Introduction to MATLAB

Neil Rayala

November 16, 2018

1 Introduction

MATLAB, short for Matrix Laboratory, is a programming interface which is used to analyze data, compute quantitative data, simulate data models, and perform many other features.

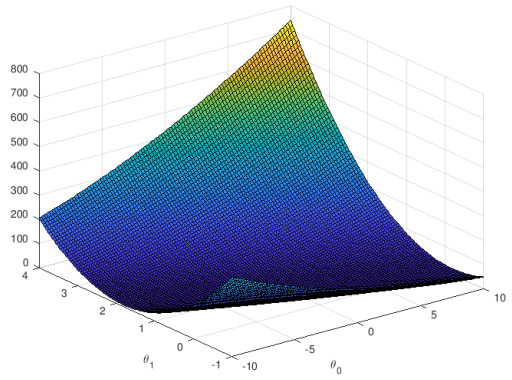

Figure 1: Visualization of Model Prediction in MATLAB

2 Environment

Visit https://bit.ly/2qg7zX6 to install MATLAB. At the bottom of the interface, you will see the command prompt. The two greater than symbols signify that MATLAB is ready to receive instructions from the user. Type your instruction and press Enter for MATLAB to execute the command. Make sure to use the *help* and *lookfor* commands to find information about known commands.

```
1 %Example of command prompt input/output
2 >> lookfor roots
3 poly                          - Convert roots to polynomial.
4 roots                         - Find polynomial roots.
```

3 Fundamentals

Data is stored in variables (i.e. a=1). Variables must begin with a letter and have a maximum length of 32 characters. Each variable can also store calculations (i.e. b=1+1). Semicolons are used at the end of a variable assignment to avoid the interface from printing the value to the screen.

Character strings are typically used to describe the program's output and is written with single quotation marks on each end.

Algorithm 1 Character String Methods

```
name = 'Intro MATLAB';
disp(name);                         %Displays Name
num = 234;
disp(int2str(num));                 %Integer to String
num = 3.14159265;
disp(num2str(num,3));               %Float to String of 3 Digits
disp(num2str(num,5));               %Float to String of 5 Digits
```

Algorithm 2 Common Math Operations

```
%Coordinate Manipulation
[x, y] = pol2cart(theta, r)         %2D Polar to Cartesian
[theta, r] = cart2pol(x, y)         %Cartesian to 2D Polar
[x, y, z] = sph2cart(a, theta, r)   %3D Spherical to Cartesian
[a, theta, r] = cart2sph(x, y, z)   %Cartesian to 3D Spherical

%Trigonometric Functions
v = [sin(x), cos(x), tan(x), csc(x), sec(x), cot(x)] %Standard
v = [asin(x), acos(x), atan(x), acsc(x), asec(x), acot(x)]
                                                     %Inverse
%Sign Manipulation
sign(x)                     %Numerical Sign of x
abs(x)                      %Absolute Value

%Exponents and Logs
exp(x)                      %e^x
sqrt(4.5)                   %Square Root of 4.5
log(x)                      %natural log of x
log2(x)                     %base 2 log of x
```

Visit https://www.mathworks.com/help/matlab/operators-and-elementary-operations.html for more information on math operators.

3.1 Vectors

Vectors are matrices with one column or one row and are notated with brackets, such as [1 5 10] or [1; 5; 10]. The semicolons in the second example are used as row separators and form a 3 x 1 column vector. Vectors support addition and subtraction with + and - and scalar multiplication with *. Element-wise operation is performed by using the . operator. For example, b = a.*2 would multiply each element of vector a by 2 and store all elements in vector b. Vectors are named using lowercase letters. Use v(i) to access an element at index i in vector v.

3.2 Matrices

Matrices are also notated with brackets, such as [1 5 10; 15 20 25], in which the semicolons are row separators and spaces are column separators. These structures are named using uppercase letters.

Algorithm 3 Basic Matrix/Vector Operations

```
M(1, 2)          %Access element in 1st row and 2nd column
M([1 2], [3 4]) %Submatrix with 1st/2nd row and 3rd/4th column
M(3, 1:end)      %Contains 3rd row and all columns
M(3, :)          %Same as above (: is same as 1:end)

[2:3:6] %Arithmetic sequence in vector (2—6 with increment 3)
M.*2    %Multiplies all elements by a scalar factor of 2
M'      %Apostrophe sign is transpose operator for matrix/vector
size(M) %Returns dimensions of matrix/vector

dot(v, w)      %Dot Product of 2 Vectors
cross(v, w)    %Cross Product of 2 Vectors
M = [r1; r2]   %Matrix built from two row vectors
M = [c1 c2]    %Matrix built from two columns
M = [A; B; C] %Matrix built from other matrices

r1 = M(1,:)    %Returns row vector from matrix
c1 = M(:,1)    %Returns column vector from matrix
v = M(2:3, 2) %Returns vector containing 2nd/3rd row and 2nd col
M(2:3, 3:4)    %Matrix with 2nd/3rd row and 3rd/4th column

zeros(r, c)    %Matrix of 0's with r rows and c columns
ones(r, c)     %Matrix of 1's with r rows and c columns
eye(n)         %n x n Identity Matrix
magic(n)       %n—order Magic Square as Matrix
```

Note: MATLAB implements one-based indexing (starts from 1 instead of 0)

4 Control Structures

If-statements make decisions based on variables stored in memory and require logical/relational operators. Common operators include ==, ~=, <, <=, >, >= , &, *or*, *all*, *any*, *xor*.

Algorithm 4 If Statement Example

```
v = [1 2 3 4];
if ne(v(2), 2)  %alternative for not equal operator
    disp("5");
elseif v(2) > 2 %otherwise checks if second element is > than 2
    disp("4");
else            %displays 1 since they are equal
    disp("1");
end
```

For loops run a block of code for a defined duration.

Algorithm 5 For Loop Examples

```
for i = 1:2:10  %1 to 10 with increment 2
    i + 5       %displays number + 5
    i ^ 2       %displays number ^ 2
end

for v = 'INTRO MATLAB'  %iterates through vector
    if v == 'A'         %if the for loop reaches element 'A'
        disp(v)
        break           %terminates loop
    end
end
```

While loops run a block of code until the logical operator returns 0 (False).

Algorithm 6 While Loop Example

```
i = 1;
while i < 10    %displays 1 to 9
    disp(i)
    i = i + 1;
end
```

Note: Use the end keyword to stop each control structure.

5 Exercises

Given a matrix M and using control structures:

1. Store each column of M into vectors and then compute the cross products of all possible combinations of column vectors. Store these products into matrix A. Assume A's dimensions are already defined.

2. Modify M so that each element is squared and multiplied by a factor of 2. Then compute the sum of M.

3. Do Problem 2 without using control structures.

Example Matrix:

$$\begin{pmatrix} 1.5 & 3.15 \\ 5.0 & 2.66 \\ 10.4 & 9.1 \\ 6.3 & 7.9 \\ 5.2 & 8.7 \end{pmatrix}$$