Introduction to MATLAB

Amir Ali Ahmadi
Princeton, ORFE

Slides/demos prepared in collaboration with:
Georgina Hall
Princeton, ORFE

Some slides/examples courtesy of:
Peter J. Ramadge, Ronnie Sircar
The format of this short course

- I’ll use these PowerPoint slides as a guideline
- Will run the scripts in MATLAB here for you to see
- There will be a 90-min tutorial by the TAs right after my lecture so you can try things on your own
- There is a second short course tomorrow at 6:30 PM
- My slides and demos will be posted on Blackboard
Getting started

• To install: http://www.princeton.edu/software/licenses/software/matlab/

Tip: make sure you install the offline version so you can access it when off campus.
Scalar variables and assignments

\[ \text{\texttt{>> 2+2}} \]

\[ \text{ans =} \]

\[ 4 \]

- Scalar assignments:
  
  \[ \text{\texttt{>> a=1.23}} \]

- Ending semicolon
  
  \[ \text{\texttt{>> a=1.23;}} \]

The ending semicolon is important. If you omit it, MATLAB gives an echo response.

\[ \text{\texttt{>> a=1.23456789}} \]

\[ a = \]

\[ 1.2346 \]

\[ \text{\texttt{>> format short}} \]

\[ \text{\texttt{>> pi}} \]

\[ \text{\texttt{>> format long}} \]

\[ \text{\texttt{>> pi}} \]
Basic mathematical functions

- Basic operations:
  +, -, *, /, ^

- Trigonometric functions:
  sin, cos, tan, atan,...

- Exponential functions:
  exp, log, log10, sqrt,...

- Other basic functions that come in handy:
  floor, ceil, round, mod, sign, nchoosek, ...
What is more useful than my lecture

>> help nchoosek

>> doc nchoosek

Google search: `matlab how to get binomial coefficients?`
Management functions

- Checking your variables:
  
  who, whos

- Clearing stuff...
  
  clc, clear, clear *all*, close *all*, clearvars –except*, ...

- Saving variables:
  
  save *filename* a b c

  load *filename*
Vector and matrix manipulation (1/2)

- Defining a vector/matrix:

\[
\begin{align*}
\gg a &= [1;2;3] \\
\gg A &= [1\ 2;\ 3\ 4;\ 5\ 6] \\
\gg A &= [1,2;3,4;5,6]
\end{align*}
\]

- Size of the data (row, then column) / length

\[
\begin{align*}
\gg \text{size}(A) &\gg \text{length}(A)
\end{align*}
\]

- Matrix/vector operations:

\[
\begin{align*}
+ & - \ 
* & \wedge \\
.\star & .^\wedge
\end{align*}
\]

standard \quad component-wise

\[
\begin{align*}
\gg A^{-1} &\gg \text{inv}(A) \gg A'
\end{align*}
\]

- Commonly used matrices:

\[
\begin{align*}
\gg A \&= \text{ones}(2,2) \gg A \&= \text{zeros}(2,2) \gg A \&= \text{eye}(2,2)
\end{align*}
\]
Vector and matrix manipulation (2/2)

- Concatenating matrices:

  ```
  >> a=[1;2;3]    >> b=[4;5;6]
  >> A=[a b]    >> A=[a';b']
  ```

- Random matrices:

  ```
  >> A=rand(10,10)    >> A=randn(10,5)
  >> A=randi([-5,5],10,10)    >> randn('seed',0)
  ```

- Submatrices and elements:

  ```
  >> A(2,2)    >> A(2,:)    >> A(1:2,1:2)
  >> A(5:end,6:end-1)    >> A([1 3],[2 4])
  ```

MATLAB indexing starts at 1
Logical operations

- Logical tests:

  ```
  >> A>0   >> A==1   >> A~=0   >> A<=0
  ```

  Returns a matrix of same size as A with 0s and 1s: 1 is the condition is met for that entry, 0 is the condition is not met for that entry.

- Find function:

  ```
  >> find(A>=0)
  >> [row, col] = find(A>=0);
  >> [row col]
  ```

- Some other basic operations:

  ```
  >> b = [2 7 3]
  >> [a, c] = max(b)  >> [a, c] = min(b)  >> sum(b)
  >> mean(b)
  ```
Symbolic computation

- Useful for quick differentiation, integration, evaluation, plotting, etc.

```matlab
>> syms x y t
>> f1=sin(x)*y^2+exp(y*x)
>> diff(f1,y)
>> diff(diff(f1,y),x)

>> f2=cos(t)+t^3;
>> int(f2)
>> int(f2,1,4)
>> double(ans)

>> subs(f2,t,4)
>> ezplot(f2)
```
Writing a MATLAB script

%This is my first script!
clear all %good idea usually (but make sure you don't lose your vars)
n=3; %good idea to start with your parameters

A=rand(3);
B=A*A';
eig_B=eig(B)

C=inv(B)
eig_C=eig(C);
[eig_B 1./eig_C]

- Easy debugging, access to variables
- Running the whole script, running sections
Writing a MATLAB function

- Easy to call multiple times (in a for loop e.g.)
- Essential for larger projects
Typical call to a MATLAB function from a script

%Count the number of real eigenvalues of a random mxm %Gaussian matrix

clear all
close all
N=1000;
m=100;
vec=zeros(m,1);

% for i=1:N
vec(i)=num_real_eigenvalues(randn(m));
end

hist(vec)
mean(vec)
Function handles

- Quick way of creating a temporary (simple) function without making a new file

```matlab
>> g=@(x,y) exp(x)*sin(y)

ans =
1.042743656235905
```
Plotting (1/2)

- Opening a new window for a figure:
  
  ```matlab
  >> figure
  ```

- Plotting $x$ vs $y$:
  
  ```matlab
  >> x=-3:0.1:3;
  >> y=sin(x);
  >> plot(x,y)
  ```

- Plotting using different colors:
  
  ```matlab
  >> plot(x,y,'color','red')
  ```

- Plotting data points:
  
  ```matlab
  >> t=0:0.1:1
  >> x=-3:0.1:3;
  >> y=sin(x);
  >> scatter(x,y)
  ```
Plotting (2/2)

- Multiple graphs on one figure:
  ```matlab
  >> hold on;
  >> hold off;
  ```

- `ezplot` (quick plotting, without defining a vector for input variables):
  ```matlab
  >> ezplot(sin(x))
  ```

- Can also be used to plot level sets:
  ```matlab
  >> ezplot('(x-1)^4+(x*y-x^2)^2-1')
  >> hold on
  >> ezplot('(x-1)^4+(x*y-x^2)^2-4')
  ```

- Figure properties, grid, xlabel, ...

- Saving a figure

- Plotting a surface:
  ```matlab
  >> syms x y
  >> ezsurf(sin(x*y))
  ```
  ```matlab
  >> x=-5:0.1:5;
  >> y=-5:0.1:5;
  >> [X,Y]=meshgrid(x,y);
  >> Z=X.^2+Y.^2;
  >> surf(X,Y,Z)
  ```
If/else statements

- If... then... else...

```matlab
>> flag=0;
a=randn(1,1);
if a>=0 || a<=-2
    flag=1;
elseif a<=0 && a>=-1
    flag=2;
else
    flag=3;
end
a
flag
```

- Checking for equalities and inequalities

```matlab
>> flagb=0;
b=2*rand;
if round(b)==1
    flagb=1;
elseif round(b)~=0
    flagb=2;
end
b
flagb
```
for/while loops

- **For loops**

  ```matlab
  >> s=0;
  for i=1:100
      s=s+i;
  end
  s
  ```

- **While loops**

  ```matlab
  >> s=1;
  while s<=100
      s=s*2;
  end
  s
  ```

- **Nested for loops**

  ```matlab
  >> A=zeros(10);
  for i=1:length(A)
      for j=1:length(A)
          A(i,j)=nchoosek(max(i,j),min(i,j));
      end
  end
  A
  ```
Practice with for loops

- Write a script that tests whether a given integer is prime

```matlab
function [ y ] = isprime_myfun( a )

% The function takes as input a number and outputs 0 or 1
% is prime or not
y=1;
for i=2:ceil(sqrt(a))
    if mod(a,i)==0
        y=0;
        break
    end
end
if a==2
    y=1;
end
```
Practice with for loops

- Write a script that lists all primes up to an integer N

```matlab
function [nber_primes, vec_primes] = allprimes(ub)
%Provides the number of primes and a list of the primes that are
nber_primes=0;
vec_primes=[];
for i=2:ub
    if isprime_myfun(i)==1
        nber_primes=nber_primes+1;
        vec_primes=[vec_primes; i];
    end
end
```
Vectorized computation

- Whenever possible, replace for loops with vectorized computation
  - More readable
  - Less error prone
  - Better performance

```
>> clear all
n=2000;
A=randn(n);
B=randn(n);

tic
for i=1:n
    for j=1:n
        C(i,j)=A(i,j)*B(i,j);
    end
end
toc

tic
C=A.*B;
toc
```
Vectorized computation

Task: Compute and plot the function:

\[ y(t) = e^{-2t}(2 \sin(\pi t) + 3 \cos(\pi t)) \]

```matlab
>> t=-2:.01:2;
>> y=exp(-2*t).*((2*sin(pi*t))+3*cos(pi*t));
>> plot(t,y,'-')
```
MATLAB toolboxes

- Collection of m-files for a specific problem domain
- You will most likely come across some toolboxes depending on your interests
Some basic image processing

- Converting an image to black and white

```matlab
close all
RGB = imread('tiger.JPG');
imshow(RGB)
I = rgb2gray(RGB);
figure
imshow(I)
imwrite(I,'bwtiger.JPG')
```
Image compression

- Compressing an image using the singular value decomposition

```matlab
close all
A = imread('nash.jpg');
A = im2double(A);
A = rgb2gray(A);
[m, n] = size(A);
figure, imshow(A)

k = 100;
[U, Sigma, V] = svd(A);
Uk = U(:, 1:k);
Sigmak = Sigma(1:k, 1:k);
Vk = V(:, 1:k);
Ak = Uk * Sigmak * Vk';
figure, imshow(Ak)
pixels_saved = m * n - (n + m + 1) * k
imwrite(I, 'compressed_nash.JPG')
```
When stuck, you know where to go...

```matlab
>> help functionname
```

```matlab
>> doc functionname
```

(Can contribute back to the MATLAB community on MATLAB Central)