MATH 212

Computer Programming with MATLAB

Lecture 3: Programming in MATLAB

Chenar Abdulla

Tishk International University

Learning outcomes of this lecture

By the end of the theoretical lecture, students will:

1. Understand and apply conditional statements in MATLAB, including the 'if', 'elseif', and 'else' statements, to control the flow of program execution based on specified conditions.
2. Learn how to construct and utilize loops in MATLAB programming, including 'for' loops for iterating over a sequence of values and 'while' loops for executing code as long as a specified condition is true.
3. Develop problem-solving skills by applying conditional statements and loops to various programming tasks, enhancing the ability to write efficient and structured MATLAB code.
4. Gain hands-on experience through solving examples and practical exercises to reinforce understanding and proficiency in using conditional statements and loops in MATLAB programming.

By the end of the theoretical lecture, students will:

1. engage in hands-on exercises, solving problems and applying concepts learned during the lecture and engaging in collaborative learning.
2. integrate the new concepts of vectors and matrices with the foundational principles covered in Lecture 1.
3. complete assignments applying the concepts learned in class, showcasing their ability to independently practice and apply MATLAB programming basics.

M-files: Scripts and functions

To take advantage of MATLAB's full capabilities, we need to know how to construct long (and sometimes complex) sequences of statements. This can be done by writing the commands in a file and calling it from within MATLAB. Such files are called "m-files" because they must have the filename extension ".m". This extension is required in order for these files to be interpreted by MATLAB.

There are two types of m-files: script files and function files.

Script files contain a sequence of usual MATLAB commands, that are executed (in order) once the script is called within MATLAB. For example, if such a file has the name compute .m , then typing the command compute at the MATLAB prompt will cause the statements in that file to be executed. Script files can be very useful when entering data into a matrix.

Example: Create the *script file* then write a program to find the roots of equation $2x^2 - 3x + 1 = 0$.

Solution:

```
% this script finds the roots of equation 2x^2-3x+1=0.
      a=2; b=-3; c=1;
      x1=(-b+sqrt(b^2-4*a*c))/2
      x2=(-b-sqrt(b^2-4*a*c))/2
```

Input and output command

Display:

- **disp(X)** displays an array, without printing the array name.
- If X contains a text string, the string is displayed.
- Another way to display an array on the screen is to type its name, but this prints a leading "X=," which is not always desirable.
- Note that **disp** does not display empty arrays.

```
>> disp('string expression');
string expression

>> A=[3,2;2,3];
disp(A);
 3 2 2 3
```

Input:

The input function can be used for requesting user input. For example,

```
r=input('value for r: ');
```

displays

```
value for r:
```

to the screen and waits for the user to enter an expression which is then assigned to r.

Conditional Statement: IF condition

- Conditional statements are very basic and important for every programmer.
- There will be some situations where a program or a particular block has to be executed only when a specific condition is True.
- These conditional statements will be very handy and fruitful in such situations.

There are three types of ***if*** constructs:

Conditional Statement: IF condition

If Syntax:

```
if expression
        block of statements
end
```

Note: The block of statements is executed only if the expression is true.

Example: Write a script to tell if a number is negative.

```
a = input('a=');
if a < 0
        disp('a is negative');
end
a
```

Note: One-line format uses comma after if expression

```
if a < 0, disp('a is negative'); end
```

if. . . else syntax: Multiple choices are allowed with if. . . else

```
if expression
        block of statements
else
        block of statements
end
```

Example: Write a script to compute the square root of a given x ?

```
x = input('x=');
if x < 0
        error('x is negative');
else
        r = sqrt(x);
end
r
```

Conditional Statement: IF condition (cont.)

If … elseif Syntax: It's a good idea to include a default else to catch cases that don't match preceding if and elseif blocks.

```
if expression
       block of statements
elseif expression
       block of statements
else
       block of statements
end
```

Example: Write a script to tell if a number x is positive, negative or zero.

```
if x > 0
       disp('x is positive');
elseif x < 0
       disp('x is negative');
else
       disp('x is exactly zero');
end
```

Example:

Q1) Write a script to read a matrix a then change the location of maximum number and minimum number between them.

Q2) Write a script to read the number n then find S where $S = 2 + 2^2 + 2^3 + \cdots + 2^n$.

Q3) Write a script to solve a linear system $AX = b$.

Q4) Write a script to find $C = A * B + A^2$ where A and B are a matrices.

Q5) Write a script or function to input the number then check that even integer number or not.

Q6) Write a program to input x then find

$$y = \begin{cases} \sin(x) & 0 \le x < 1 \\ e^{|x|} + 2 & x < 0 \\ x^2 + 3 & 1 \le x < 10 \\ \dfrac{1}{x} & x \ge 10 \end{cases}$$

Loops:

- There may be a situation when you need to execute a block of code several times.
- In general, statements are executed sequentially.
- The first statement in a function is executed first, followed by the second, and so on.

A loop statement allows us to execute a statement or group of statements multiple times.

There are two types of loops in Matlab:

for: the most used one, number of repetitions is known in advance.
while: condition is known ensuring loop continuation as long as it remains true

A sequence of calculations is repeated until either

1. All elements in a vector or matrix have been processed.
2. The calculations have produced a result that meets a predetermined termination criterion.

for loops:

for loops are most often used when each element in a vector or matrix is to be processed.

```
for index = expression
        block of statements
end
```

```
for n=1:4
  disp(n)
end
```

```
1
2
3
4
```

```
a = [24,18,17,23,28]
for j=1:length(a)
   disp(a(i))
end
```

```
24
18
17
23
28
```

Example: Write a script to sum all the numbers between 2 and 15.
Solution: We want to compute 2+3+4+ … + 15 ?

```
sum=0;
for k=2:15
  sum=sum+k;
end
disp(sum)
```

Example: Write a script to sum of all even numbers between 10 and 50.
Solution: We want to compute 10+12+14+…+48+50 ?

```
sum=0;
for k=10:2:48
  sum=sum+k;
end
disp(sum)
```

Examples

Example: Write a script to find the factorial of n.

Solution: For a positive number n, factorial is $\boldsymbol{n! = n \times (n-1) \times (n-2) \times \cdots \times 2 \times 1}$.

```
n=input('n=')
f=1;
if n<0
   error('we cannot find factorial of a negative number')
else
   for k=2:n
      f=f*k;
   end
end
f
```

Example: Print all elements in the matrix $A = \begin{bmatrix} 2 & 4 & 3 \\ 3 & -1 & 0 \end{bmatrix}$.

Solution:

```
for i=1:2
   for j=1:3
      disp(A(i,j))
   end
end
```

while loops:

while loops are most often used when an iteration is repeated until some termination criterion is met.

```
While expression
     statements
end
```

The block of statements is executed as long as expression is true.

What is the value of x at the end of this code snippet?

```
x=10
while x>0
   x=x-3
end
```

```
x=2;
while x<20
   x=3x-1
end
```

Example: Write a program to find the sum:

$$\sum_{n=0}^{\infty} \frac{1}{2^n}.$$

Solution:

```
s=0;
error=0.0005;
while 1/2^n > error
        s = s + 1/2^n;
        n = n + 1;
end
s
```

Thank you!

End of Slides.

References

- A Guide to MATLAB for Beginners and Experienced Users Second Edition, Brian R. Hunt, Cambridge, 2006
- An Introduction to MATLAB, Winfried Auzinger, Vienna, 2002 - Learning MATLAB 7, The MathWorks, 2005
- Matlab: A Practical Introduction to Programming and Problem Solving, Stormy Attaway, 2009