Unit 3: Programming with MATLAB

Introduction

Algorithms are the main product of numerical analysis. A mathematical algorithm is a formal procedure describing an ordered sequence of operations to be performed a finite number of times. Like recipes, algorithms form the basic building blocks of addition, subtraction, multiplication, and division, as well as programming constructs like for, while, and if. Therefore, in this unit, our primary concerns for programming in MATLAB will be three fold: to give students basic knowledge of MATLAB programming, to prepare students for other courses or modules where MATLAB is used, and to give you an insight into state-of-the-art tool for technical computation and visualization.

Unit Objectives

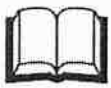
On successful completion of the Unit, students should be able to:

- Explain the basic concepts of flow control in MATLAB
- Write MATLAB codes
- Execute function files
- Plot functions
- Relate MATLAB codes to other programming environments

Key Terms

As you go through this unit, ensure that you understand the key terms or phrases used in this unit as listed below:

- Statement
- Code
- Run
- Loop
- Function
- Plot



Flow Control

What is key to writing and executing MATLAB programs is the ability to understand the flow control. By this we mean a group of statements which form a basic building block of each programming language, not only in MATLAB. Thus section therefore explains the basic concepts of flow control in MATLAB. The topics include

- If-else statement
- Switch and case statement
- For loop
- While loop

1. The If-else statement

This kind of logical statement evaluates a logical expression and executes a group of statements when the expression is true. Remember the ***TRUTH*** table of an "***If...then...***" statement from College Algebra module. The options ***elseif*** and ***else*** keywords provide for the execution of alternate groups of statements. An ***end*** keyword, which matches with the ***if***, terminates the last group of statements. Then the groups of statements are delineated by the four keywords with no braces or brackets involved. Basically we have a structure as

```
if.....condition 1 % is true
        % execute these commands
elseif.....condition 2 % is true
        % execute these commands
        ⋮
elseif.....condition n  % is true
        % execute these commands
else.....otherwise % the default
        % execute these commands
end
```

For example, if on the command window, we write a MATLAB code of "if-else" statement for executing an $n \times n$ matrix under some unspecified value of n, then we get an error "Undefined function or variable 'n'".

```
>> if n>2
M=eye(n)
elseif n<2
M=ones(n)
else
M=zeros(n)
end
Undefined function or variable 'n'.
```

However, if we write same lines of code, but by beginning with specifying the value of n, first, then we get the result upon the TRUE condition as

```
>> n=5;
>> if n>2
M=eye(n)
elseif n<2
M=ones(n)
else
M=zeros(n)
end
M =
     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
```

Here, we see that we get a 5×5 identity matrix, executed by the line of code "***M=eye(n)***" since the condition "***if*** $\boldsymbol{n > 2}$" was ***TRUE*** given that $n = 5$. All other statements' conditions are FALSE and therefore cannot be executed. To see this, suppose $n = 2$, we get

```
>> n=2;
>> if n>2
M=eye(n)
elseif n<2
M=ones(n)
else
M=zeros(n)
end
M =
   0   0
   0   0
```

2. Switch and case statement

The switch-case statement executes groups of statements based on the value of a variable or expression. The keywords ***case*** and ***otherwise*** delineate the groups. Only the first matching case is executed. There must always be an ***end*** to match the switch. The format is

```
Switch (input)
        case 1
        case 2
                ⋮
        Case  n
end
```

For example, using same code for "if-else" statements, we now write

```
>> n=2;
>> switch(n)
case 1
M=eye(n)
case 2
M=ones(n)
case 3
M=zeros(n)
end
M =
   1   1
   1   1
```

3. For loop

Key to running a sequence of steps in a MATLAB code is the "***for…loop***" statement. The for loop repeats a group of statements a fixed, predetermined number of times. A matching end delineates the statements. Its basic structure is

```
for.....x=array
        % execute these statements
end
```

For example

```
>> n=5;
>> for m=1:n
r(m)=rank(magic(m));
end
>> r
r =
   1   2   3   3   5
```

Or another example

```
>> n=6;
>> for i=1:n
i=i+1
end

i =
    2  3  4  5  6  7
```

4. Do While loop

Another kind of important executable programming statement is the "***Do...while...loop***". The while loop repeats a group of statements an indefinite number of times under control of a logical condition. A matching end delineates the statements. Therefore, the basic structure of a while loop appears as

```
while.....expression x is true
        do % execute these statements
end
```

For example,

```
>> p=5;
>> while p > 1
p=p-1;
r=zeros(p)
end
r =
   0   0   0   0
   0   0   0   0
   0   0   0   0
   0   0   0   0
r =
   0   0   0
   0   0   0
   0   0   0
r =
   0   0
   0   0

r =

   0
```

5. Other flow structures

   a) The ***break*** statement. A while loop can be terminated with the break statement, which passes control to the first statement after the corresponding end. The break statement can also be used to exit a for loop.

b) The ***continue*** statement can also be used to exit a for loop to pass immediately to the next iteration of the loop, skipping the remaining statements in the loop.
c) Other control statements include ***return***, ***continue***, ***switch***, etc.

Writing programs in MATLAB is supported by six relational and three logical operators as presented in Table 6 and Table 7, respectively.

Table 6: MATLAB Relational Operators

Expression	**Symbol**
Less than	<
Less than or equal to	<=
Greater than	>
Greater than or equal to	>=
Equal to	==
Not equal to	~ =

Table 7: MATLAB Logical Operators

Expression	**Symbol**
not	~
and	&
or	|

Activity 3 a

a) Consider the following system of linear equations

$$\begin{cases} x + 2y + 3z = 1 \\ 3x + 3y + 4z = 1 \\ 2x + 3y + 3z = 2 \end{cases}$$

   a. Use the MATLAB editor to create a script file called ***exercise1.m***

   b. Hence, use the script file to solve for $\boldsymbol{x}$ using $A\backslash b$ method.

b) Use a MATLAB script file called ***exercise2.m*** to plot the following cosine functions, $y_1 = 2\cos x$, $y_2 = \cos x$, and $y_3 = 0.5\cos x$, in the interval $0 \leq x \leq 2\pi$.

c) Consider the following quadratic equation $y = 2x^2 + 3x - 4$.

   a. Express it in the form of $ax^2 + bx + c = 0$ and deduce the values of a, b and c.

   b. Hence, write an ***if...else*** statement to classify the type of roots using discriminant.

d) Write a for...loop statements which form the 5-by-5 symmetric matrix A with (i,j) element i/j for $j \geq i$.

e) Discuss the following while...loop and understand the output.

```
function trial_3
x = 1;
while x <= 10
    x = 3*x;
    x
end
```