files, which are written in a binary format, each variable preserves its name, type, size, and value. These files cannot be read by other appli- cations. The save command can also be used for saving only some of the vari- ables that are in the workspace. For example, to save two variables named varl and var2, the command is:

```
save file_name varl var2
```

or

```
save('file_name1_'varl1_'var21)
```

The save command can also be used for saving in ASCII format, which can be read by applications outside MATLAB. Saving in ASCII format is done by adding the argument -ascii in the command(for example, save file_name -ascii). In the ASCII format the variable's name, type, and size are not pre- served. The data is saved as characters separated by spaces but without the vari- able names. For example, the following shows how two variables(a 1 x 4 vector and a 2 x 3 matrix) are defmed in the Command Window and then saved in ASCII format to a file named DatSavAsci:

```
>> v=[3 16 -4 7.3l;                    Create a 1 x 4 vector V.
>> A=[6 -2.1 15.5; -6.1 8 111;         Create a 2X3 matrix A.
>> save -ascii DatSavAsci              Save variables to a file named DatSavAsci.
```

Once saved, the file can be opened by any application that can read ASCII files. For example, Figure 4-5 shows the data when the file is opened with Notepad.

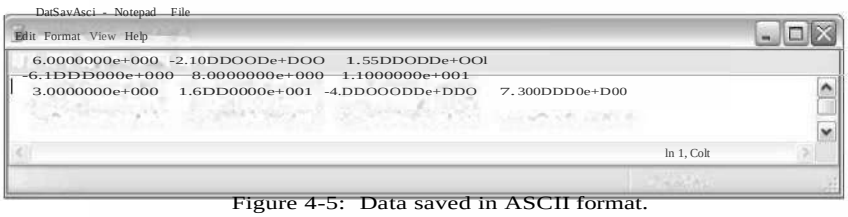


Figure 4-5: Data saved in ASCII format.

Note that the file does not include the names of the variables, just the numerical values of the variables (first A and then V) are listed.

4.4.2 The load Command

The load command can be used for retrieving variables that were saved with the save command back to the workspace, and for importing data that was created with other applications and saved in ASCII format or in text (.txt) files. Variables that were saved with the save command in .mat files can be retrieved with the command:

load file name or load ('file_name 1)

When the command is executed, all the variables in the file (with the name, type, size, and values as were saved) are added (loaded back) to the workspace. If the workspace already has a variable with the same name as a variable that is retrieved with the load command, then the variable that is retrieved replaces the existing variable. The load command can also be used for retrieving only some of the variables that are in the saved .mat file. For example, to retrieve two vari- ables named varl and var2, the command is:

file name varl or

load ('file_name1 , 'var11 , 'var21)

The load command can also be used to import data that is saved in ASCII or text (.txt) to the workspace. This is possible, however, only if the data in the file is in the form of a variable in MATLAB. Thus, the file can have one number (sca- lar), a row or a column of numbers (vector), or rows with the same number of numbers in each (matrix). For example, the data shown in Figure 4-5 cannot be loaded with the load command (even though it was saved in ASCII format with the save command), because the number of elements is not the same in all rows. (Recall that this file was created by saving two different variables.)

When data is loaded from an ASCII or text ftle into the workspace, it has to be assigned to a variable name. Data in ASCII format can be loaded with either of the following two forms of the load command:

```
load file_name
```

or

```
VarName=load('file_name')
```

If the data is in a text ftle, the extension .txt has to be added to the file name. The form of the 1oad command is then:

```
load file_name.txt
```

or

```
VarName=load('file_name.txt')
```

In the first form of the command the data is assigned to a variable that has the name of the file. In the second form the data is assigned to a variable named VarName.

For example, the data shown in Figure 4-6 (a 3 x 2 matrix) is typed in Notepad, and then saved as DataFromText.txt.

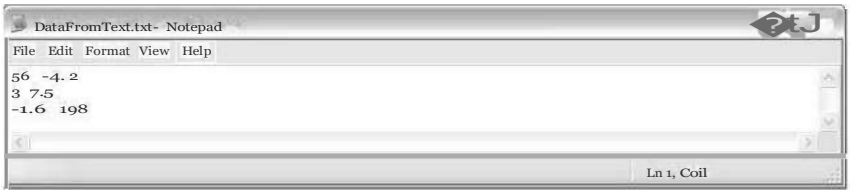


Figure 4-6: Data saved as .txt file.

Next, two forms of the load command are used to import the data in the text ftle to the Workspace ofMATLAB. In the first command the data is assigned to a variable named DfT. In the second command the data is automatically assigned to a variable named DataFromText, which is the name of the text file where the data was saved.

```
>> DfT=load( 'DataFromText.txt•)
DfT =
    56.0000    -4.2000
     3.0000     7.5000
    -1.6000   198.0000
>> load DataFromText.txt
>> DataFromText
DataFromText =
    56.0000    -4.2000
     3.0000     7.5000
    -1.6000   198.0000
```

Load the file DataFromText and assign the loaded data to the variable Dft.

Use the load command with the file DataFromText.

The data is assigned to a vari-able named DataFromText.

Importing data to (or exporting from) other applications can also be done, with MATLAB commands that are presented in the next section.

4.5 *IMPORTING* AND *EXPORTING DATA*

MATLAB is often used for analyzing data that was recorded in experiments or generated by other computer programs. This can be done by first importing the data into MATLAB. Similarly, data that is produced by MATLAB sometimes needs to be transferred to other computer applications. There are various types of data (numerical, text, audio, graphics, and images). This section describes only how to import and export numerical data, which is probably the most common type of data that needs to be transferred by new users of MATLAB. For other types of data transfer, look in the Help Window under File I/0.

Importing data can be done either by using commands or by using the Import Wizard. Commands are useful when the format of the data being imported is known. MATLAB has several commands that can be used for importing vari- ous types of data. Importing commands can also be included in a script file such that the data is imported when the script is executed. The Import Wizard is useful when the format of the data (or the command that is applicable for importing the data) is not known. The Import Wizard determines the format of the data and automatically imports it.

4.5.1 *Commandsfor Importing and Exporting Data*

This section describes-in detail-how to transfer data into and out of Excel spreadsheets. Microsoft Excel is commonly used for storing data, and Excel is compatible with many data recording devices and computer applications. Many people are also capable of importing and exporting various data formats into and from Excel. MATLAB also has commands for transferring data directly to and from formats such as csv and ASCII, as well as to the spreadsheet program Lotus
123. Details of these and many other commands can be found in the Help Window under File 1/0

Importing and exporting data into and from Excel:

Importing data from Excel is done with the `xlsread` command. When the com- mand is executed, the data from the spreadsheet is assigned as an array to a vari- able. The simplest form of the `xlsread` command is:

```
variable_name=xlsread('filename')
```

- `'filename'` (typed as a string) is the name of the Excel file. The directory of the Excel file must be either the current directory or listed in the search path.
- If the Excel file has more than one sheet, the data will be imported from the first sheet.

When an Excel file has several sheets, the xlsread command can be used to import data from a specified sheet. The form of the command is then:

```
variable_name=xlsread('filename1', 'sheet_name1)
```

- The name of the sheet is typed as a string.

Another option is to import only a portion of the data that is in the spreadsheet. This is done by typing an additional argument in the command:

```
variable_name = xlsread('filename','sheet_name','range')
```

- The 'range1 (typed as a string) is a rectangular region of the spreadsheet defined by the addresses (in Excel notation) of the cells at opposite comers of the region. For example, 'C2: ES 1 is a 4 x 3 region of rows 2, 3, 4, and 5 and columns C, D, and E.

Exporting data from MA TLAB to an Excel spreadsheet is done by using the xlswrite command. The simplest form of the command is:

```
xlswrite(' filename' ,variable_name)
```

- 'filename1 (typed as a string) is the name of the Excel file to which the data is exported. The file must be in the current directory. If the file does not exist, a new Excel file with the specified name will be created.
- variable_name is the name of the variable in MATLAB with the assigned data that is being exported.
- The arguments 'sheet_name' and 'range' can be added to the xls- write command to export to a specified sheet and to a specified range of cells, respectively.

As an example, the data from the Excel spreadsheet shown in Figure 4-7 is imported into MA TLAB by using the xlsread command.

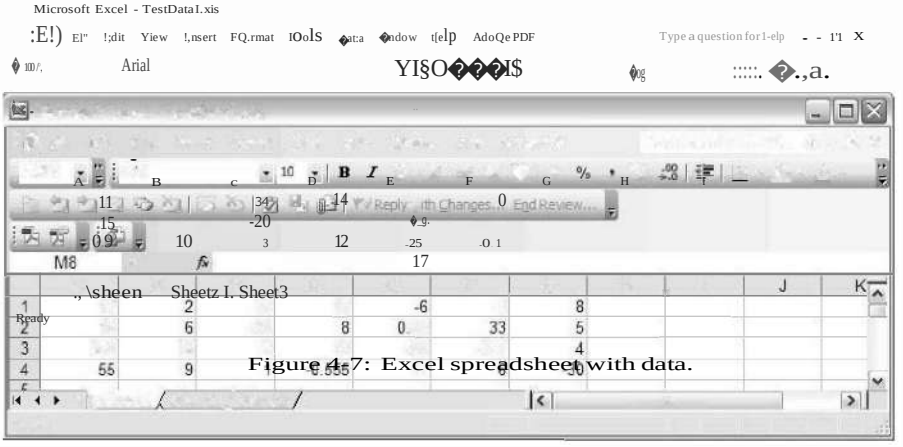


Figure 4-7: Excel spreadsheet with data.

The spreadsheet is saved in a file named TestDatal in a disk in drive A. After the Current Directory is changed to drive A, the data is imported into MAT- LAB by assigning it to the variable DATA:

```
>> DATA = xlsread('TestData1')
DATA =
   11.0000    2.0000   34.0000   14.0000   -6.0000         0    8.0000
   15.0000    6.0000  -20.0000    8.0000    0.5600   33.0000    5.0000
    0.9000   10.0000    3.0000   12.0000  -25.0000   -0.1000    4.0000
   55.0000    9.0000    1.0000   -0.5550   17.0000    6.0000  -30.0000
```

4.5.2 Using the Import Wizard

Using the Import Wizard is probably the easiest way to import data into MAT- LAB since the user does not have to know, or to specify, the format of the data.
The Import Wizard is activated by selecting Import Data in the File menu of the Command Window. (It can also be started by typing the command uiimport.) The Import Wizard starts by displaying a file selection box that shows all the data
files recognized by the Wizard. The user then selects the file that contains the data to be imported, and clicks Open. The Import Wizard opens the file and displays a portion of the data in a preview box so that the user can verify that the data is the correct choice. The Import Wizard tries to process the data, and if the wizard is successful, it displays the variables it has created with a portion of the data. The user clicks next and the wizard shows the Column Separator that was used. If the
variable has the correct data, the user can proceed with the wizard (click next);
otherwise the user can choose a different Column Separator. In the next window the wizard shows the name and size of the variable to be created in MATLAB. (When the data is all numerical, the variable in MATLAB has the same name as
the file from which the data was imported.) When the wizard ends (click finish),
the data is imported to MATLAB.

As an example, the Import Wizard is used to import numerical ASCII data saved in a .txt file. The data saved with the file name TestData2 is shown in Figure 4-8.

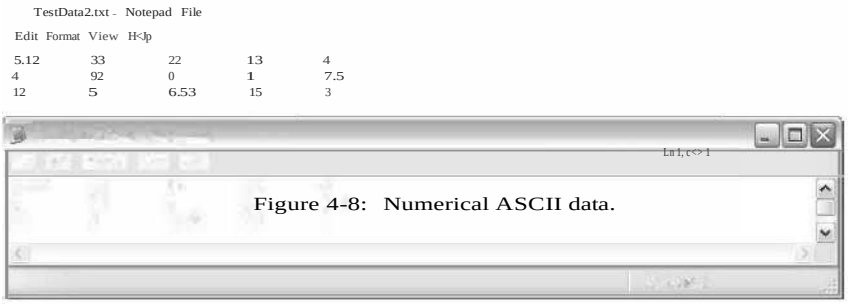


Figure 4-8: Numerical ASCII data.

The display of the Import Wizard during the import process for the TestData2 file is shown in Figures 4-9 and 4-10. Figure 4-10 shows that the name of the variable in MATLAB is TestData2 and its size is 3 x 5.

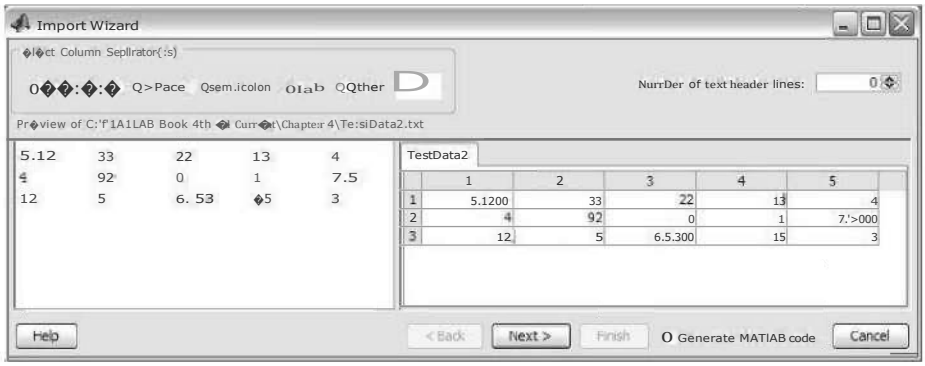


Figure 4-9: Import Wizard, first display.

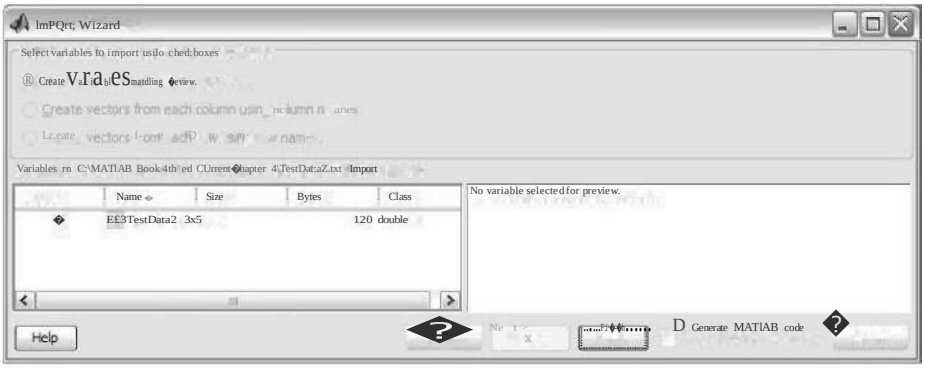


Figure 4-10: Import Wizard, second display.

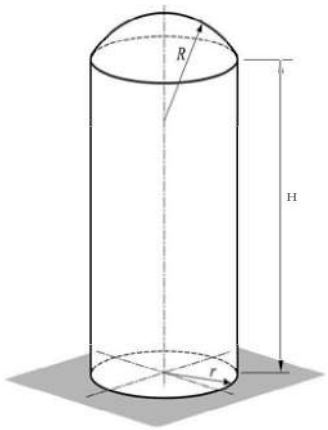
In the Command Window of MATLAB, the imported data can be displayed by typing the name of the variable.

```
>> TestData2

TestData2 =
    5.1200   33.0000   22.0000   13.0000    4.0000
    4.0000   92.0000         0    1.0000    7.5000
   12.0000    5.0000    6.5300   15.0000    3.0000
```

4.6 EXAMPLES OF MATLAB APPLICATIONS

Sample Problem 4-1: Height and surface area of a silo

A cylindrical silo with radius r has a spher- ical cap roof with radius R. The height of the cylindrical portion is H. Write a pro- gram in a script file that determines the height H for given values of r, R, and the volume V. In addition, the program calcu- lates the surface area of the silo.



Use the program to calculate the height and surface area of a silo with r = 30 ft, R = 45 ft, and a volume of200,000 ft3. Assign val- ues for r, R, and V in the Command Win- dow.

Solution

The total volume of the silo is obtained by adding the volume of the cylindrical part and the volume of the spherical cap. The volume of the cylinder is given by

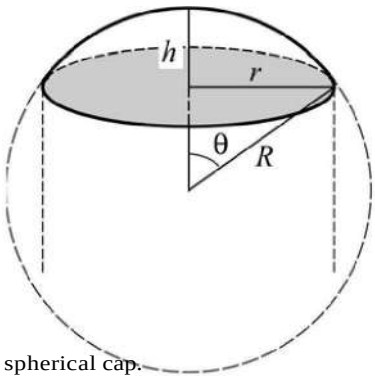
$$V_{cyl} = 1tr2H$$

and the volume of the spherical cap is given by:

$$V_{cap} = {}^{1}1th^{2}(3R- h)$$

where $h = R- Rcose^{3} = R(1- cos8)$,

and e is calculated from sine = ?.



Using the equations above, the height, H, of the cylindrical part can be expressed by

$$H = \frac{V_{-}}{1tr2}$$

The surface area of the silo is obtained by adding the surface areas of the cylindrical part and the spherical cap.

$$S = S_{cyl} + S_{cap} = 21trH + 21tRh$$

A program in a script file that solves the problem is presented below:

```
theta=asin(r/R); h=R*(1-                    Calculating e.
cos(theta));                                Calculating h.
Vcap=pi*hA2*(3*R-h)/3;          Calculating the volume of the cap.
```

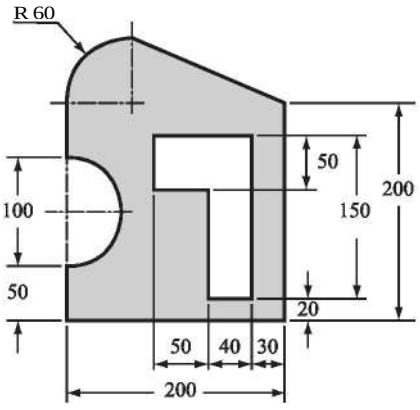
```
H=(V-Vcap)/(pi*r^2);                          % Calculating H.
S=2*pi*(r*H + R*h);                           % Calculating the surface area S.
fprintf('The height H is: %f ft.',H)
fprintf('\nThe surface area of the silo is: %f square ft.',S)
```

The Command Window where the script file, named silo, was executed is:

```
>> r=30; R=45; V=200000;                      % Assigning values to r, R, and V.
>> silo                                       % Running the script file named silo.
The height H is: 64.727400 ft.
The surface area of the silo is: 15440.777753 square ft.
```

Sample Problem 4-2: Centroid of a composite area

Write a program in a script file that calculates the coordinates of the centroid of a composite area. (A composite area can easily be divided into sections whose centroids are known.) The user needs to divide the area into sections and know the coordinates of the centroid (two numbers) and the area of each section (one number). When the script file is executed, it asks the user to enter the three numbers as a row in a matrix. The user enters as many rows as there are sections. A section that represents a hole is taken to have a negative area. For output, the program displays the coordinates of the centroid of the composite area. Use the program to calculate the centroid of the area shown in the figure.



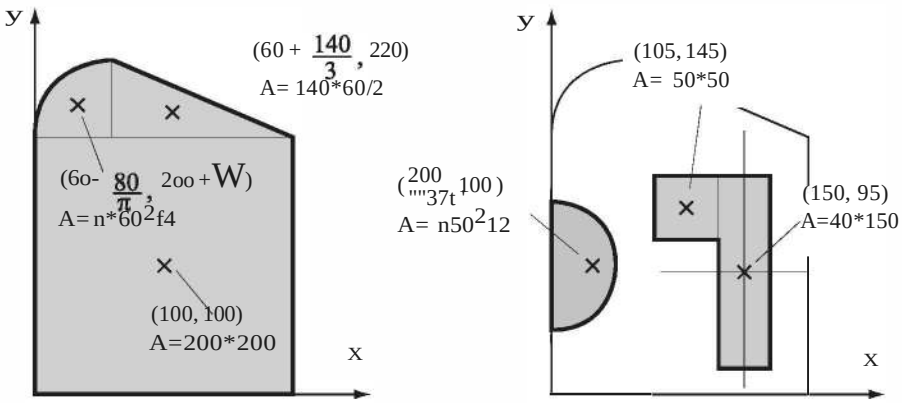
Dimensions in mm

Solution

The area is divided into six sections as shown in the following figure. The total area is calculated by adding the three sections on the left and subtracting the three sections on the right. The location and coordinates of the centroid of each section are marked in the figure, as well as the area of each section.

The coordinates X and Y of the centroid of the total area are given by $X = \frac{\sum xA}{\sum A}$ and $Y = \frac{\sum yA}{\sum A}$, where x, y, and A are the coordinates of the centroid and area of each section, respectively.

A script file with a program for calculating the coordinates of the centroid of a composite area is provided below.



Units: coordinates mm, area mm2

```
% The program calculates the coordinates of the centroid
% of a composite area.
clear C xsys As
C=input(1Enter a matrix in which each row has three ele-
ments.\nin each row enter the x andy coordinates of the
centroid and the area of a section.\n');
xs=C(: , 1) 1;
ys=C(: , 2) 1;
As=C(: , 3) 1;
A=sum(As);
x=sum(As.*xs)/A;
y=sum(As.*ys)/A;
fprintf(1The coordinates of the centroid are: ( %f, %f )\n1, x,y)
```

Creating a row vector for the x coordinate of each section (first column of c).

Creating a row vector for they coordinate of each section (second column of c).

Creating a row vector for the area of each section (third column of c).

Calculating the total area.

Calculating the coordinates of the centroid of the composite area.

The script file was saved with the name Centroid. The following shows the Com- mand Window where the script file was executed.

```
>> Centroid
Enter a matrix in which each row has three elements.
In each row enter the x and y coordinates of the centroid
and the area of a section.
```

```
[100 100 200*200
60-80/pi 200+80/pi pi*60A2/4
60+140/3 220 140*60/2
200/{3*pi) 100 -pi*50A2/2
105 145 -50*50
150 95 -40*150]
The coordinates of the centroid are: { 85.387547 , 131.211809 )
```

Entering the data for matrix C. Each row has three elements: the x, y, and A of a section.

Sample Problem 4-3: Voltage divider

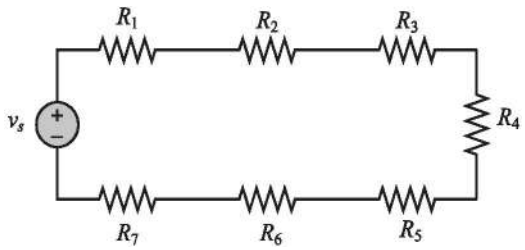
When several resistors are connected in an electrical circuit in series, the voltage across each of them is given by the voltage divider rule:

$$v_n = \frac{R_n}{R_{eq}} v_s$$

where vn and Rn are the voltage across resistor n and its resistance, respectively, $R_{eq} = \sum R_n$ is the equivalent resistance, and v_s is the source voltage. The power dissipated in each resistor is given by:

$$P_n = \frac{R_n}{R_{eq}^2} v_s^2$$

The figure below shows a circuit with seven resistors connected in series.



Write a program in a script file that calculates the voltage across each resistor, and the power dissipated in each resistor, in a circuit that has resistors connected in series. When the script file is executed, it requests that the user first enter the source voltage and then to enter the resistances of the resistors in a vector. The program displays a table with the resistance listed in the first column, the voltage across the resistor in the second column, and the power dissipated in the resistor in the third column. Following the table, the program displays the curent in the cir- cuit and the total power.

Execute the file and enter the following data for v_s and the R's.

v_s = 24V, R1 = 20Q, R2 = 14Q, R3 = 12Q, $R_4 = 18\Omega$, R5 = 8Q, R6 = 15Q, R1 = lOQ.

Solution

A script file that solves the problem is shown below.

```
% The program calculates the voltage across each resistor
% in a circuit that has resistors connected in series.
vs=input{'Please enter the source voltage ');
Rn=input{'Enter the values of the resistors as elements in a
row vector\n');
Req=sum{Rn);
vn=Rn*vs/Req;
Pn=Rn*vsA2/ReqA2;
i = vs/Req;
Ptotal = vs*i;
Table = [Rn', vn•, Pn'];
diSp{' ')
disp{' Resistance Voltage Power')
disp{' {Ohms) {Volts) {Watts)')
diSp{' ')
disp{Table)
diSp{' ')
fprintf{'The current in the circuit is %f Amps.•,i)
fprintf{'\nThe total power dissipated in the circuit is %f
Watts.',Ptotal)
```

Calculate the equivalent resistance. Apply the voltage divider rule.

Calculate the power in each resistor.

Calculate the current in the circuit.

Calculate the total power in the circuit.

Create a variable tab 1 e with the vectors Rn, vn, and Pn as columns.

Display headings for the columns.

Display an empty line.

Display the variable Table.

The Command Window where the script file was executed is:

```
>> VoltageDivider
Please enter the source voltage 24
Enter the value of the resistors as elements in a row vector

[20 14 12 18 8 15 10]
  Resistance   Voltage     Power
    {Ohms)     {Volts)    {Watts)
    20.0000     4.9485     1.2244
    14.0000     3.4639     0.8571
    12.0000     2.9691     0.7346
    18.0000     4.4536     1.1019
     8.0000     1.9794     0.4897
```

Name of the script file.

Voltage entered by the user.

Resistor values entered as a vector.

```
    15.0000          3.7113        0.9183
    10.0000          2.4742        0.6122

The current in the circuit is 0.247423 Amps.
The total power dissipated in the circuit is 5.938144 Watts.
```

4.7 PROBLEMS

Solve the following problems by first writing a program in a script file and then executing the program.

1. The Heat Index HI , calculated from the air temperature and relative humid- ity, is the apparent temperature felt by the body. An equation used by the National Weather Service for calculating the HIis given by:

   HI= -42.379 + 2.04901523T+ 10.14333127R-0.22475541R-6.83783 x 10-3Tl
   -5.481717 X 10-2R2 + 1.22874 X 10-3J'1R+ 8.5282 X 10--4TR2-1.99 X 10-6J'1R2

   where Tis the temperature in degrees F and R is the relative humidity in inte- ger percentage. Write a MATLAB program in a script file that calculates HI. For input the program asks the user to enter values forTand R. For output the program displays the message: "The Heat Index temperature is: XX." where XX is the value of the heat index rounded to the nearest integer. Execute the program entering T = 90°F and R = 90 %.

2. The monthly saving P that has to be deposit in a saving account that pays an annual interest rate of r in order to save a total amount ofF in N years can be calculated by the formula:

$$p = \frac{F(r/12)\ (1+}{r/12)12N-1}$$

   Calculate the monthly saving that has to be deposit in order to save $100,000 in 5, 6, 7, 8, 9, and 10 years if the annual interest rate is 4.35%. Display the results in a two-column table where the first column is the number of years and the second column is the monthly deposit.

3. The growth of some bacteria populations can be described by

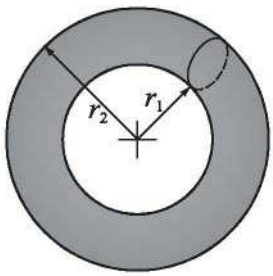
$$N = Noekt$$

   where N is the number of individuals at time t, N0 is the number at time t = 0 , and k is a constant. Assuming the number of bacteria doubles every 40 min- utes, determine the number of bacteria every two hours for 24 hours starting from an initial single bacterium.

4. The volume V and the surface area S of a torus-shaped water tube are given by:

$$V = \frac{1}{4}\pi^2(r1+r2)(r2-r1)^2 \quad \text{and} \quad S = \pi^2(ri-rf)$$

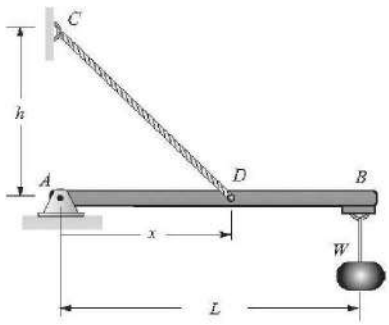
If $r1 = 0.7r2$, determine V and S for $r2 = 12, 16, 20, 24,$ and 28 in. Display the results in a four-column table where the first column is r2, the second r1, the third V, and the fourth S.



5. A beam with a lengthL is attached to the wall with a cable as shown. A load W = 500 lb is attached to the beam. The tension force, T, in the cable is given by:

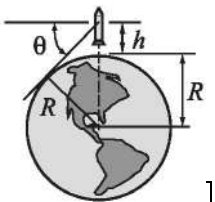
$$T = \frac{WL\sqrt{h^2+x^2}}{hx}$$

For a beam with L = 120 in. and h = 50 in., calculate T for x = 10, 30, 50, 70, 90, and 110 in.



6. Write a MATLAB program in a script file that calculate the average, standard deviation, and median of a list of grades as well as the number of grades on the list. The program asks the user (input command) to enter the grades as elements of a vector. The program then calculates the required quantities using MATLAB's built-in functions length, mean, std, and median. The results are displayed in the Command Window in the following format: "There are XX grades." where XX is the numerical value.
"The average grade is XX" where XX is the numerical value.
"The standard deviation is XX" where XX is the numerical value. "The median grade is XX." where XX is the numerical value.
Execute the program and enter the following grades: 92, 74, 53, 61, 100, 42, 80, 66, 71, 78, 91, 85, 79, and 68.

7. A rocket flying straight up measures the angle e with the horizon at different heights h. Write a MATLAB program in a script file that calculates the radius of the earth R (assuming the earth is a perfect sphere) at each data point and then determines the average of all the val- ues.



h(km)	4	8	12	16	20	24	28	32	36	40
B(deg)	2.0	2.9	3.5	4.1		5.0	5.4	5.7	6.1	6.4

8. Decay of radioactive materials can be modeled by the equation $A = A0ekt$, where A is the amount at time t, A0 is the amount at $t = 0$, and k is the decay constant (k > 0). Iodine-132 is a radioisotope that is used in thyroid function tests. Its half-life time is 13.3 hours. Calculate the relative amount oflodine- 132 (AIA0) in a patient's body 48 hours after receiving a dose. After deter- mining the value of k, defme a vector $t = 0, 4, 8, \ldots, 48$ and calculate the cor- responding values of AIA0•

9. The monthly payment, P, of aN years mortgage of an amount L that with a yearly interest rate of r is given by:

$$P = L\frac{\frac{r}{100 \cdot 12}\left(1+\frac{r}{100 \cdot 12}\right)^{12N}}{\left(1+\frac{r}{100 \cdot 12}\right)^{12N}-1}$$

where r is in % (e.g., 7.5% entered as 7.5). Write a MATLAB program in a script file that calculates P. When the program is executed it asks the user to enter the mortgage amount, the number of years, and the interest rate. The output is displayed in the following format: " The monthly payment of a XX years :XXX:XXX.XX mortgage with interest rate of :XX.XX percent is $X:XXX.:XX", where XX stands for the corresponding quantities. Use the program for determining the monthly payment of a $250,000 mortgage for 30 years and 4.5% yearly interest rate.

10. The balance of a loan, B, after n monthly payments is given by

$$B = A\left(1+\frac{r}{1200}\right)^{n} - \frac{P}{r/1200}\left[\left(1+\frac{r}{1200}\right)^{n}-1\right]$$

where A is the loan amount, P is the amount of a monthly payment, and r is the yearly interest rate entered in % (e.g., 7.5% entered as 7.5). Consider a 5-year, $20,000 car loan with 6.5% yearly interest that has a monthly payment of $391.32. Calculate the balance of the loan after every 6 months (i.e., at $n = 6, 12, 18, 24, \ldots, 54, 60$). Each time, calculate the percent of the loan that is already paid. Display the results in a three-column table, where the first col- umn displays the month and the second and third columns display the corre- sponding value of B and percentage of the loan that is already paid, respectively.

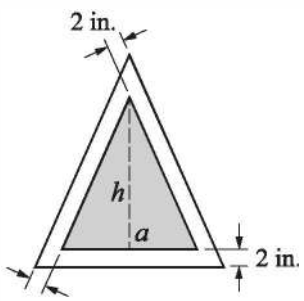
11. Early explorers often estimated altitude by measuring the temperature of boil- ing water. Use the following two equations to make a table that modem-day hikers could use for the same purpose.

$$p = 29.921(1-6.8753x10\text{--}\{;h), \qquad Tb = 49.1611np+44.932$$

where p is atmospheric pressure in inches of mercury, Tb is boiling tempera- ture in oF, and h is altitude in feet. The table should have two columns, the

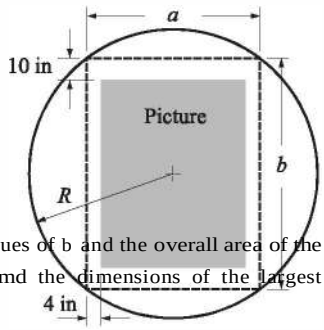
first altitude and the second boiling temperature. The altitude should range between -500ft and 10,000ft at increments of 500ft.

12. An isosceles triangle sign is designed to have a triangular printed area of 600 in.2 (shaded area with a base length of a and height of h in the fig- ure). As shown in the figure, there is a 2 in. gap between the sides of the triangles. Write a MAT- LAB program that determine the dimensions a and h such that the overall area of the sign will be as small as possible. In the program defme a vec-



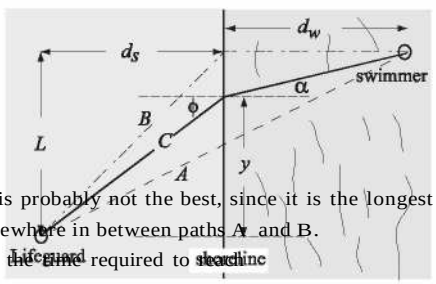
tor a with values ranging from 10 to 120 with increments of 0.1. Use this vector for calculating the corresponding values of h and the overall area of the sign. Then use MAT- LAB's built-in function min to fmd the dimensions of the smallest sign.

13. A round billboard with radius R – 55in. is designed to have a rectangular picture placed inside a rectangle with sides a and b. The mar- gins between the rectangle and the picture are 10 in. at the top and bottom and 4 in. at each side. Write a MATLAB program that deter- mines the dimensions a and b such that the overall area of the picture will be as large as possible. In the program define a vector a with values ranging from 5 to 100 with increments



of 0.25. Use this vector for calculating the corresponding values of b and the overall area of the picture. Then use MATLAB's built-in function max to fmd the dimensions of the largest rectangle.

14. A student has a sumer job as a lifeguard at the beach. After spotting a swimmer in trouble, he tries to deduce the path by which he can reach the swimmer in the shortest time. The path of shortest distance (path A) is obviously not the best since it maximizes the time spent swimming (he can run faster than he can swim).



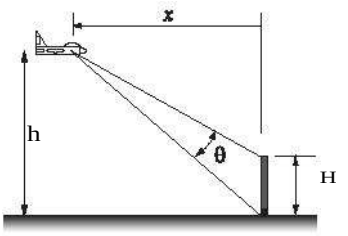
Path B minimizes the time spent swimming but is probably not the best, since it is the longest (reasonable) path. Clearly the optimal path is somewhere in between paths A and B.

Consider an intermediate path C and determine the time required to reach the

the swimmer in terms of the lifeguard speed $v_{lg} = 3$ mls the swim speed $v_{sw} = 1$ mls; the distances L = 48 m, d. = 30m, and $d_w = 42$ m; and the lateral distancey at which the lifeguard enters the water. Create a vectory that ranges between path A and path B (y = 20, 21, 22, ..., 48 m) and compute a time t for each y. Use MA1LAB built-in function min to find the minimum time t.1, and the entry pointy for which it occms. Determine the angles that coJ.TeSpond to the calculated value of y and investigate whether your result satisfies Snell's law ofrefraction:

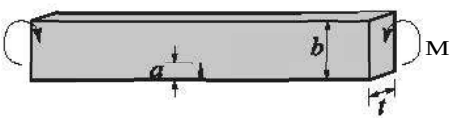
$$\frac{\sin\phi}{\sin\alpha} = \frac{v_{lg}}{v_{sw}}$$

15. An aiiplane is flying at a height of h = 900 ft while watching a target dtat is 70 ft tal (H = 70 ft)t as shown in the figure. The best view of the target is when e is maximum. Write a MATLAB program that determin the distance x at which e is maximum. Define a vector x with elements ranging ftom 50 to 1500 with spacing of 0.5. Use this vector to calculate the corresponding values of e. Then use MATLAB's built-in func- tion max to find the value of x that corresponds to the largest value of e.



16. The stress intensity factor K at a crack in a beam exposed to pure M bending Mis given by:

$$K = C\sigma_{cr}\sqrt{\pi a}$$

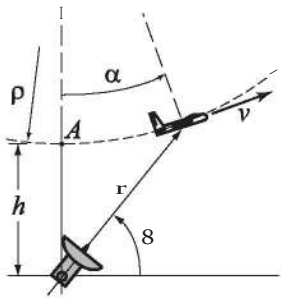


where $\sigma_{cr} = \frac{M}{t b^2}$. a is the crack length. b is the widtht tis the thickness, and C is a parameter that depends on the geometry of the specimen and crack. For the case of pure bending,

$$C = \sqrt{\frac{\tan\beta}{p}}\left[\frac{0.923 + 0.199(1-\sin\beta)^2}{\cos\beta}\right] \quad \text{where } \alpha = a/b \text{ and } \beta = (\pi a)/2$$

Write a program in a script file that calculates the stress intensity factor K. The program should read values of M, b, t, and a :from an ascii text file using the load command. The output should be in the form of a paragraph combining text and numbers,- i.e., somethin like: "The stress intensity fac- tor for a beam that is 0.25 m wide and 0.01 m thick with an edge crack of0.05 m and an applied moment of20N-m is XX Pa-sqrt(m)." where XX stands for the value of K. Use the program to calculate K when M = 20 N-m, b = 0.25 m, t = 0.01 m, and a = 0.25 m.

17. The airplane shown is flying at a constant speed of $v = 50$ mls in a circular path of radius $= 2000$ m and is being tracked by a radar station positioned a distance $h = 500$ m below the bottom of the plane path (point A). The airplane is at point A at $t = 0$, and the angle a as a function of time is given (in radians) by $a = \frac{\pi}{180}t$. Write a MATLAB program that calculates θ and r as functions of time. The program should first determine the time at which $a = 90°$. Then construct a vectort having 15 elements over the interval $0 \le t \le 90$, and calculate θ and r at each time. The program should print the values of p, h, and v, followed by a 15×3 table where the first column ist, the second is the angle e in degrees, and the third is the corresponding value of r.



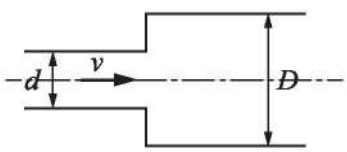
18. The intrinsic electrical conductivity cr of a semiconductor can be approxi- mated by:

$$o = e^{(c-\frac{\xi}{2kT})}$$

where cr is measured in (Q-m)-1, Eg is the band gap energy, k is Boltzmann's constant ($8.62 \times 10\text{-}5$ ev/K), and Tis temperature in kelvins. For Germanium, C $= 13.83$ and Eg $= 0.67$ ev. Write a program in a script file that calculates the intrinsic electrical conductivity for Germanium for various tem- peratures. The values of the temperature should be read from an xls spread- sheet using the `xlsread` command. The output should be presented as a table where the frrst column is the temperature and the second column is the intrinsic electrical conductivity. Use the following values for temperature: 400, 435, 475, 500, 520, and 545 K.

19. The pressure drop l!.p in Pa for a fluid flowing in a pipe with a sudden increase in diameter is given by:

$$\Delta p = H_1 - (\rho v^2$$

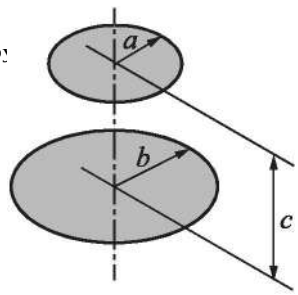


where is the density of the fluid, v, the velocity of the flow, and d andD are defmed in the figure. Write a program in a script file that calculates the pres- sure drop l!.p. When the script file is executed it request the user to input the density in kg/m^3, the velocity in m/s, and values of the non-dimensional ratio d/D as a vector. The program displays the inputted values of and v fol- lowed by a table with the values of d/D in the first column and the corre-

sponding values of Ap inthe second column.

Executethe program assuming flow of gasoline (ρ = 737 kg/m^3) at ν = 5 ml s andthe following ratios of diameters d/D = 0.9, 0.8, 0.7, 0.6, 0.5, 0.4, 0.2.

20. The net heat exchange by radiation from plate 1 with radius b to plate 2 with radius athat are sep- arated by distance c is given by:



$$q = cmb^2F1\_2(Tf-T1)$$

Where T1 and T2 are the absolute temperatures of the plates, a = 5.669 x IQ-8 W/(m^2-K^4) is the Stefan-Boltzmann constant, and F1_2 is a shape factor which, for the arrangement inthe figure, is given by:

$$Fl\text{-}2 = \frac{1}{2}[Z - \sqrt{Z^2 - 4X^2Y^2}]$$

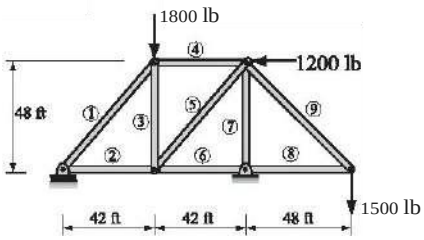
Where X= ale, Y = c/b, and Z = 1 + (1 + X2)J'2. Write a script file that calculates the heat exchange q. For input the program asks the user to enter values for T1, T2, a, b, and c. For output the program prints a sumary of the geometry and temperatures and then print the value of q. Use the script to cal- culate the results for T1 = 400 K, T2 = 600 K, a = 1 m, b = 2m, and c = 0.1, 1, and 10m.

21. Given the coordinates ofthree points (x1,y1), (x2,J2), and (x3,y3) it is possi- ble to find the coordinates of the center of the circle (Cx, Cy) that passes through the three points by solving the following simultaneous equations:

$$2\begin{bmatrix} (x1-x2) & (y1-y2) \\ (x2-x3) & (y2-y3) \end{bmatrix} \begin{bmatrix} cx \\ CJI \end{bmatrix} = \begin{bmatrix} (xr+yf)-(x\square+y_2^2) \\ (x\square+YD-(x\square+y_3^2) \end{bmatrix}$$

Write a program in a script file that calculates the coordinates of the center and the radius of a circle that passes through three given points. When exe- cuted the program asks the user to enter the coordinates of the three points. The program then calculates the center and radius and displays the results in the following format: "The coordinates of the center are (xx.x, xx.x) andthe radius is xx.x.", where xx.x stands for the calculated quantities rounded to the nearest tenth. Execute the program entering the following three points: (10.5, 4), (2 8.6), and (--4, -7).

22. A truss is a structure made of members joined at their ends. For the truss shown in the figure, the forces in the nine members are determined by solv- ing the following system of nine equations:



$F2 + \cos(48.81°)F1 - 0$

$F6 + \cos(48.81°)F5 - F2 - 0,$

$\sin(48.81°)F5 + F3 - 0 \qquad -\cos(48.81°)F1 + F4 - 0$

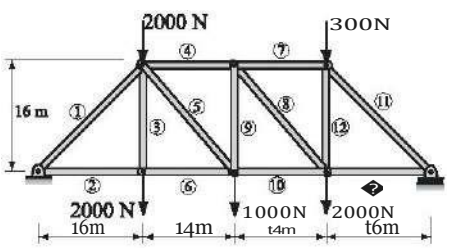
$-\sin(48.81°)F1 + F3 - 1800, \quad -F\cos(48.81°)F5 - 1200,$

$-F7 - \sin(48.81°)F5 - \sin(45°)F9 - 0, \quad \sin(45°)F9 - 1500,$

$-\cos(45°)F9 - F8 - 0$

Write the equations in matrix form and use MATLAB to determine the forces in the members. A positive force means tensile force and a negative force means compressive force. Display the results in a table where the first column displays the member number and the second column displays the correspond- ing force.

23. A truss is a structure made of mem- bers joined at their ends. For the truss shown in the figure, the forces in the 13 members are determined by solving the following system of 13 equations.



$F2 + 0.7071F1 - 0, \qquad -F2 + F6 - 0$

$F3 - 2000 - 0,$

$F4 + 0.6585F5 - 0.7071F1 - 0$

$0.7071F1 + F3 + 0.7526F5 + 2000 - 0, \quad F_7 + 0.6585F_8 - F_4 = 0$

$0.7526F8 + F9 - 0, \quad F10 - 0.6585F5 - F6 - 0, \quad F9 + 0.7526F5 - 1000 - 0$

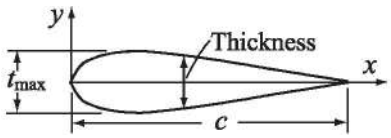
$0.7071F11 - F7 - 0, \qquad 0.7071F11 + F12 + 3000 - 0,$

$F12 + 0.7526F8 - 2000 - 0, \quad F13 + 0.7071F11 - 0$

Write the equations in matrix form and use MATLAB to determine the forces in the members. A positive force means tensile force and a negative force means compressive force. Display the results in a table where the first column displays the member number and the second column displays the correspond- ing force.

24. The graph of the function $f(x) = ax^3 + bx^2 + ex + d$ passes through the points (-2.6 , --68), (0.5, 5.7), (1 .5, 4.9), and (3.5, 88). Determine the constants a, b, c, and d. (Write a system of four equations with four unknowns, and use MAT- LAB to solve the equations.)

25. The surface of many airfoils can be described with an equation of the form



$$y = \pm \frac{tc}{0.2}\left[a_0\sqrt{x/c} + a1(x/c) + a2(x/c)^2 + a3(x/c)^3 + a4(x/c)^4\right]$$

where t is the maximum thickness as a fraction of the chord length c (e.g., $t_{max} = ct$). Given that $c = 1$ m and $t = 0.2$ m, the following values for y have been measured for a particular airfoil:

x(m)	0.15	0.35	0.5	0.7	0.85
y(m)	0.08909	0.09914	0.08823	0.06107	0.03421

Determine the constants a0, a1, a2, a3, and a4• (Write a system of five equa- tions and five unknowns, and use MATLAB to solve the equations.)

26. During a golf match, a certain number of points are awarded for each eagle and a different number for each birdie. No points are awarded for par, and a certain number of points are deducted for each bogey and a different number deducted for each double bogey (or worse). The newspaper report of an important match neglected to mention what these point values were, but did provide the following table of the results:

Golfer	Eagles	Birdies	Pars	Bogeys	Doubles	Points
A	1	2	10	1	1	5
B	2	3	11	0	1	12
C	1	4	10	1	10	11
D	1	3	10	12	0	8

From the information in the table write four equations in terms of four unknowns. Solve the equations for the unknown points awarded for eagles and birdies and points deducted for bogeys and double bogeys.

27. The dissolution of copper sulfide in aqueous nitric acid is described by the following chemical equation:

$$aCuS + bNO_3^- + cH^+ \rightarrow dCu^{2+} + eSO_4^{2-} + fNO + gH_2O$$

where the coefficients a, b, c, d, e,J, and g are the numbers of the various mol- ecule participating in the reaction and are unknown. The unknown coeffi- cients are determined by balancing each atom on left and right and then balancing the ionic charge. The resulting equations are:

$a = d$, $a = e$, $b = f$, $3b = 4e+f+g$, $c = 2g$, $-b+c = 2d-2e$

There are seven unknowns and only six equations. A solution can still be obtained, however, by taking advantage of the fact that all the coefficients must be positive integers. Add a seventh equation by guessing a – 1 and solve the system of equations. The solution is valid if all the coefficients are positive integers. If this is not the case, take a – 2 and repeat the solution. Continue the process until all the coefficients in the solution are positive inte- gers.

28. The wind chill temperature, Twc, is the air temperature felt on exposed skin due to wind. In U.S. customary units it is calculated by:

$$T_{wc} = 35.74 + 0.6215T - 35.75v^{0.16} + 0.4275Tv^{0.16}$$

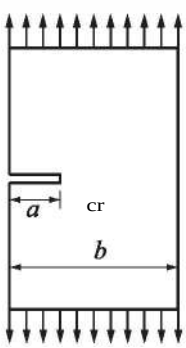
where Tis the temperature in degrees F, and v is the wind speed in mi/h. Write a MATLAB program in a script file that displays the following chart of wind chill temperature for given air temperature and wind speed in the Command Window:

	Temperature (F)								
	40	30	20	10	0	-10	-20	-30	-40
Speed (mi/h)									
10	34	21	9	-4	-16	-28	-41	-53	-66
20	30	17	4	-9	-22		-48	-61	-74
30	28	15	1	-12	-26	-39	-53	-67	-80
40	27	13	-1	-15	-29	-43	-57	-71	-84
so	26	12	-3	-17	-31	-45	-60	-74	-88
60	25	10	-4	-19	-33	-48	-62	-76	-91

29. The stress intensity factor K at a crack is given by (j

$K = C\sigma\sqrt{\pi a}$ where a is the far-field stress, a is the crack length, and C is a parameter that depends on the geometry of the specimen and crack. For the case of the edge crack shown in the figure, C is given by:

$$C = 0.265\left(1 - \frac{a}{b}\right) + \frac{0.857 + 0.265\frac{a}{b}}{\left(1 - \frac{a}{b}\right)^{3/2}}$$



Write a script file that will print out a table of values with the ratio a/b in the first column and the corresponding parameter C in the second column. let a/b range between O and0.95 with increments of0.05.

Unit 5

Two-Dimensional Plots

Plots are a very useful tool for presenting information. This is true in any field, but especially in science and engineering, where MATLAB is mostly used. MATLAB has many commands that can be used for creating different types of plots. These include standard plots with linear axes, plots with logarithmic and semi-logarith- mic axes, bar and stairs plots, polar plots, three-dimensional contour surface and mesh plots, and many more. The plots can be formatted to have a desired appear- ance. The line type (solid, dashed, etc.), color, and thickness can be prescribed, line markers and grid lines can be added, as can titles and text comments. Several graphs can be created in the same plot, and several plots can be placed on the same page. When a plot contains several graphs and/or data points, a legend can be added to the plot as well.

This chapter describes how MATLAB can be used to create and format many types of two-dimensional plots. Three-dimensional plots are addressed sep- arately in Chapter 9. An example of a simple two-dimensional plot that was cre- ated with MATLAB is shown in Figure 5-l. The figure contains two curves that show the variation of light intensity with distance. One curve is constructed from data points measured in an experiment, and the other curve shows the variation of light as predicted by a theoretical model. The axes in the figure are both linear, and different types of lines (one solid and one dashed) are used for the curves. The theoretical curve is shown with a solid line, while the experimental points are con- nected with a dashed line. Each data point is marked with a circular marker. The dashed line that connects the experimental points is actually red when the plot is displayed in the Figure Window. As shown, the plot in Figure 5-1 is formatted to have a title, axis titles, a legend, markers, and a boxed text label.

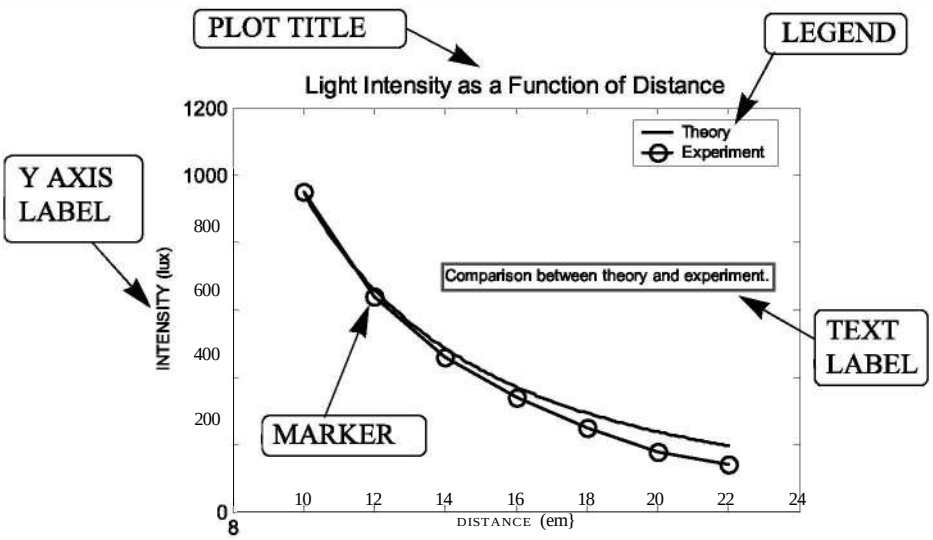
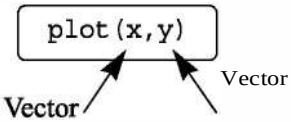


Figure 5-1: Example of a formatted two-dimensional plot.

5.1 THE plot COMMAND

The plot command is used to create two-dimensional plots. The simplest form of the command is:



The arguments x andy are each a vector (one-dimensional array). The two vec- tors must have the same number of elements. When the plot command is exe- cuted, a figure is created in the Figure Window. If not already open, the Figure Window opens automatically when the command is executed. The figure has a single curve with the x values on the abscissa (horizontal axis) and they values on the ordinate (vertical axis). The curve is constructed of straight-line segments that connect the points whose coordinates are defined by the elements of the vec- tors x andy. Each of the vectors, of course, can have any name. The vector that is typed first in the plot command is used for the horizontal axis, and the vector that is typed second is used for the vertical axis. If only one vector is entered as an input argument in the plot command (for example plot (y)) than the figure will show a plot of the values of the elements of the vector (y(l), y(2), y(3), ...) versus the element number (1, 2, 3, ...).

The figure that is created has axes with a linear scale and default range. For example, if a vector x has the elements 1, 2, 3, 5, 7, 7.5, 8, 10, and a vector y has

the elements 2, 6.5, 7, 7, 5.5, 4, 6, 8, a simple plot ofy versus x can be created by typing the following in the Command Window:

```
>> X=[l.l  1.8  3.2  5.5  7  7.5  8  10];
>> y=[2  6.5  7  7  5.5  4  6  8];
>> plot(x,y)
```

Once the plot command is executed, the Figure Window opens and the plot is displayed, as shown in Figure 5-2.

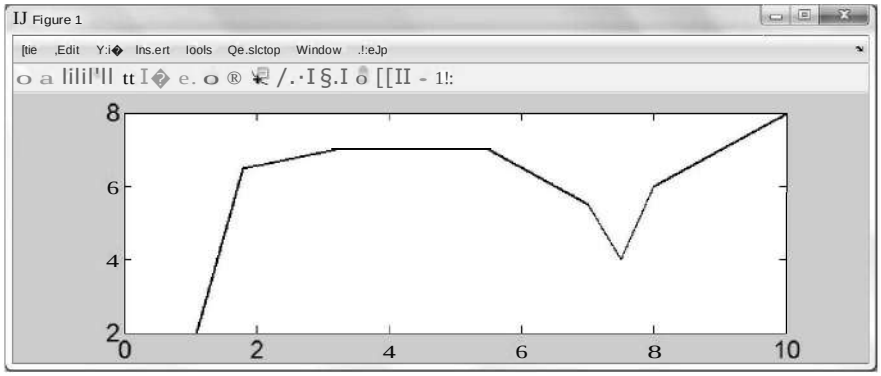
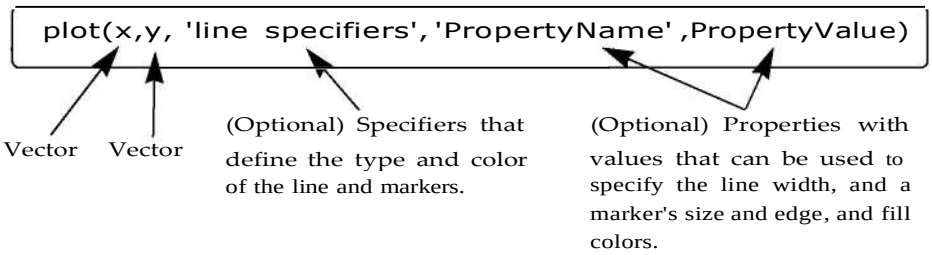


Figure 5-2: The Figure Window with a simple plot.

The plot appears on the screen in blue, which is the default line color.

The plot command has additional, optional arguments that can be used to specify the color and style of the line and the color and type of markers, if any are desired. With these options the command has the form:



Line Specifiers:

Line specifiers are optional and can be used to defme the style and color of the line and the type of markers (if markers are desired). The line style specifiers are:

Line Style	Specifier
solid (default)	-
dashed	--

Line Style	Specifier
dotted	:
dash-dot	-.

The line color specifiers are:

Line Color	Specifier
red	r
green	g
blue	b
cyan	c

Line Color	Specifier
magenta	m
yellow	y
black	k
white	w

The marker type specifiers are:

Marker Type	Specifier		Marker Type	Specifier
plus sign	+		square	s
circle	o		diamond	d
asterisk	*		five-pointed star	p
point	.		six-pointed star	h
cross	x		triangle (pointed left)	<
triangle (pointed up)	^		triangle (pointed right)	>
triangle (pointed down)	v			

Notes about using the specifiers:

- The specifiers are typed inside the plot command as strings.
- Within the string the specifiers can be typed in any order.
- The specifiers are optional. This means that none, one, two, or all three types can be included in a command.

Some examples:

`plot(x,y)` A blue solid line connects the points with no markers (default).

`plot(x,y 'r')` A red solid line connects the points.

`plot(x, y, '--y')` A yellow dashed line connects the points.

`plot(x,y '*')` The points are marked with * (no line between the points).

`plot(x,y 'g:d')` A green dotted line connects the points that are marked with diamond markers.

Property Name and Property Value:

Properties are optional and can be used to specify the thickness of the line, the size of the marker, and the colors of the marker's edge line and fill. The Property Name is typed as a string, followed by a comma and a value for the property, all inside the `plot` command.

Four properties and their possible values are:

Property name	Description	Possible property values
LineWidth (or linewidth)	Specifies the width of the line.	A number in units of points (default 0.5).
MarkerSize (or markersize)	Specifies the size of the marker.	A number in units of points.
MarkerEdgeColor (or markeredgecolor)	Specifies the color of the marker, or the color of the edge line for filled markers.	Color specifiers from the table above, typed as a string.
MarkerFaceColor (or markerfacecolor)	Specifies the color of the filling for filled markers.	Color specifiers from the table above, typed as a string.

For example, the command

```
plot(x,y,'-mo','LineWidth',2,'markersize'.12,'MarkerEdgeColor','g',
     'markerfacecolor','y')
```

creates a plot that connects the points with a magenta solid line and circles as markers at the points. The line width is 2 points and the size of the circle markers is 12 points. The markers have a green edge line and yellow filling.

A note about line specifiers and properties:

The three line specifiers, which indicate the style and color of the line, and the type of the marker can also be assigned with a PropertyName argument fol- lowed by a PropertyValue argument. The Property Names for the line speci- fiers are:

Specifier	Property Narne	Possible property values
Line style	linestyle (or LineStyle)	Line style specifier from the table above, typed as a string.
Line color	color (or Color)	Color specifier from the table above, typed as a string.
Marker	marker (or Marker)	Marker specifier from the table above, typed as a string.

As with any command, the plot command can be typed in the Command Window, or it can be included in a script file. It also can be used in a function file. It should also be remembered that before the plot command can be executed, the vectors x and y must have assigned elements. This can

be done. as was explained in Chapter 2. by entering values directly, by using commands, or as the result of mathematical operations. The next two subsections show examples of creating simple plots.

5.1.1 Plot ofGiven Dlda

In 1his case given data is used to create vectors that are then used in the **plot command**. The following table contains sales data of a company from 1988 to 1994.

Year	1988	1989	1990	1991	1992	1993	1994
Sales (millions)	8	12	20	22	18	24	27

To plot this data, the list of years is assigned to one vector (named yr), and the corresponding sales data is assigned to a second vector (named sle). The Com Window where the vectors are created and the plot command is used is shown below:

```
>> yr:[1988sls1994];
>> ale•[l   12   20   22   18   24   27];
>>   plot(yr,ala,•--r••,•linawidth•,2,•markaraisa•,12)
>>
```

Line Specifiers: dashed red line and asterisk marker.

Property Name and Property Value: the line width is 2 points and the marker size is 12 points.

Once the plot comd is executed, the Figure Wm.dow with the plot, as shown in Figure S-3, opens. The plot appears on the screen in red.

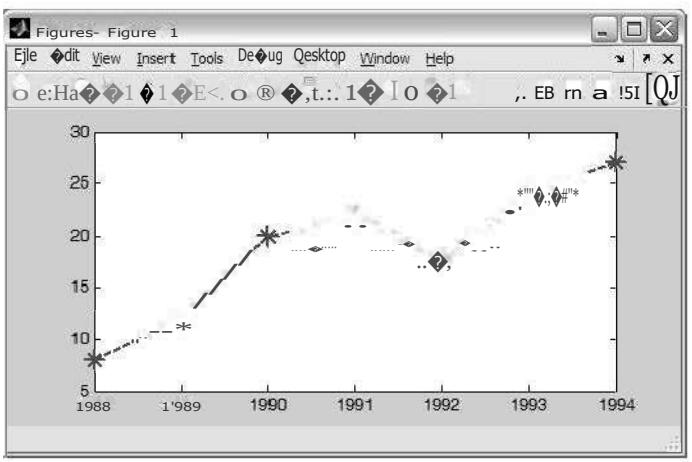


Figure 5-3: The Figure Wmdow with a plot ofthe lalel data.

5.1.2 Plot of a Function

In many situations there is a need to plot a given function. This can be done in MATLAB by using the plot or the fplot command. The use of the plot command is explained below. The fplot command is explained in detail in the next section.

Inordertoplot a functiony . ft.x) with the plot command, the user needs to first create a vector of values of x for the domain over which the function will be plotted. Then a vector y is created with the corresponding values of f(x) by using element-by-element calculations (see Chapter 3). Once the two vectors are defined, they can be used in the plot command

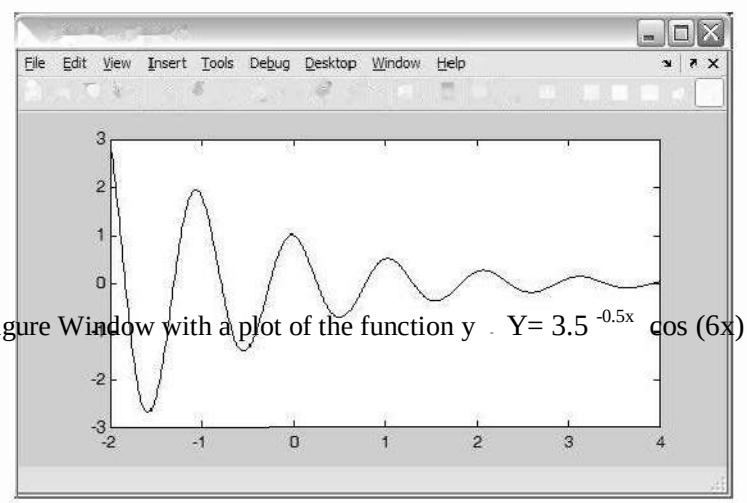
As an example, the plotcommand is used to plot the function $Y= 3.5^{-0.5x} \cos(6x)$ for $-2 \le x \le 4$. A program that plots this function is shown in the following script file.

```
% A script file that create• a plot of
% tha function: 3.5.^(-0.5•x).•cos(6x) X• [-2:0.
01:4] ,
                                    Create vector y with the function
                                    values at each point in x
                                    Plot y as a function of x
plot(x,y)
y=3.5.^(-0.5*x).*cos(6*x);
```

Once the script file is executed, the plot is created in the Figure Window, as shown in Figure S-4. Since the plot is made up of segments of straight lines that conect the points, to obtain an accurate plot of a function, the spacing between the elements of the vector x must be appropriate. Smaller spacing is needed for a

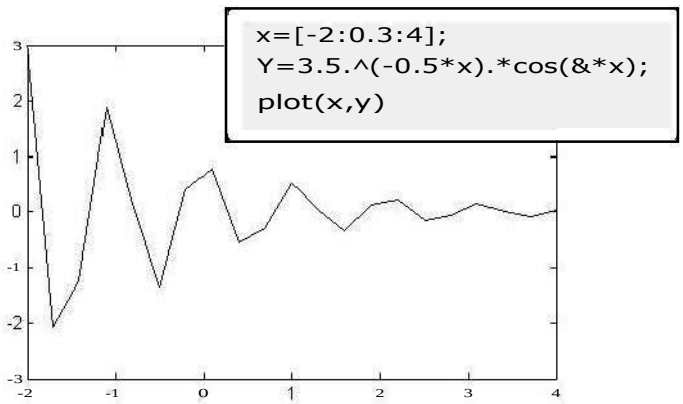
IPJ Figures- Figure 1

06 ▯▯▯▯ ▯I▯El0®▯/.·1@,1 D lEI• l1:!l EE rna e[Q]



FigunS-4: The Figure Window with a plot of the function y . $Y= 3.5^{-0.5x} \cos(6x)$

function that changes rapidly. In the last example a small spacing of 0.01 pro- duced the plot that is shown in Figure 54. However, if the same function in the same domain is plotted with much larger spacing-for example, 0.3--the plot that is obtained, shown in Figure 5-5, gives a distorted picture of the function. Note



F1gare 5-!: A plot of the function y - $Y = 3.5^{-0.5x} \cos(6x)$ with large spacing.

also that in Figure 54 the plot is shown with the Figure Window while in Figure 5-5 only the plot is shown. The plot can be copied from the Figure Window (in the Edit menu, select Copy Figure) and then pasted into other applications.

5.2 THE fplot COMMAND

The fplot command plots a function with the form y = f(x) between specified limits. The command has the form:

```
fplot('function',limits,'line specifiers')
```

The function to be plotted.

The domain of x and, optionally, the limits they axis.

Specifiers that define the type and color of the line of and markers (optional).

'function' : The function can be typed directly as a string inside the com- mand. For example, if the function that is being plotted is $f(x) = 8x^2 + 5\cos(x)$, it is typed as: '8*x^2+5*cos (x) '.The functions can include MATLAB built-in functions and functions that are created by the user.

- The function to be plotted can be typed as a function of any letter. For example, the function in the previous paragraph can be typed as '8* z^2 +5*cos(z) ' or '8*t^2+5*cos(t) '.

- The function cannot include previously defmed variables. For example, in the function above it is not possible to assign 8 to a variable, and then use the vari- able when the function is typed in the **fplot** command.

limits: The limits argument is a vector with two elements that specify the domain of x [xmin,xmax], or a vector with four elements that specifies the domain ofx and the limits of they-axis [xmin, xmax, ymin,ymax].

line specifiers: The line specifiers are the same as in the **plot** com- mand. For example, a plot of the function $y = x^2 + 4\sin(2x) - 1$ for $-3 \leq x \leq 3$ can be created with the **fplot** command by typing:

```
>> fplot('x^2+4*sin(2*x)-1', [-3 3])
```

in the Command Window. The figure that is obtained in the Figure Window is shown in Figure 5-6.

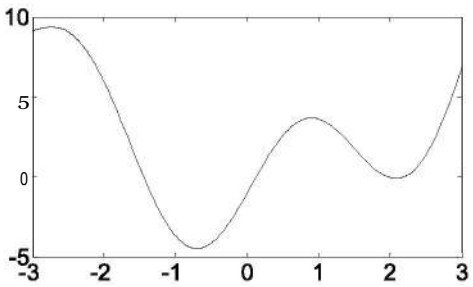


Figure 5-6: A plot of the function $y = x^2 + 4\sin(2x) - 1$.

5.3 PLOTTING MULTIPLE GRAPHS IN THE SAME PLOT

In many situations, there is a need to make several graphs in the same plot. This is shown, for example, in Figure 5-1 where two graphs are plotted in the same fig- ure. There are three methods to plot multiple graphs in one figure. One is by using the **plot** command, the second is by using the **hold on** and **hold off** com- mands, and the third is by using the **line** command.

5.3.1 Using the plot Command

Two or more graphs can be created in the same plot by typing pairs of vectors inside the **plot** command. The command

plot(x,y,u,v,t,h)

creates three graphs-y vs. x, v vs. u, and h vs. t–all in the same plot. The vec- tors of each pair must be of the same length. MATLAB automatically plots the graphs in different colors so that they can be identified. It is also possible to add line specifiers following each pair. For example the command

plot(x,y, '-b',u,v, '--r',t,h, 'g:')

plots y vs. x with a solid blue line, v vs.u with a dashed red line, and h vs. t with a dotted green line.

Sample Problem 5-1: Plotting a function and its derivatives

Plot the function $y = 3x^3 - 26x + 10$, and its first and second derivatives, for $-2 \leq x \leq 4$, all in the same plot.

Solution

The first derivative of the function is: $y' = 9x^2 - 26$.

The second derivative of the function is: $y'' = 18x$.

A script file that creates a vector x and calculates the values of y, y', and y'' is:

```
X=[-2:0.01:4];
y=3*x.^3-26*x+6;
yd=9*x.^2-26;
ydd=l8*x;
plot(x,y,'-b',x,yd,'--r',x,ydd,':k')
```

Create vector x with the domain of the function.

Create vector y with the function value at each x.

Create vector yd with values of the first derivative.

Create vector ydd with values of the second derivative.

Create three graphs, y vs. x, yd vs. x, and ydd vs. x, in the same figure.

The plot that is created is shown in Figure 5-7.

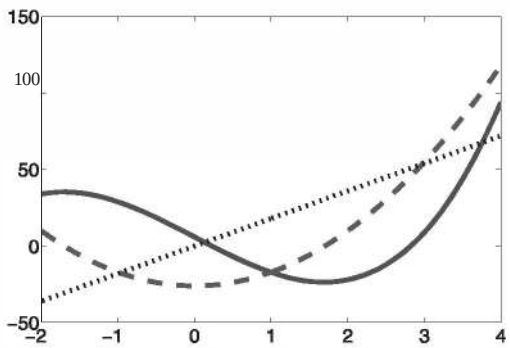


Figure 5-7: A plot of the function $y = 3x^3 - 26x + 10$ and its first and second derivatives.

5.3.2 Using the hold on and hold off Commands

To plot several graphs using the hold on and hold off commands, one graph is plotted first with the plot command. Then the hold on command is typed. This keeps the Figure Window with the first plot open, including the axis proper-

ties and formatting (see Section 5.4) if any was done. Additional graphs can be added with plot commands that are typed next. Each plot command creates a graph that is added to that figure. The hold off command stops this process. It returns MATLAB to the default mode, in which the plot command erases the previous plot and resets the axis properties.

As an example, a solution of Sample Problem 5-l using the hold on and hold off commands is shown in the following script file:

```
X=[-2:0.01:4];
y=3*x.^3-26*x+6;
yd=9*x.^2-26;
ydd=18*x;
plot(x,y,'-b')
hold on
plot(x,yd,'--r')
plot(x,ydd,':k')
hold off
```

The first graph is created.

Two more graphs are added to the figure.

5.3.3 Using the line Command

With the line command additional graphs (lines) can be added to a plot that already exists. The form of the line command is:

```
line(x,y,'PropertyName',PropertyValue)
```

(Optional) Properties with values that can be used to specify the line style, color, and width, marker type, size, and edge and fill colors.

The format of the line command is almost the same as the plot command (see Section 5.1). The line command does not have the line specifiers, but the line style, color, and marker can be specified with the Property Name and property value features. The properties are optional, and if none are entered MATLAB uses default properties and values. For example, the command:

```
line(x,y,'linestyle','--','color','r','marker','o')
```

will add a dashed red line with circular markers to a plot that already exists.

The major difference between the plot and line commands is that the plot command starts a new plot every time it is executed, while the line command adds lines to a plot that already exists. To make a plot that has several graphs, a plot command is typed first and then line commands are typed for additional graphs. (If a line command is entered before a plot command, an error message is displayed.)

The solution to Sample Problem 5-1, which is the plot in Figure 5-7, can be obtained by using the **plot** and **line** commands as shown in the following script file:

```
X=(-2:0.01:4];
y=3*x."3-26*x+6;
yd=9*x."2-26;
ydd=l8*x;
plot(x,y,'LineStyle•,•--•,'color','b')
line(x,yd,'LineStyle•,•---•,'color','r')
line(x,ydd,'linestyle','::','color','k')
```

5.4 PLOTS WITH LOGARITHMIC AXES

Many science and engineering applications require plots in which one or both axes have a logarithmic (log) scale. Log scales provide means for presenting data over a wide range of values. It also provides a tool for identifying characteristic of data and possible forms of mathematical relationships that can be appropriate for modeling the data .

MATLAB commands for making plots with log axes are:

semilogy (x, y)	Plots y versus x with a log (base 10) scale for they axis and linear scale for the x axis.
semilogx (x, y)	Plots y versus x with a log (base 10) scale for thex axis and linear scale for they axis.
loglog (x, y)	Plots y versus x with a log (base 10) scale for both axes.

Line specifiers and Property Name and Property Value arguments can be added to the commands (optional) just as in the **plot** command. As an example, Figure 5-9 shows a plot of the function $y = 2^{(-0.2x+I0)}$ for $0.1 \leq x \leq 60$. The figure shows four plots of the same function: one with linear axes, one with a log scale for the y axis, one with a log scale for the x axis, and one with a log scale on both axes.

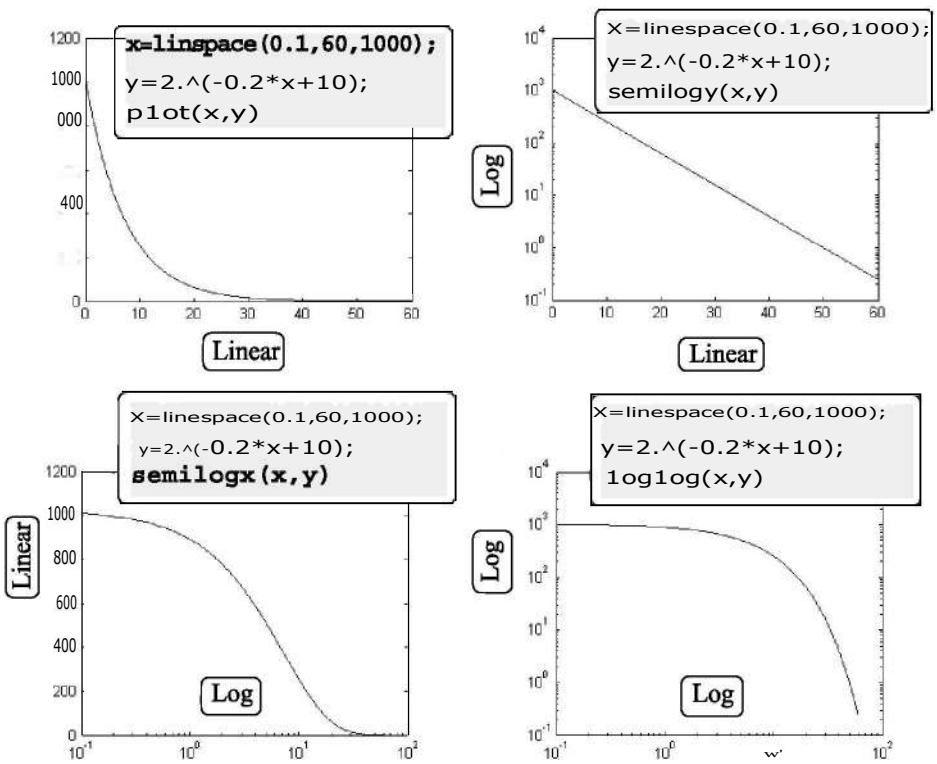


Figure 5-9: Plots of $y = 2^{(0.2x+10)}$ with linear, semilog, and log-log scales.

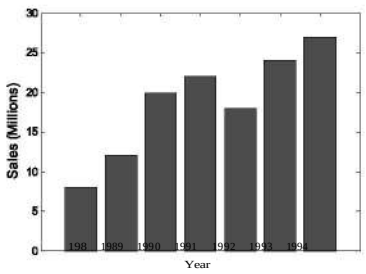
Notes for plots with logarithmic axes:

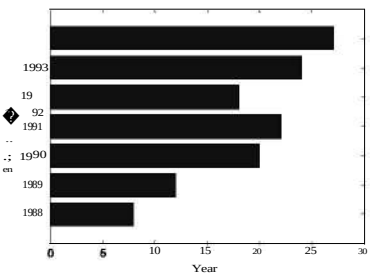
- The number zero cannot be plotted on a log scale (since a log of zero is not defined).
- Negative numbers cannot be plotted on log scales (since a log of a negative number is not defined).

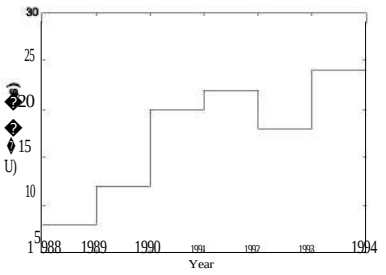
5.5 PLOTS WITH SPECIAL GRAPHICS

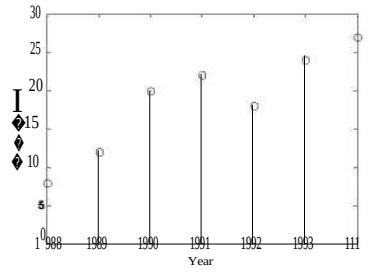
All the plots that have been presented so far in this chapter are line plots in which the data points are connected by lines. In many situations plots with different graphics or geometry can present data more effectively. MATLAB has many options for creating a wide variety of plots. These include bar, stairs, stem, and pie plots and many more. Fallowing are some of the special graphics plots that can be created with MATLAB. A complete list of the plotting functions that MATLAB offers and information on how to use them can be found in the Help Window. In this window first choose "Functions by Category," then select "Graphics" and then select "Basic Plots and Graphs" or "Specialized Plotting."

Bar (vertical and horizontal), stairs, and stem plots are presented in the fol- lowing charts using the sales data from Section 5.1.1.

Vertical Bar Plot
 	
Function format:
 	
bar(x,y)
		yr=[1988:1994];
		sle=[8 12 20 22 18 24 27];
		bar(yr,sle,'r')
		x1abe1('Year')
		ylabel('Sales (Millions) ')
		The bars are in red.

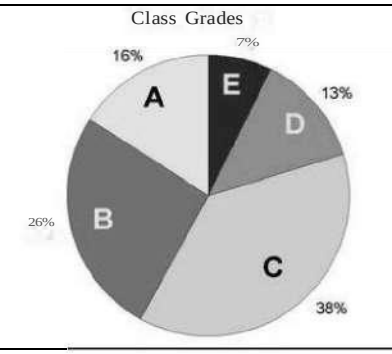
Horizontal Bar Plot
 	
Function format:
 	
barh(x,y)
		yr=[1988:1994];
		sle=[8 12 20 22 18 24 27];
		barh(yr,sle)
		x1abe1(1Sales (Millions) 1)
		ylabel('Year')

Stairs Plot Function format: stairs(x,y)		yr=[1988:1994]; s1e=[8 12 20 22 18 24 27] stairs(yr,s1e)

Stem Plot
 	
Function Format
 	
stem(x,y)
		yr=[1988:1994];
		s1e=[8 12 20 22 18 24 27];
		stem(yr,sle)

Pie charts are useful for visualizing the relative sizes of different but related quantities. For example, the table below shows the grades that were assigned to a class. The data is used to create the pie chart that follows.

Grade	A	B	C	D	E
Number of students	11	18	26	9	5

Pie Plot Function format: pie (x)		grd=[11 18 26 9 5]; pie(grd) tit1e('C1ass Grades') MATLAB draws the sections in different col- ors. The letters (grades) were added using the Plot Editor.

5.6 HISTOGRAMS

Histograms are plots that show the distribution of data. The overall range of a given set of data points is divided into subranges (bins), and the histogram shows how many data points are in each bin. The histogram is a vertical bar plot in which the width of each bar is equal to the range of the corresponding bin and the height

of the bar corresponds to the number of data points in the bin. Histograms are cre- ated in MATLAB with the **hist** command. The simplest form of the command lS:

```
hist (y)
```

y is a vector with the data points. MATLAB divides the range of the data points into 10 equally spaced subranges (bins) and then plots the num- ber of data points in each bin.

For example, the following data points are the daily maximum temperature (in °F) in Washington, DC, during the month of April 2002: 58 73 73 53 50 48 56 73 73 66 69 63 74 82 84 91 93 89 91 80 59 69 56 64 63 66 64 74 63 69 (data from the U.S. National Oceanic and Atmospheric Administration). A histogram of this data is obtained with the commands:

```
>> y=[58 73 73 53 50 48 56 73 73 66 69 63 74 82 84 91 93 89
91 80 59 69 56 64 63 66 64 74 63 69];
>> hist(y)
```

The plot that is generated is shown in Figure 5-11 (the axis titles were added using the Plot Editor). The smallest value in the data set is 48 and the largest is 93, which means that the range is 45 and the width of each bin is 4.5. The range of the first bin is from 48 to 52.5 and contains two points. The range of the second bin is from 52.5 to 57 and contains three points, and so on. Two of the bins (75 to 79.5 and 84 to 88.5) do not contain any points.

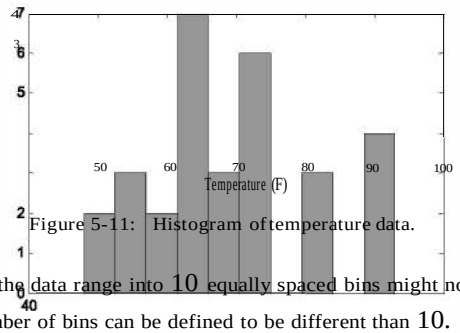


Figure 5-11: Histogram of temperature data.

Since the division of the data range into 10 equally spaced bins might not be the division that is preferred by the user, the number of bins can be defined to be different than 10. This can be done either by specifying the number of bins, or by specifying the center point of each bin as shown in the following two forms of the

hist command:

```
hist(ylnbins)
```

or

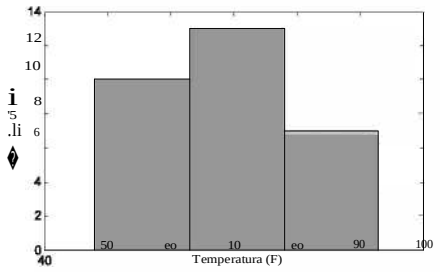
```
hist(y,x)
```

nbins is a scalar that defmes the number of bins. MATLAB divides the range in equally spaced subranges.

x is a vector that specifies the location of the center of each bin (the distance between the centers does not have to be the same for all the bins). The edges of the bins are at the middle point between the centers.

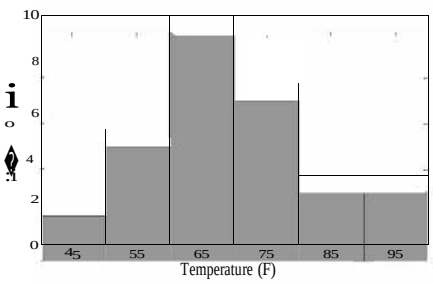
In the example above the user might prefer to divide the temperature range into three bins. This can be done with the command:

```
>> hist(y,3)
```



As shown in the top graph, the histogram that is generated has three equally spaced bins.

The number and width of the bins can also be specified by a vector x whose elements defme the centers of the bins. For example, shown in the lower graph is a histogram that displays the temperature data from above in six bins with an equal width of 10 degrees. The elements of the vector x for this plot are 45, 55, 65, 75, 85, and 95. The plot was obtained with the following commands:




```
>> X=[45:10:95]
X =
    45    55    65    75    85    95
>> hist(y,x)
```

The hist command can be used with options that provide numerical out- put in addition to plotting a histogram. An output of the number of data points in each bin can be obtained with one of the following commands:

```
n=hist(y)
```

```
n=hist(y,nbins)
```

```
n=hist(y,x)
```

The output n is a vector. The number of elements in n is equal to the number of bins, and the value of each element of n is the number of data points (frequency count) in the corresponding bin. For example, the histogram in Figure 5-11 can

also be created with the following command:

```
>> n = hist(y)
n =
     2     3     2     7     3     6     0     3     0     4
```

The vector n shows how many elements are in each bin.

The vector n shows that the first bin has two data points, the second bin has three data points, and so on.

An additional, optional numerical output is the location of the bins. This output can be obtained with one of the following commands:

```
[n xout] =hist (y)
```

```
[n xout]=hist(y,nbins)
```

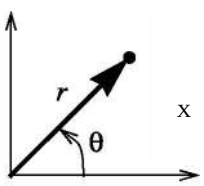
xout is a vector in which the value of each element is the location of the center of the corresponding bin. For example, for the histogram in Figure 5-11:

```
>> [n xout]=hist(y)
n =
     2     3     2     7     3     6     0     3     0     4
xout
   50.2500   54.7500   59.2500   63.7500   68.2500   72.7500
77.2500   81.7500   86.2500   90.7500
```

The vector xout shows that the center of the first bin is at 50.25, the center of the second bin is at 54.75, and so on.

5.7 POLAR PLOTS

Polar coordinates, in which the position of a point in a plane is defmed by the angle e and the radius (distance) to the point, are frequently used in the solution of science and engineering problems. The polar command is used to plot functions in polar coordinates. The command has the form:




```
polar(theta,radius,'line specifiers')
```

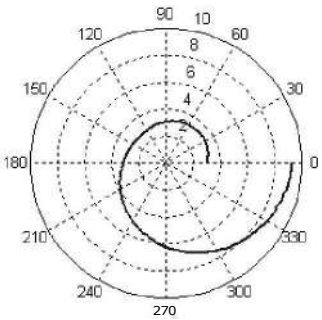
Vector Vector (Optional) Specifiers that defme the type and color of the line and markers.

where theta and radius are vectors whose elements define the coordinates of the points to be plotted. The polar command plots the points and draws the polar grid. The line specifiers are the same as in the plot command. To plot a function $r = /(8)$ in a certain domain, a vector for values of8 is created first, and then a vector r with the corresponding values of [(8) is created using element-by-

element calculations. The two vectors are then used in the polar command.

For example, a plot of the function $r = 3\cos 2(0.58) + e$ for $0 \le e \le 27t$ 1s shown below.

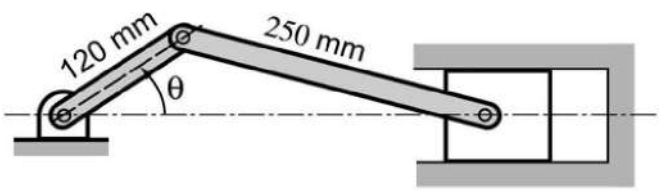
```
t=linspace(0,2*pi,200);
r=3*cos(0.5*t).A2+t;
polar(t,r)
```



5.12 EXAMPLES OF MATLAB APPliCATIONS

Sample Problem 5-2: Piston-crank mechanism

The piston-rod-crank mechanism is used in many engineering applications. In the mechanism shown in the following figure, the crank is rotating at a constant speed of500 rpm.



Calculate and plot the position, velocity, and acceleration of the piston for one revolution of the crank. Make the three plots on the same page. Set a = oo when t=O.

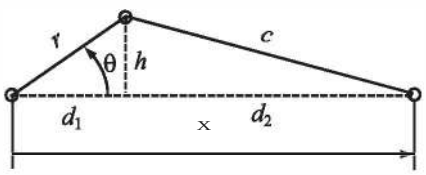
Solution

The crank is rotating with a constant angular velocity e. This means that if we set a = oo when t= 0, then at time t the angle a is given by a = St, and means that e = 0 at all times.

The distances d1 and h are given by:

d1 = rcosa and h = rsina

With h known, the distance d2 can be calculated using the Pythagorean Theorem:



$$d_2 = (c^2 - h^2)^{1/2} = (c^2 - r^2\sin^2\theta)^{1/2}$$

The position x of the piston is then given by:

$$x = d_1 + d_2 = r\cos a + (c^2 - r^2\sin^2 a)^{1/2}$$

The derivative of x with respect to time gives the velocity of the piston:

$$\dot{x} = -ra\,\dot{s}\,\dot{n}a - \frac{r^2 S\sin 2a}{2(c^2 - r^2\sin^2 a)^{1/2}}$$

The second derivative of x with respect to time gives the acceleration of the pis- ton:

$$\ddot{x} = -ra^2\cos a - \frac{4r^2 a^2\cos 2\theta(c^2 - r^2\sin^2\theta) + (r^2\theta\sin 2\theta)}{4(c^2 - r^2\sin^2 a)^{3/2}}$$

In the equation above, e was taken to be zero.

A MATLAB program (script file) that calculates and plots the position, velocity, and acceleration of the piston for one revolution of the crank is shown below.

```
THDrpm=SOO; r=O.l2; c=0.25;
THD=THDrpm*2*pi/60;                    the units of e from rpm to
tf=2*pi/THD;                           the time for one revolution of the
t=linspace(0,tf,200);                  a vector for the time with 200
TH=THD*t;                              Calculatee for each t.
d2s=c"2-r"2*sin(TH)."2;                Calculate dz squared for each
x=r*cos(TH)+sqrt(d2s);
xd=-r*THD*sin(TH)-(r"2*THD*sin(2*TH))./(2*sqrt(d2s));
```

Define θ, *r*, and *c*.

Change rad/s.

Calculate crank.

Create elements.

θ.

Calculate *x* for each θ.

```
xdd=-r*THDA2*cos(TH)-(4*rA2*THDA2*cos(2*TH).*d2s+
(r^2*sin(2*TH)*THD).^2)./(4*d2s.^(3/2));
subplot(3,1,1)                          Calculate ẋ and ẍ for each θ.
plot(t,x)                                       Plotx vs. t.
grid                                    Format the first plot.
xlabel('Time (s)')
ylabel('Position (m)')
subplot(3,1,2)
plot(t,xd)                                      Plot x vs. t.
grid                                    Fonnat the second plot.
xlabel('Time (s)')
ylabel('Velocity (m/s)')
subplot(3,1,3)
plot(t,xdd)                                     Plot x vs. t.
grid                                    Fonnat the third plot.
xlabel('Time (s)')
ylabel('Acceleration (m/sA2)')
```

When the script file uns it generates the th ee plots on the same page as shown in Figu e 5-13. The fig re nicely shows that the velocity of the piston is zero at the end points of the travel range where the piston changes the di ection of the motion. The acceleration is maxim m (di ected to the left) when the piston is at the right end.

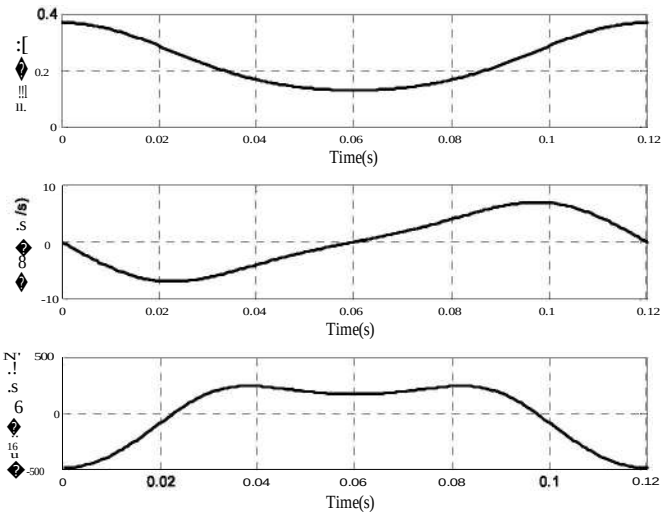
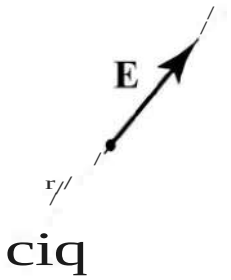


Figure 5-15: Position, velocity, and acceleration of the piston vs. time.

Sample Problem 5-3: Electric Dipole

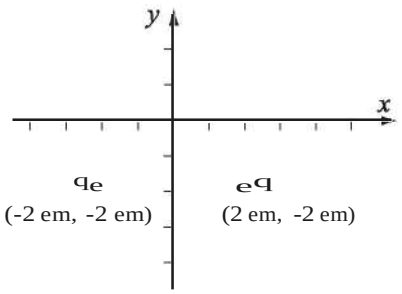
The electric field at a point due to a charge is a vector E with magnitude E given by Coulomb's law:

$$E = \frac{1}{4\pi\varepsilon_0}\frac{q}{r^2}$$



where $\varepsilon_0 = 8.8541878 \times 10^{-12}\ C^2/N\cdot m^2$ is the permittivity constant, q is the magnitude of the charge, and r is the distance between the charge and the point. The direction of E is along the line that connects the charge with the point. E points outward from q if q is positive, and toward q if q is negative. An electric dipole is created when a positive charge and a negative charge of equal magnitude are placed some distance apart. The electric field, E, at any point is obtained by superposition of the electric field of each charge.

An electric dipole with q = 12 x I0-9 C is created, as shown in the figure. Determine and plot the magnitude of the electric field along the x axis from x = -5 em to x =Scm.

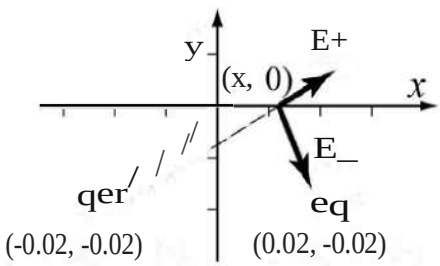


Solution

The electric field E at any point (x, 0) along the x axis is obtained by adding the electric field vectors due to each of the charges.

$$E = E_- + E_+$$

The magnitude of the electric field is the length of the vector E.



The problem is solved by following these steps:

Step 1: Create a vector x for points along the x axis.

Step 2: Calculate the distance (and distance squared) from each charge to the points on the x axis.

$$r_{minus} = \sqrt{(0.02 - x)^2 + 0.022} \qquad r_{plus} = \sqrt{(x + 0.02x)^2 + 0.022}$$

Step 3: Write unit vectors in the direction from each charge to the points on the x axis.

$$E_{minusUV} = \frac{1}{7minus}((0.02 - x)\mathbf{i} - 0.02\mathbf{j})$$

$$E_{plusUV} = \frac{1}{7plus}((x + 0.02)\mathbf{i} + 0.02\mathbf{j})$$

Step 4: Calculate the magnitude of the vector E_ and E+ at each point by using Coulomb's law.

$$E_{minusMAG} = \frac{1}{4\pi\varepsilon_0}\frac{q}{r_{minus}^2} \qquad E_{plusMAG} = \frac{1}{4\pi\varepsilon_0}\frac{q}{r_{plus}^2}$$

Step 5: Create the vectors E_ and E_+ by multiplying the unit vectors by the magnitudes.

Step 6: Create the vector E by adding the vectors E_ and E+.

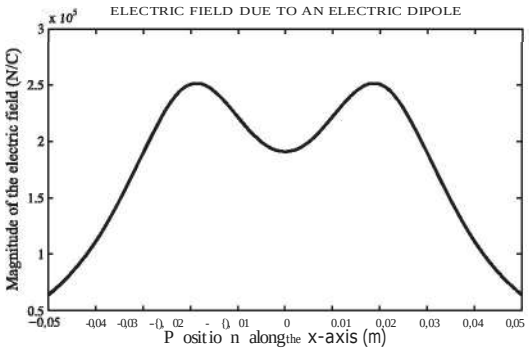
Step 7: Calculate E, the magnitude (length) of E.

Step 8: Plot E as a function of x.

A program in a script file that solves the problem is:

```
q=12e-9;
epsilon0=8.8541878e  -12;
X=[ -0.05:0.001:0.05]';                          Create a column vector x.
r.minusS=(0.02-x).A2+0.02A2;
r.minus=sqrt(r.minusS);                          Step 2. Each variable is a column vector.
rplUSS=(X+0.02).  A2+0.02A2;
rplus=sqrt( rplusS);                             Steps 3 & 4. Each variable is a two column matrix. Each row is the vector for the corresponding x.
lmlinusUV=[((0.02-x)./:cninus), (-0.02./z:minus)];
!plusUV=[((x+0.02)./rplus), (0.02./rplus)];
Eminus=(q/(4*pi*epsilonO))./rminusS;
EplusMAG=(q/(4*pi *epsilonO))./ rplusS;
Eminus=[Eminus.*EminusUV(:,1),  EminusMAG.*EminusUV(:, 2)];
Eplus=[EplusMAG.*EplusUV(:,1),  EplusMAG.*EplusUV(:, 2)];
E=Eminus+Eplus;                                  Step 6.
EMAG=sqrt(E(:,1).A2+E(:,2).A2);                  Step 7.        Step 5.
plot(x ,EMAG,'k','linewidth',1)
xlabel('Position along the x-axis (m)','FontSize',12)
ylabel('Magnitude of the electric field (N/C)','FontSize',12)
title('ELECTRIC FIELD DUE TO AN ELECTRIC DIPOLE','FontSize',12)
```

When this script file is executed in the Command Window, the following figure is created in the Figure Window:



5.13 PROBLEMS

1. Plot the function $f(x) = \frac{x^2 - 3x + ?}{\sqrt{2x + 5}}$ for $-1 \leq x \leq 5$.

2. Plot the function $f(x) = (3\cos x - \sin x)e^{-0.2x}$ for $-4 \leq x \leq 9$.

3. Plot the function $f(x) = \frac{x^2}{2 + \sin x + x}$ for $-4 \leq x \leq 4$.

4. Plot the function $f(x) = \frac{1}{2}x^2 - 10\sin 2x - e^{0.9x}$ and its derivative for $-2 \leq x \leq 4$ in one figure. Plot the function with a solid line, and the derivative with a dashed line. Add a legend and label the axes.

5. Make two separate plots of the function $f(x) = -3x^4 + 10x^2 - 3$, one plot for $-4 \leq x \leq 3$ and one for $-4 \leq x \leq 4$.

6. Use the fplot command to plot the function
   $f(x) = (\sin 2x + \cos 25x)e^{-0.2x}$ in the domain $-6 \leq x \leq 6$.

7. Plot the function $f(x) = \sin^2(x)\cos(2x)$ and its derivative, both on the same plot, for $0 \leq x \leq 2\pi$. Plot the function with a solid line, and the derivative with a dashed line. Add a legend and label the axes.

8. Make a plot of a circle with its center at (4.2, 2.7) and radius of 7.5.

9. A parametric equation is given by
   $$x = \sin(t)\cos(t), \quad y = 1.5\cos(t)$$
   Plot the function for $-1t \leq t \leq 1t$. Format the plot such that the both axes will range from -2 to 2.

10. Two parametric equations are given by:

$$x = \cos 3(t), \quad y = \sin 3(t)$$

$$u = \sin(t), \quad v = \cos(t)$$

In one figure, make plots of y versus x and v versus u for $0 \le t \le 21t$. Format the plot such that the both axes will range from -2 to 2.

11. Plot the funetlan $f(x) = \dfrac{x2 - 5x - 12}{x2 - x - 6}$ m the domam $-1 \le x \le 7$. Notlce that the function has a vertical asymptote at $x = 3$. Plot the function by creating two vectors for the domain of x. The first vector (name it x1) includes ele- ments from -1 to 2.9, and the second vector (name it x2) includes elements from 3. 1 to 7. For each x vector create ay vector (name themy1 and y2) with the corresponding values of y according to the function. To plot the function make two curves in the same plot {y1 vs. x1, and y2 vs. x2). Format the plot such that they-axis ranges from-20 to 20.

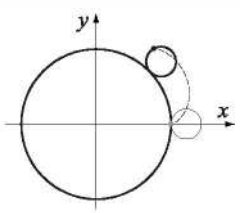
12. Plot the function $f(x) = \dfrac{x^2 + 3x - 5}{x^2 - 3x - 10}$ for $-4 \le x \le 9$. Notice that the function has two vertical asymptotes. Plot the function by dividing the domain of x into three parts: one from -4 to near the left asymptote, one between the two asymptotes, and one from near the right asymptote to 9. Set the range of they axis from -20 to 20.

13. A parametric equation is given by:

$$x = \frac{3t}{1+t3'} \qquad y = \frac{3t2}{1+t3}$$

(Note that the denominator approaches 0 when t approaches-1.) Plot the func- tion (the plot is called the Folium of Descartes) by plotting two curves in the same plot- one for $-30 \le t \le -1.6$ and the other for $-0.6 \le t \le 40$.

14. An epicycloid is a curve (shown partly in the figure) obtained by tracing a point on a circle that rolls around a fixed circle. The parametric equation of a cycloid is given by:

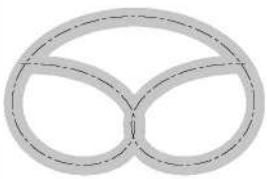


$$x = 13\cos(t) - 2\cos(6.5t)$$

$$y = 13\sin(t) - 2\sin(6.5t)$$

Plot the cycloid for $0 \le t \le 41t$.

15. The shape of the pretzel shown is given by the following parametric equations:

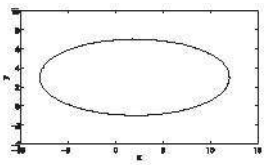


$x = (3.3 - 0.4t2)\sin(t) \quad y = (2.5 - 0.3t2)\cos(t)$

where $-4 \le t \le 3$. Make a plot of the pretzel.

16. Make a polar plot of the function r = 2sin(38)sin8 for 0 s e s 27t.

17. Plot an ellipse with major axes of a = 10 and b = 4 and a center at x = 2 andy= 3.



18. The following data gives the approximate populati of the world for selected years ftom 1850 unti12000.

Year	1850	1910	1950	1980	2000	2010
Population (billions)	1.3	1.75	3	4.4	6	6.8

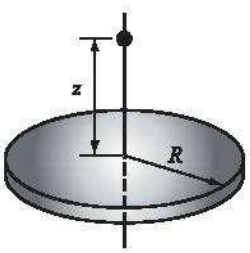
The populati P, since 1900 can be modeled by the logistic function:

$$p = \frac{11.55}{1 + 18.7e\text{-}0.^{0}193t}$$

where Pis in billions and tis years since 1850. Make a plot of population ver- sus years. The figure should show the information from the table above as datapoints and the population modeled by dle equation as a solid line. Set the range of the horizont axis ftom 1800 to 2200. Add a legend, and label the axes.

19. The force F (inN) acting between a particle with a charge q and a round disk with a radius R and a charge Q is given by the equation:

$$F = \frac{}{2e_0}\left(1 - \frac{z}{\sqrt{z^2 + R^2}}\right)$$

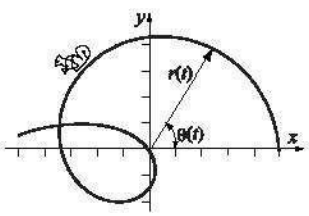


where £o = 0.885 X 10-12 C2/(Nm2) is the permittiv- ity constant and z is the distance to the particle. Consider the case where Q = 9.4 x 10-6 C, q = 2.4 x 10-s C, and R = 0.1 m. Make a plot of F as a function of z for O z 0.3 m. Use MATLAB's built-in function max to find the maximum value ofFand the corresponding distance z.

20. The position as a function of time of a squirrel nmning on a grass field is given in polar coordinates by:

$$r(t) = 25 + 30[1 - e\sin(0.07t)] \text{ m}$$

$$O(t) = 27t(1 - e\text{-}4.2t)$$

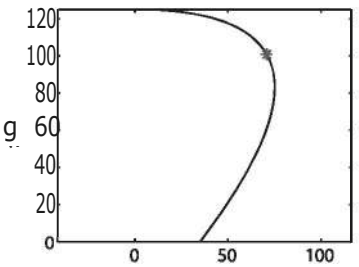


(a) Plot dle trajectory (position) of the squirl for O t 20 s.

21. Consider the motion of the squirrel in the previous problem. The components of the velocity vector of the squirrel are given by v_r = [illegible] and $v\theta$ = r[illegible]. The speed of the squirrel is v = $\sqrt{v_r^2 + vij}$. Plot for the speed of the squirrel as a function of time for $0 \leq t \leq 20$ s.

22. The curvilinear motion of a particle is defmed by the following parametric equa- tions:

$x = 52t - 9t2$ m and $y = 125 - 5t2$ m The velocity of the particle is given by $v = \sqrt{v_x^2 + v_y^2}$, where $v_x = \frac{dx}{dt}$ and $v_Y = \frac{cJ}{dt}$.



For $0 \leq t \leq 5$ s make one plot that shows the position of the particle (y versus x), and a second plot (on the same page) of the velocity of the particle as a function of time. In addition, by using MATLAB's min function, determine the time at which the velocity is the lowest, and the corresponding position of the parti- cle. Using an asterisk marker, show the position of the particle in the ftrst plot. For time use a vector with spacing of 0.1 s.

23. The demand for water during a ftre is often the most important factor in the design of distribution storage tanks and pumps. For communities with popula- tions less than 200,000, the demand Q (in gallons/min) can be calculated by:

$$Q = 1020\$(1 - 0.0L/P)$$

where P is the population in thousands. Plot the water demand Q as a function of the population P (in thousands) for $0 \leq P \leq 200$. Label the axes and provide a title for the plot.

24. The position x as a function of time of a particle that moves along a straight line is given by:

$$x(t) = (-3 + 4t)e^{-0.4t} \text{ ft}$$

The velocity v(t) of the particle is determined by the derivative of x(t) with respect to t, and the acceleration a(t) is determined by the derivative of v(t) with respect to t.

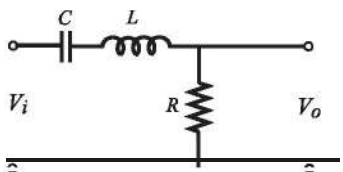
Derive the expressions for the velocity and acceleration of the particle, and make plots of the position, velocity, and acceleration as functions of time for $0 \leq t \leq 20$ s. Use the **subplot** command to make the three plots on the same page with the plot of the position on the top, the velocity in the middle, and the acceleration at the bottom. Label the axes appropriately with the cor- rect units.

25. The area of the aortic valve, A_V in cm^2, can be estimated by the equation (Hakki Formula):

$$A_V = \frac{Q}{\sqrt{PG}}$$

where Q is the cardiac output in L/min, and PG is the difference between the left ventricular systolic pressure and the aortic systolic pressure (in mm Hg). Make one plot with two curves of A_V versus PG, for 2:5: PG:5: 60 mm Hg- one curve for Q = 4 Llmin and the other for Q = 5 Llmin. Label the axes and use a legend.

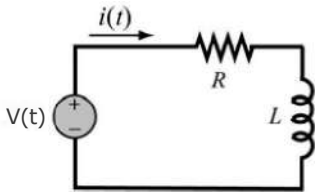
26. A bandpass filter passes signals with frequencies that are within a certain range. In this filter the ratio of the magnitudes of the voltages is given by



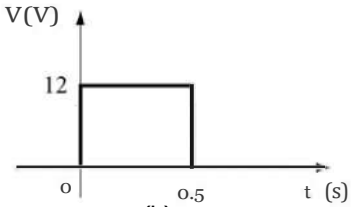
$$RV = \left|\frac{V_o}{V_i}\right| = \frac{mRC}{\sqrt{(1-m^2LC)^2+(mRC)^2}}$$

where m is the frequency of the input signal. Given R = 200 n, L = 8 mH, and C = 5 J.F, make two plots of RV as a function of m for 10:5: m:5: 500000. In the first plot use linear scale for both axis, and in the second plot use logarithmic scale for the horizontal (m) axis, and linear scale for the vertical axis. Which plot provides a better illustration of the filter?

27. A resistor, R = 4 n, and an inductor, L = 1.3 H, are connected in a circuit to a voltage source as shown in Figure (a) (an RL circuit). When the voltage



(a)



(b)

12

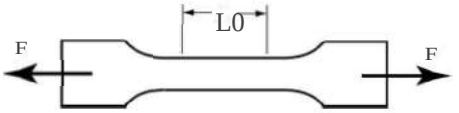
source applies a rectangular voltage pulse with an amplitude of V= V and a duration of 0.5 s, as shown in Figure (b), the curent i(t) in the circuit as a function of time is given by:

$$i(t) = \frac{12}{4}\left(1-e^{<-Rt)IL}\right) \quad \text{for } 0:5: t:5: 0.5 \text{ s}$$

$$i(t) = e^{-(Rt)IL}\frac{12}{4}\left(e^{(O.SR)IL}-1\right) \quad \text{for } 0.5:5: t \text{ s}$$

Make a plot of the curent as a function of time for 0:5: t:5: 2 s.

28. In a typical tension test a dog bone shaped specimen is pulled in a machine. During the test, the force F needed to pull the specimen and the length L of a gauge section are measured. This data is used for plotting a stress-strain diagram of the material. Two defmitions, engineering and true, exist for stress and strain. The engineering stress σ_e and strain ε_e are defmed by



$\sigma_e = \frac{F}{A_0}$ and $\varepsilon_e = \frac{L - L_0}{L_0}$, where L_0 and A_0 are the initial gauge length and the initial cross-sectional area of the specimen, respectively. The true stress σ_t and strain ε_t are defined by $\sigma_t = \frac{F}{A_o}\frac{L}{L_o}$ and $\varepsilon_t = \ln\frac{L}{L_o}$.

The following are measurements of force and gauge length from a tension test with an aluminum specimen. The specimen has a round cross section with radius 6.4 mm (before the test). The initial gauge length is $L_0 = 25$ mm. Use the data to calculate and generate the engineering and true stress-strain curves, both on the same plot. Label the axes and use a legend to identify the curves. Units: When the force is measured in newtons (N) and the area is calculated in m2, the unit of the stress is pascals (Pa).

F(N)	0	13,031	21,485	31,963	34,727	37,119	37,960	39,550
L(mm)	25.4	25.474	25.515	25.575	25.615	25.693	25.752	25.978
F(N)	40,758	40,986	41076	41,255	41,481	41,564		
L(mm)	26.419	26.502	26.600	26.728	27.130	27.441		

29. According to special relativity, a rod of length L moving at velocity v will shorten by an amount 8 , given by:

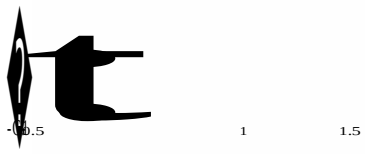
$$8 = L\left(1 - \sqrt{1 - \frac{v^2}{c^2}}\right)$$

where c is the speed of light (about 300×10^6 m/s). Consider a rod of 2 m long, and make three plots of 8 as a function of v for $0 \le v \le 300 \times 10^6$ m/s. In the first plot use linear scale for both axes. In the second plot use logarithmic scale for v and linear scale for 8 , and in the third plot use logarithmic scale for both v and 8 . Which of the plots is the most informative?

30. The shape of a symetrical four-digit NACA airfoil is described by the equation

$$y = \pm \frac{t}{0.2} [0.2969\sqrt{x/c} - 0.1260(x/c) - 0.3516(x/c)^2 + 0.2843(x/c)^3 - 0.1015(x/c)^4]$$

where c is the cord length and t is the maximum thickness as a fraction of the cord length (t_c = maximum thickness). Symmetrical four-digit NACA airfoils are designated NACA OOXX; where XX is $100t$ (i.e., NACA 0012 has $t = 0.12$). Plot the shape of a NACA 0020 airfoil with a cord length of 1.5 m.

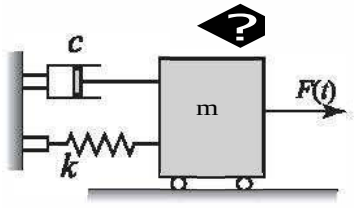


31. The ideal gas law relates the pressure P, volume V, and temperature T of an ideal gas:

$$PV = nRT$$

where n is the number of moles and $R = 8.3145$ J/(K. mol). Plots of pressure versus volume at constant temperature are called isotherms. Plot the isotherms for one mole of an ideal gas for volume ranging from 1 to 10 m3, at temperatures of $T = 100, 200, 300,$ and 400 K (four curves in one plot). Label the axes and display a legend. The units for pressure are Pa.

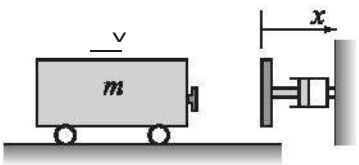
32. The vibrations of the body of a helicopter due to the periodic force applied by the rotation of the rotor can be modeled by a frictioness [illegible] from external periodic force. The position x(t) of the mass is given by the equation:



$$x(t) = \frac{2F_0}{\omega_n^2 - \omega^2} \sin\left(\frac{\omega_n - \omega}{2} t\right) \sin\left(\frac{\omega_n + \omega}{2} t\right)$$

where $F(t) = F_0 \sin \omega t$, and $f_0 = F_0/m$, ω is the frequency of the applied force, and ω_n is the natural frequency of the helicopter. When the value of ω is close to the value of ω_n, the vibration consists of fast oscillation with slowly changing amplitude called beat. Use $F_0/m = 12$ N/kg, $\omega_n = 10$ rad/s, and $\omega = 12$ rad/s to plot x(t) as a function of t for $0 \le t \le 10$ s.

33. A railroad bumper is designed to slow down a rapidly moving railroad car. After a 20,00 kg railroad car traveling at 20 mls engages the bumper, its displacement x (in meters) and velocity v (in mls) as a function of timet (in seconds) is given by:

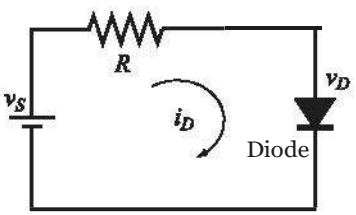


$$x(t) = 4.219(e\text{-}1.581\text{-}e\text{-}6\cdot321) \text{ and } v(t) = 26.67e\text{-}6.321\text{-}6.67e\text{-}1.5Bt$$

Plot the displaceme and the velocity as a fimction of time for 0st s4s.

Make two plots on one page.

34. Consider the diode circuit shown in the fig- ure. The CWTent i» and the voltage v» can be determin from the solution of the following system of equations:



$$i» = Io\left(ekT - 1\right), \quad i» = \frac{Vs\text{-}VD}{R}$$

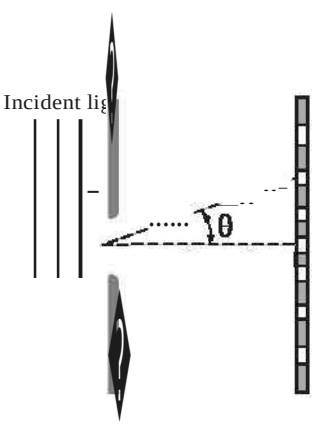
The system can be solved numerically or graphically. The graphical solution is found by plot i» as a function of v» from both equations. The solution is the intersection of the two curves. Make the plots and estimate the solution for the case where 10 = to-14 A, V_S = 1.5 V, R = 1200 O, and $\frac{leT}{q}$ = 30mV.

35. When monochromatic light passes through a narw slit it produces on a screen a diffractio pat consisting of bright and dark fringes. The intensity of the bright fringes. I, as a function of 9 can be calculated by

$$I = I_0\left(\frac{\sin a}{a}\right)^2 \text{ where } a = \frac{na}{A}\sin\theta$$

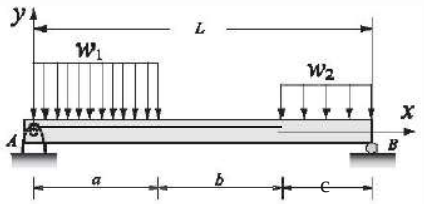
where A is the light wave length and a is the width of the slit Plot the relative intensity III,a as a fimction of 9 for -20°s9s 20°.

Make one plotthat contains thre graphs for the cases a = a = A. Label the axes, and display a legend.



I 0A, a = SA, aod

36. A simply supported beam is subjected to distributed loads w1 and w2 as shown. The bending moment as a function of x is given by the following equations:



$$M(x) = R_A x - \frac{WtXZ}{2} \quad \text{for } 0 \le x \le a$$

$$M(x) = RAx - \frac{w1a}{T}(2x-a) \quad \text{for } a \le x \le (a+b)$$

$$M(x) = R8(L-x) - \frac{w2(L-x)^2}{2} \quad \text{for } (a+b) \le x \le L$$

where $R_A = [w1a(2L-a)+w2c^2]/(2L)$ and $R_B = [w2c(2L-a)+w1a^2]1(2L)$ *are* the reactions at the supports. Make a plot of the bending moment Mas a function of x (one plot that shows the moment for $0 \le x \le L$) Take L = 16 ft, a = b = 6 ft, w1 = 400 lb/ft, and w2 = 200 lb/ft.

37. Biological oxygen demand (BOD) is a measure of the relative oxygen depletion effect of a waste contaminant and is widely used to assess the amount of pollution in a water source. The BOD in the eftluent (Lc in mg/L) of a rock filter without recirculation is given by:

$$L_c = \frac{L_o}{1+\frac{(2.5D^{7/3})}{\sqrt{Q}}}$$

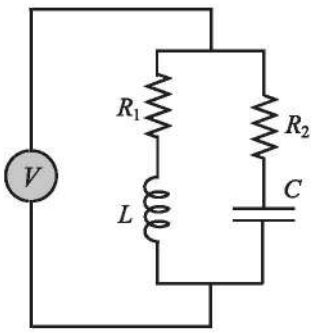
where L0 is influent BOD (mg/L),Dis the depth of the filter (m), and Q is the hydraulic flow rate (U(m^2-day)). Assuming Q = 300 Ll(m^2-day) plot the efiluent BOD as a function of the depth of the filter ($100 \le D \le 2000$ m) for L0 = 5, 10, and 20 mg/L. Make the three plots in one figure and estimate the depth of filter required for each of these cases to obtain drinkable water. Label the axes and display a legend.

38. The temperature dependence of the diffusion coefficientD(cm^2/s) is given by an Arrhenius type equation:

$$D = D0e\left(-\frac{!i}{}\right)$$

where D0 (cm^2/s) is pre-exponential constant, *Ea* (J/mol) is activation energy for diffusion, R = 8.31 (J/mol-K) is the gas constant, and T is temperature in Kelvin. For diffusion of carbon into stainless steel D0 = 6.18 cm^2/s, and *Ea* = 187 KJ/mol. Make two plots of D versus T for $200 \le T \le 800$C. In the first plot use linear scale for both axes and in the second plot use linear scale for T and logarithmic scale for D. Which plot is more useful?

39. The resonant frequency j(in Hz) for the circuit shown is given by:



$$f = \frac{1}{2\pi}\sqrt{\frac{R_1^2C - L}{LCR_2^2C - L}}$$

Given L = 0.2 H, C = 2 x 10^{-6} F, make the following plots:

(a) f versus R2 for 500 ≤ R2 ≤ 2000 Ω, given R_1 = 1500 Ω.

(b) f versus R_1 for 500 ≤ R1 ≤ 2000 Ω, given R2 = 1500 Ω.

Plot both plots on a single page (two plots in a column).

$$1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6i} + \frac{x^8}{8i} - \frac{x^{10}}{10!} + \ldots$$

Plot the figure on the right, which shows, for -27t ≤ x ≤ 27t, the graph of the function cos(x) and graphs of the Taylor series expansion of cos(x) with two, four, and six terms. Label the axes and display a legend.