Introduction to MATLAB Programming

Spring 2019

Introduction

Algorithm

An **algorithm** is a sequence of steps needed to solve a problem.

- We will use MATLAB to develop algorithms to solve specific problems.
- The basic algorithm consists of 3 basic steps
  1. Get input(s)
  2. Calculate the result(s)
  3. Display result(s)

Scripts

- A script is a sequence of MATLAB instructions that are stored in a M-file and saved.
- Before creating a script, make sure the current folder is set to the folder in which you want so save your files
- To start a new script `>>edit script1.m`

`input` function

Objective: Take input from the user

- To call the input function - pass the prompt for input: If the expected input is a number

```
>>radius =input('Enter the radius:');
```

`input` function

Objective: Take input from the user

- To call the input function - pass the prompt for input: If the expected input is a number

```
>>radius =input('Enter the radius:');
```

- If the expected input is a character or string of characters

```
>> letter=input('Enter a char:','s')
```

Output statements: `disp`

- Output statements display strings and/ or results of calculations.

Output statements: `disp`

- Output statements display strings and/ or results of calculations.
- The simplest output function is `disp`

```
>> disp('Hello World')
Hello World
>> disp(4^2)
    16
```

Output statements: `disp`

- Output statements display strings and/ or results of calculations.
- The simplest output function is `disp`

```
>> disp('Hello World')
Hello World
>> disp(4^2)
    16
```

- `disp` will display the result of an expression or a string without assigning any value to `ans`.

Output statements: `disp`

- Output statements display strings and/ or results of calculations.
- The simplest output function is `disp`

```
>> disp('Hello World')
Hello World
>> disp(4^2)
    16
```

- `disp` will display the result of an expression or a string without assigning any value to `ans`.
- `disp` does not allow formatting.

Formatted output: `fprintf`

- Formatted output can be printed to the screen using `fprintf`.

```
>> fprintf('The answer is %d. \n',42)
The answer is 42.
```

- Specify decimal places for real numbers

```
>> x=2;
>> fprintf('The square root of %d is %.6f.\n',x,sqrt(x))
The square root of 2 is 1.414214.
```

Formatted output: `fprintf`

- Formatted output can be printed to the screen using `fprintf`.

```
>> fprintf('The answer is %d. \n',42)
The answer is 42.
```

- Specify decimal places for real numbers

```
>> x=2;
>> fprintf('The square root of %d is %.6f.\n',x,sqrt(x))
The square root of 2 is 1.414214.
```

- We can also specify field width

```
>> fprintf('The square root of %d is ...
    %20.6f.\n',x,sqrt(x))
The square root of 2 is             1.414214.
```

Formatted output: `fprintf`

- We can also specify field width

```
1 >> fprintf('The square root of %d is ...
      %20.6f.\n',x,sqrt(x))
2 The square root of 2 is              1.414214.
```

- If the field with is negative, the printing is left aligned

```
1 >> fprintf('The square root of %d is ...
      %-20.6f.\n',x,sqrt(x))
2 The square root of 2 is 1.414214             .
```

Formatted output: `fprintf`

- We can also print vectors or matrices

Formatted output: `fprintf`

- We can also print vectors or matrices

```
>> x = [0, 0.5, 1];
>> y = [x; exp(x)];
>> fprintf('%6.1e %12.4e\n',y);
 0.0e+00    1.0000e+00
 5.0e-01    1.6487e+00
 1.0e+00    2.7183e+00
```

- And strings

```
>> fprintf('My string is %s! \n','Hello World')
My string is Hello World!
```

Formatted output: `fprintf`

- We pass to `fprintf` text to be printed and *conversion specifications* and expressions to be printed.
- Each conversion specification is introduced by a % character and ended by a letter

	The argument
`d`	is converted into decimal notation
`c`	is taken to be a single character
`s`	is a string
`e`	is converted into decimal notation of the form `m.nnnnnExx` where the length of n's is specified
`f`	is converted into decimal notation of the form `mmm.nnnnn` where the length of n's is specified

Special formats

Special character	Format specifier
Backspace	`\b`
New line	`\n`
Horizontal tab	`\t`

Additional options can be found here

User defined functions

Scripts vs Functions

- All variables and parameters of a script are accessible in the workspace, i.e. externally accessible.

User defined functions

Scripts vs Functions

- All variables and parameters of a script are accessible in the workspace, i.e. externally accessible.
- This makes scripts good for testing and experimenting.

User defined functions

Scripts vs Functions

- All variables and parameters of a script are accessible in the workspace, i.e. externally accessible.
- This makes scripts good for testing and experimenting.
- In general, create a function to solve a given problem for arbitrary parameters.

User defined functions

Scripts vs Functions

- All variables and parameters of a script are accessible in the workspace, i.e. externally accessible.
- This makes scripts good for testing and experimenting.
- In general, create a function to solve a given problem for arbitrary parameters.
- Use a script to run functions for specific parameters required.

Anatomy of MATLAB functions

A function returning a single result consists of the following:

- Function header (the first line), comprised of

```
function outputargument = functionname(input arguments)
```

Anatomy of MATLAB functions

A function returning a single result consists of the following:

- Function header (the first line), comprised of

```
function outputargument = functionname(input arguments)
```

- Comments that describe what the function does (these comments will be printed when `help` is called)

Anatomy of MATLAB functions

A function returning a single result consists of the following:

- Function header (the first line), comprised of

```
function outputargument = functionname(input arguments)
```

- Comments that describe what the function does (these comments will be printed when `help` is called)
- The body of the function that should manipulate the `inputvariable` and assign a value to the `outputvariable`

Anatomy of MATLAB functions

A function returning a single result consists of the following:

- Function header (the first line), comprised of

```
function outputargument = functionname(input arguments)
```

- Comments that describe what the function does (these comments will be printed when `help` is called)
- The body of the function that should manipulate the `inputvariable` and assign a value to the `outputvariable`
- `end` at the end of the function

Anatomy of MATLAB functions

```
1 function outputargument = functionname(input arguments)
2 %Comments that describe what this function does
3
4 Statements and computations
5 end % end of function
```

Programming Style Guidelines

- Make sure your comments describing functions or scripts contain useful information

Programming Style Guidelines

- Make sure your comments describing functions or scripts contain useful information
- Put a newline character at the end of every string printed by `fprintf`

Programming Style Guidelines

- Make sure your comments describing functions or scripts contain useful information
- Put a newline character at the end of every string printed by `fprintf`
- Suppress the output from all assignment statements in a function
- Functions that return a value do not normally print the value

Single input and output

- Compute the area of a circle of radius r.

```
1 function area = calcarea(r)
2 %Computes area of a circle of radius r
3 %02/03 MA302
4
5 area = pi*pow2(r,2);
6 end
```

Single input and multiple outputs

- Comupute the average $\overline{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ and standard deviation $\sqrt{\frac{\sum_{i=1}^{n}(x-\overline{x})^2}{n}}$.

```
1  function [avg,sd] = stat(x)
2  %computes the average value and standard
3  %deviation of a vector
4  %PC 02/03 MA302
5
6  n = length(x);
7  avg = sum(x)/n;
8  sd = sqrt(sum((x - avg).^2)/n);
9  end
```

WARNING - the functions `mean` and `std` already exist so do not use these as variable names otherwise MATLAB will not perform these functions.

Multiple inputs and outputs

- Evaluate $f(x, y) = cx^2y$

```
1  function z = myfun(x,y,c)
2  %evaluates f(x) = cx^2*y
3  %x and y are coordinates from meshgrid
4  %PC 02/03 MA302
5
6  z = c*(x.^2).*y;
7  end
```