W1005
Intro to CS and Programming in MATLAB

MATLAB Basics

Fall 2014
Instructor: Ilia Vovsha

http://www.cs.columbia.edu/~vovsha/w1005

Outline

- Workspace
- Variables & Data Types
- Arrays (vectors & matrices)
- Indexing and access
- Operators
- Scripts
- Internal type representation
- Numeric data types
- Character data type

Basic Functionality – Workspace

- You type commands in the *command window*, MATLAB tracks your variables in the *workspace* and your commands in the *command history*
- Can manipulate directories from the command line:
  - cd, mkdir, rmdir, ls
- Help window
  - ‘help’ command is about to become your new best friend

Basic Functionality – Help

- Can search using the window
- Typically easier to use:
  - doc < ***function_name***>
  - help < ***function_name***>
- Documentation for a particular function often lists related functions as well

Variables & Data Types

- A *variable* is a name associated with a memory cell whose contents can change during program execution
- A *data type* is a set of values and operations that can be performed on those values
  - int (integer), float, boolean, char, string
- What is the purpose?
  - Determine which operations are valid
  - Represent value in memory

Variables

- Valid variable name starts with any letter, followed by letters / digits / underscores
- MATLAB does not use explicit type initialization:
  - int a (WRONG!)
  - a = 2 (Good)
  - a = ‘hello’ (also good)
- Add operator ‘;’ to avoid printing the variable:
  - a = 2;
- Use ‘disp’ to improve display of variable:
  - disp(a)

Variables

- Can assign and reassign a variable directly:
  - a = (1+1) /4;
  - a = ‘hello’
- Case sensitive:
  - a = 2 & A = 4 are two different variables
- Avoid using *reserved words* (keywords) or functions as variable names:
  - sort = 2 (DANGEROUS!)
  - Double check with exist(‘< variable >’) to be safe

Variables

- Some reserved words (keywords):
  - for, if, else, while, end, return…
  - Use ‘iskeyword’ to check
- Some built-in variables:

Inf & -Inf	+/- infinity
pi	3.141…
NaN	Not a number (0.0/0.0)

Basic Functionality – Commands

- Clean up:
  - clear < ***variable*** >
  - clear all
  - clc (clear command window)
- Status check:
  - what (returns the MATLAB files (.m , .mat) in the current directory)
  - who (returns the variables in your workspace)
  - whos (variables with additional info, e.g. size)

Data Structures – Arrays

- The secret behind MATLAB’s popularity (?)
- Started out as a ‘matrix laboratory’
- It turns out we do a lot of computation with vectors and matrices
- MATLAB makes it easy to manipulate these data structures

Data Structures – Vectors

- Define a row vector:
  - r = [2 3 5 7]
  - r = [2,3,5,7];

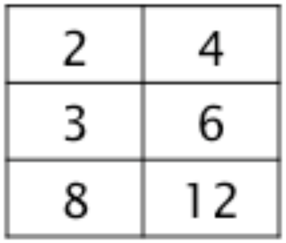
2	3	5	7

- Define a column vector:
  - c = [2; 3; 5; 7];
  - c = [2,3,5,7]’;
  - We can use a transpose operator (`)

2
3
5
7

Data Structures – Matrices

- Define a matrix:
  - M = [2,4; 3,6; 8,12];



2	4
3	6
8	12

- Concatenate two vectors:
  - v1 = [2,4]; v2 = [3,6];
  - M = [v1; v2];
- Can concatenate vectors and matrices. Dimensions and types must agree

Data Structures – Matrices

- Special constructor “:”
  - r = 1:5; ➔ [1,2,3,4,5]
  - r = 1:2:5; ➔ [1,3,5]
  - M = [1:5; 1:5]
  - M = [1:5; 1:2:5] (ERROR!)

Data Structures – Matrices

- Some predefined matrix creation functions:
  - M = zeros(2,3);   Matrix of zeros
  - M = ones(2,3);   Matrix of ones
  - M = eye(2);   Identity matrix (2 by 2)
  - M = rand(2,3);   Random numbers (uniformly distr.)
  - 1st argument = # of rows, 2nd argument = # of columns.
- We will consider 2D matrices almost exclusively, but may just as well use 3D cubes e.g. M = zeros(5,5,5)

Data Structures – Matrices

- Matrix dimensionality:
  - M = rand(2,3);
  - [r,c] = size(M);
  - r = size(M,1);   # rows
  - c = size(M,2);   # columns
  - 1st argument = # of rows, 2nd argument = # of columns.

Data Structures – Matrices

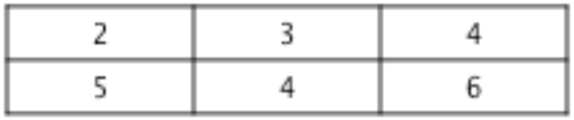
- Indexing & Accessing array elements:
  - Index starts from 1
  - e1 = M(1,2); Explicit access
  - e1 = v(4); Explicit access
  - v1 = M(1,1:2); Return 1st two columns from row 1
  - v1 = M(:,2); Return 2nd column
  - e1 = M(1,end) Return element from last column in row 1

Basic Operators

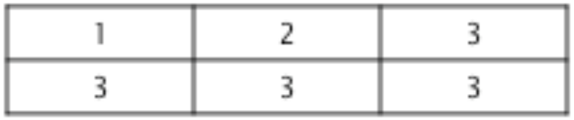
- The usual math operators:
  - + - * / ^
- Logical operators:
  - & | ~
- Relational operators:
  - > < >= <= == ~=
- Matrix operators:
  - Ambiguous, element wise or matrix operation?
  - .* ./ .^ use dot to disambiguate

Matrix Operators

- X = [2 3 4; 5 4 6]; Y = [1 2 3; 3 3 3];
- Operations:
  - Z = X + Y ~~(X .+ Y)~~
  - Z = X – Y ~~(X .- Y)~~
  - Z = X .* Y ~~(X * Y)~~
  - Z = (X’) * Y
  - Z = X .^ 2 ~~(X ^ 2)~~



2	3	4
5	4	6



1	2	3
3	3	3

Scripts

- MATLAB specific script files (M-files)
- Files have the extension .m (example: test.m)
- To run your script just type ‘test’ in the command line
- A script is a list of commands that should be executed
- To pass arguments to the script we will use functions (more on that later)

Comments

- Comments make the code more manageable
- Can add a one-line comment by using the '%' symbol:
  - **% this is a comment**
- Anything that follows a % is ignored by the compiler (not executed)
- When you type help <func_name> you get the comments at the top of the <func_name> script
- Can also use block comments: %{ …. %}

Debugging

- Debugging is the proper way of tracking program execution and fixing errors
- Type 'help debug' or use 'debug' from the workspace menu
- Simple alternative: pause() function
  - pause(): halt program until user strikes any key
  - pause(n): halt program for n seconds
- Approach: add output statements followed by pause() to follow progress

Numeric Data Types

- Two numeric types: *integer* & *float*
  - Integer type: faster & more precise operations, less storage space
  - Float type: more accuracy in computation
- How is data represented internally in memory?
  - Strings of binary digits (bits): 1101
  - Floating-point numbers converted to scientific notation: 3.57×10^3
  - Float-number = mantissa x $2^{characteristic}$
  - Range for float type numbers is much larger than integer type
  - Range also depends on computer/compiler
  - 'short' or 'long' types reflect size (storage bits) of types

Numeric Types (conversion)

- What happens when we mix types in expressions?
  - If mixing is allowed, compiler *promotes (converts)* operands to make them the same (this is automatic)
  - The result of the operation is the same type as the operands after promotion
  - Such conversions are intended to be *value-preserving*
  - The opposite of promotion is *truncation* (chopping off fractional part)
  - Explicit type conversion: *casting*

Integer Types (commands)

- Integer data types in MATLAB:
  - 8,16,32,64-bits
  - Unsigned or signed
  - Range: 0 to 2^bits OR -2^(bits-1) to 2^(bits-1)
- Commands:
  - int8(-12.5); uint8(-12.5) % Casting
  - v = zeros(3,3,'int8');
  - class(v); % Check type
  - intmax(<class>); intmin(<class>); % Range limit
  - Mathematical operations not defined for different integer types

Float Types (commands)

- Default data type is ‘double’ (floating-point type):
  - ‘single’-precision data as storage-efficient alternative
  - Similar commands to integer types <class> is ‘double’ or ‘single’
  - ‘inf’ and ‘NaN’ are of class double
- Commands:
  - v = eye(4,‘single’);
  - class(v); % Check type
  - realmax(<class>); realmin(<class>); % Range limit
  - isnumeric(v); % Check type

Character Type

- Character type (char):
  - Single printable character or escape sequence (‘\n’, ‘\t’)
  - Internally: char associated with unique code, code stored in memory cell in binary form
  - Common character set: *ASCII* (American Standard Code for Information Interchange)
  - Consecutive codes to represent digit characters ‘0’, ‘9’ and letters (lower, upper case)
  - Printable chars have codes from 32 to 126
  - Nonprintable control characters: 0-31 & 127
  - When comparing chars we rely on the ASCII order: ‘2’ > ‘12’