INTRODUCTION TO MATLAB PROGRAMMING
Lec 1.1: MATLAB Basics

Dr. Niket Kaisare
Department of Chemical Engineering
IIT–Madras

NPTEL Course: MATLAB Programming for Numerical Computations — Week-1

About this Module

• We will cover the following topics
  • MATLAB basics
  • Arrays: Unlocking potential of MATLAB
  • Loops and Execution Control
  • MATLAB files: Scripts and Functions
  • Program Output and Plotting
Starting and Exiting MATLAB

• We will go over starting a MATLAB session, layout of MATLAB window, MATLAB editor, etc.
• Also see video “Getting Started with MATLAB” on MATLAB site http://in.mathworks.com/videos/getting-started-with-matlab-68985.html

MATLAB Programming Example

Indian captain, Mahendra Singh Dhoni, hits a ball with initial velocity of 35 m/s and angle of 45°. If the boundary is at a distance of 75 m, will he score a six?

• Setting up the problem:
  • \( v_{\text{net}} = 35; u_0 = v_{\text{net}} \cos(\pi/4); v_0 = v_{\text{net}} \sin(\pi/4) \)
  • \( x' = u; y' = v \)
  • \( u' = -\kappa u; v' = -g \)
%% Define Parameters and Initial Conditions
param.g = 9.81; % gravitational acceleration
param.kappa = 0.006; % air drag coefficient
u0 = 35*cos(pi/4);
v0 = 35*sin(pi/4);

%% Setting up and Solving the problem
X0 = [0; 0; u0; v0]; % starting position is the origin
% starting velocity is given
tSpan = [0 20]; % simulation time
[tOut, XOut] = ode45(@ballTrajectoryFun,tSpan,X0, [], param);

%% Displaying the results
figure(1);
plot(XOut(:,1),XOut(:,2), 'bo');
xlabel('x (m)'); ylabel('y (m)');

%% Animating results
exitCode = ballAnimation(tOut,XOut);
### MATLAB Code: Main Code Blocks

%% Define Parameters and Initial Conditions
```
param.g = 9.81;  % gravitational acceleration
param.kappa = 0.006;  % air drag coefficient
u0 = 35*cos(pi/4);
v0 = 35*sin(pi/4);
```

%% Setting up and Solving the problem
```
X0 = [0; 0; u0; v0];  % starting position is the origin
% starting velocity is given
tSpan = [0 20];  % simulation time
[tOut, XOut] = ode45(@ballTrajectoryFun, tSpan, X0, [], param);
```

%% Displaying the results
```
figure(1);
plot(XOut(:,1), XOut(:,2), 'bo');
xlabel('x (m)'); ylabel('y (m)');
```

%% Animating results
```
exitCode = ballAnimation(tOut, XOut);
```

### MATLAB Code: Key Parts

#### Comment
```
%% Define Parameters and Initial Conditions
```

#### Assignment
```
param.g = 9.81;
```

#### (Math) Expression
```
u0 = 35*cos(pi/4);
```

#### Calling a function
```
[tOut, XOut] = ode45(@ballTrajectoryFun, tSpan, X0, [], param);
```

#### Calling a function
```
plot(XOut)
```
MATLAB Code

%% Define Parameters and Initial Conditions
param.g = 9.81; % gravitational acceleration
param.kappa = 0.006; % air drag coefficient
u0 = 35*cos(pi/4);
v0 = 35*sin(pi/4);

%% Setting up and Solving the problem
X0 = [0; 0; u0; v0]; % starting position is the origin
% starting velocity is given
tSpan = [0 20]; % simulation time
[tOut, XOut] = ode45(@ballTrajectoryFun,tSpan,X0, [], param);

%% Displaying the results
figure(1);
plot(XOut(:,1), XOut(:,2), 'bo');
xlabel('x (m)'); ylabel('y (m)');

%% Animating results
exitCode = ballAnimation(tOut,XOut);

---

Basic Data Types

- Matlab easily works with arrays
  - Scalars, vectors and arrays
  - Assigning variables
  - Row vs. column vectors
  - Arrays / Matrices
- Suppress “echo”
- Variables are case-sensitive
Basic Mathematical Expressions

Scalar Operations
• +  -  *  /  ^
• log, exp
• pow, sqrt
• sin, cos, tan
• asin, acos, atan
• rem, round, ceil, floor

Special Variables

<table>
<thead>
<tr>
<th>Variable</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>pi</td>
<td>Number π</td>
</tr>
<tr>
<td>eps</td>
<td>Machine precision</td>
</tr>
<tr>
<td>i</td>
<td>Imaginary unit</td>
</tr>
<tr>
<td>inf</td>
<td>Infinity</td>
</tr>
<tr>
<td>NaN</td>
<td>Not a Number (e.g., 0/0)</td>
</tr>
<tr>
<td>ans</td>
<td>Last displayed result</td>
</tr>
<tr>
<td>end</td>
<td>Last element of array</td>
</tr>
<tr>
<td>realmax</td>
<td>Largest real number</td>
</tr>
<tr>
<td>intmax</td>
<td>Largest integer</td>
</tr>
</tbody>
</table>

End of Lecture 1-1
Arrays are the most powerful aspect of MATLAB

- We will learn
  - Building arrays
  - Colon notations
  - Array operations and functions
- Also view “Working with Arrays in MATLAB” on MATLAB website:
Building Arrays

• Recall that we can build arrays as:
  >> A = [1, 2; 3 4];

• We can also build arrays from existing arrays (if correct size):
  >> B = [b, c];

Array Building Functions

<table>
<thead>
<tr>
<th>Command</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>ones(m,n)</td>
<td>Build m×n matrix of 1’s</td>
</tr>
<tr>
<td>zeros(m,n)</td>
<td>Build m×n matrix of 0’s</td>
</tr>
<tr>
<td>eye(n)</td>
<td>Identity matrix</td>
</tr>
<tr>
<td>diag(vec)</td>
<td>Create diagonal matrix</td>
</tr>
<tr>
<td>diag(A)</td>
<td>Diagonal elements of A</td>
</tr>
<tr>
<td>rand(m,n)</td>
<td>Uniform random number array</td>
</tr>
<tr>
<td>randn(m,n)</td>
<td>Gaussian Random number array</td>
</tr>
<tr>
<td>magic(m)</td>
<td>Magic square matrix</td>
</tr>
<tr>
<td>hilb</td>
<td>Hilbert matrix</td>
</tr>
</tbody>
</table>

Basic Mathematical Expressions

“Scalar” Operations

• log, exp
• power, sqrt
• sin, cos, tan
• asin, acos, atan
• rem, round, ceil, floor

Matrix Operations

• +, −, *, /, ^
• logm, expm
• mpower, sqrtm
• sum, prod, cumsum, cumprod
• min, max, mean, std
• length, size, eig
### Basic Mathematical Expressions

#### “Scalar” Operations
- `+`, `-`, `.*`, `./`, `.^`
- `log`, `exp`
- `power`, `sqrt`
- `sin`, `cos`, `tan`
- `asin`, `acos`, `atan`
- `rem`, `round`, `ceil`, `floor`

#### Matrix Operations
- `+`, `-`, `*`, `/`, `^`
- `logm`, `expm`
- `mpower`, `sqrtm`
- `sum`, `prod`, `cumsum`, `cumprod`
- `min`, `max`, `mean`, `std`
- `length`, `size`, `eig`

---

End of Lecture 1-2
Tapping some Array Operations in MATLAB

- Also view “Working with Arrays in MATLAB” on MATLAB website:

- Consider the following example (Marks earned by students)

<table>
<thead>
<tr>
<th>Name</th>
<th>Math</th>
<th>Programming</th>
<th>Thermodynamics</th>
<th>Mechanics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amit</td>
<td>24</td>
<td>44</td>
<td>36</td>
<td>36</td>
</tr>
<tr>
<td>Bhavna</td>
<td>52</td>
<td>57</td>
<td>68</td>
<td>76</td>
</tr>
<tr>
<td>Chetan</td>
<td>66</td>
<td>53</td>
<td>69</td>
<td>73</td>
</tr>
<tr>
<td>Deepak</td>
<td>85</td>
<td>40</td>
<td>86</td>
<td>72</td>
</tr>
<tr>
<td>Elizabeth</td>
<td>15</td>
<td>47</td>
<td>25</td>
<td>28</td>
</tr>
<tr>
<td>Farah</td>
<td>79</td>
<td>72</td>
<td>82</td>
<td>91</td>
</tr>
</tbody>
</table>
Some things to try

• Create a 6×3 matrix allMarks to contain marks for first three courses
• Append marks for the Mechanics course to allMarks when received
• Do the following computations
  • Mechanics course was out of 50. Scale the marks to half
  • Extract row 3 and give the marks. Then extract single marks
  • Extract marks of our best students, Deepak and Farah for first three courses
  • Calculate average marks obtained in each of the four courses
  • Scale all the marks out of 10

We will use matrix fundaes for this:

\[
\begin{bmatrix}
a & b \\
c & d
\end{bmatrix}
\begin{bmatrix}
2 & 0 \\
0 & 0.1
\end{bmatrix}
= 
\begin{bmatrix}
2a & 0.1b \\
2c & 0.1d \\
2e & 0.1f
\end{bmatrix}
\]

End of Lecture 1-2b
INTRODUCTION TO MATLAB PROGRAMMING
Lec 1.3: Loops and Execution Control

Dr. Niket Kaisare
Department of Chemical Engineering
IIT–Madras

NPTEL Course: MATLAB Programming for Numerical Computations — Week-1

Various Loops in MATLAB

• For Loop (commands below will execute 10 times)

```matlab
for i=1:10
    <statement 1>
    ;
    <statement n>
end
```

• While Loop (commands below will execute if the condition is true)

```matlab
while i<10
    <statement 1>
    ;
    <statement n>
    i=i+1;
end
```
When to use For Loop

• For loop is used when a set of operations are to be repeated a specific number of times

• Examples
  • Find first 10 terms of Fibonacci series
  • Find factorial of a number $n$
  • ...

When to use While Loop

• While loop is used when a set of operations is to be repeated if a certain condition is met

• Find all terms of Fibonacci series less than value 200

• Location of a ball thrown upwards is given by $y = v_0 t - \frac{1}{2} g t^2$. Calculate the location of the ball for every 0.1 seconds until it reaches the ground (i.e., $y > 0$)
MacLaurin Series

• Calculate approximate value of $e^{0.5}$ using the infinite series:

$$e^a = 1 + a + \frac{a^2}{2!} + \frac{a^3}{3!} + \frac{a^4}{4!} + \cdots$$

These calculations are to be performed with 2 to 7 terms in the series

End of Lecture 1-3
INTRODUCTION TO MATLAB PROGRAMMING
Lec 1.4: Working with Files – Scripts & Functions

Dr. Niket Kaisare
Department of Chemical Engineering
IIT–Madras

NPTEL Course: MATLAB Programming for Numerical Computations — Week-1

Working with MATLAB files

- Type “edit <fileName>” at the command prompt to open MATLAB code editor with the file fileName.m.
- MATLAB files are of two types: Scripts and Functions
- More help from MATLAB website on “Writing a MATLAB Program”:
MATLAB Files: Scripts vs. Functions

- **Scripts**
  Files containing sequence of MATLAB commands
  - MATLAB statements are executed as if typed on command prompt

- **Functions**
  Files that take certain input(s), executes sequence of steps, and returns output(s) at the end
  - MATLAB statements are executed in function’s own variable space

Scope of Variables

- **script** shares the variables with workspace from where it was called
  - Typically, that means MATLAB workspace

- **function** has its own workspace
  - Variables used in a function have local scope
  - Functions “talk” through input and output variables:
    \[
    [\text{out1}, \text{out2}, \ldots] = \text{function fcnName(in1, in2, \ldots)}
    \]
Script and Function Examples:

- Write a script to calculate factorial
  \[ n! = 1 \times 2 \times \cdots \times n \]

- Write a function to calculate
  \[ f = c_0 + c_1 x + c_2 x^2 + \cdots + c_n x^n \]

Note: Such functions are commonly used to calculate physical properties of fluids.
Today, we will consider a simple case of:
\[ c_0 = 1, \quad c_m = 1/m \]

When to use Scripts vs. Functions (beginners)

- Use scripts when you want to...
  - Make small calculations (e.g., factorial, plotting, basic computing etc.)

- Use functions when you want to...
  - Calculate values \( r \) as a function of variables \( t, y, \ldots \): \( r = f(t, y, \ldots) \)
  - Pass on the function values to MATLAB function for solving something; e.g.,:
    \[ \frac{dy}{dt} = f(t, y) \rightarrow \text{function } dy = \text{myODEfun}(t, y) \]
    \[ <\ldots>\text{ode45(@myOdefun, <\ldots>)} \]
  - Calculate properties as a function of temperature, concentration, current, etc.

- All other purposes, you are likely to use scripts (instead of functions)
INTRODUCTION TO MATLAB PROGRAMMING
Lec 1.5: Plotting and Output

Dr. Niket Kaisare
Department of Chemical Engineering
IIT–Madras

NPTEL Course: MATLAB Programming for Numerical Computations — Week-1

Various forms of output

• Display on the screen
  • Variables will echo if command ends without semicolon
  • Other options...

• Plotting data
  • Using plot command
  • Other options...

• More help from MATLAB website on “Using Basic Plotting Functions”
  http://in.mathworks.com/videos/using-basic-plotting-functions-69018.html
Displaying on the screen

• Recall various methods we used in this module:
  • Echo result on screen:  \( \gg b = [1, 2; 7 1] \);
  • Using \texttt{disp} command: \texttt{disp(b)}
  • \texttt{disp some text}: \texttt{disp('Hello world')}
  • More “beautiful” output:
    \[ \text{disp(['Factorial value is ', num2str(factValue)])} \]
  • More advanced output using \texttt{fprintf}:
    \[ \texttt{fprintf('Factorial Value is: %4i\n',factValue)} \]

Plotting

• Consider the example of a ball thrown vertically upwards
  • Plot location vs. time
  • Labeling the axes
  • Other plotting options
  • Plot-\textit{ting} multiple lines
  • Log-Log \textit{plot}
MODULE – 1
INTRODUCTION TO MATLAB PROGRAMMING

Dr. Niket Kaisare
Department of Chemical Engineering
IIT–Madras

NPTEL Course: MATLAB Programming for Numerical Computations — Week-1
Summary of Module-1

- MATLAB basics
  - Familiarized with MATLAB command window and editor
  - Variables: scalars, vectors and arrays
  - Mathematical operations: both scalar and matrix operations
- Arrays: Unlocking potential of MATLAB
  - Array operations vs. elemental operations
  - Using arrays for more efficient use of MATLAB

Summary of Module-1

- Execution control
  - for and while loops
  - if-then statements
- MATLAB files
  - Scripts and Functions
  - When to use scripts vs. functions
- Plotting in MATLAB