Machine Learning

Prof. Jose L. Mendoza-Cortes

Scientific Computing Department, Dirac Science Building
Materials Science and Engineering, High Performance Materials Institute
Florida State University
jmendozacortes@fsu.edu

Condensed Matter Theory, National High Magnetic Field Laboratory
Florida State University
mendoza@magnet.fsu.edu

Chemical and Biomedical Engineering
Florida State University — Florida A&M University — College of Engineering
mendoza@eng.famu.fsu.edu

Web: http://mendoza.eng.fsu.edu/

Functions and Scripts: Introduction

Notes: Functions and Visualization

1. Review last class
2. Functions
3. Visualization
4. Combination with flow diagram

Remember

- The percent sign % denotes the start of a comment, and MATLAB ignores it.
- The operators `.*` and `./` tell MATLAB to do element by element multiplication

How to look for Help

- You can always get help on a command (say `plot`) by typing `help plot` in MATLAB's command window.
- You can also use the upper right corner section called "Search Documentation"
- And of course, there is also Google. Just make sure that in your search you include 'MATLAB and the question'

Functions and Scripts: Introduction

It is far more powerful, and liberating to use it in a "batch" mode using:

Functions

provide definitions of new functions

Example file `Pressure.m`

```
function P = Pressure(T,V,R)
% Pressure(a,b,c) returns
% ideal gas pressure
P = R.*T./V;
end
```

You can use this function from the command window as:
`Pressure(298,0.022,8.314)`

scripts

list of commands to be executed.

Example file `IdealGasScript.m`

```
% script M-file IdealGasScript.m
R = 8.314; % Gas Constant
T = 298;
V = 0.0022;
Pressure(T,V,R)
```

You can run this file from the `Run Script` menu or by typing the filename without the .m on the command line.

Functions and Scripts: Introduction

Be careful

Functions

- When you call the function `Pressure` MATLAB looks for the file `Pressure.m` in the current directory and in the MATLAB search path.
- If you want to change a function or a script you must tell MATLAB to use the new version by entering the command `clear Pressure` from the command line to remove the old version from memory. The command `clear all` removes all definitions so can start with a blank slate of MATLAB. Use this command with great care.

scripts

- The script file `IdealGasScript.m` must be in your MATLAB path
- If script files are not doing what you expect the command `clear all` will remove all old definitions from memory. This is often necessary when you are changing the script file. Again, "with great care, you should use this command"

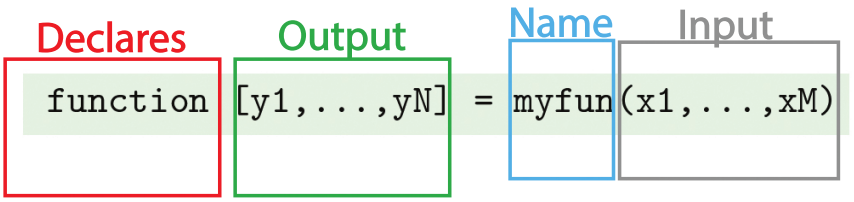
Functions and Scripts: Definition

Functions (according to MATLAB)

Syntax

```
function [y1,...,yN] = funcname(x1,...,xM)
```

- A function files is defined by writing `function` in the first line
- That accepts inputs `x1,...,xM` %notice the ( )
- Return outputs `y1,...,yN` %notice the [ ]
- The name of the file should match the name of the first function in the file (e.g. `funcname.m`). Valid function names begin with an alphabetic character, and can contain letters, numbers, or underscores.



Functions and Scripts: Definition

Functions (Helpcomments definition)

Syntax

```
function [y1,...,yN] = funcname(x1,...,xM)
%HELPCOMMENTS (1st line): Description of the function
%HELPCOMMENTS (2nd line): Use of the function (How to)
%HELPCOMMENTS (3rd line): Definition of Inputs
%HELPCOMMENTS (4th line): Definition of Outputs
```

Helpcomments

- The HELPCOMMENTS section is a very important part of creating a function, perhaps the most important
- 1st line of HELPCOMMENTS will be found by the command `>>lookfor word` where word is in the 1st line of your function
- All the lines from 1st - *n* line in Helpcomments will be found by the command `>>Help funcname`

Functions and Scripts: Examples

File: `Absolute.m`

Using the HELPCOMMENTS

```
function [ numberabs ] = Absolute( anynumber )
%Description:This func takes the absolute value of a number
%Use:         numberabs = Absolute (anynumber)
%input:       number = any number (positive/negative)
%output:      numberabs = absolute of a number
  if anynumber < 0
  numberabs=-1.*anynumber; %transf a neg into pos
  else
  numberabs=anynumber;
  end
end
```

Now in Matlab type the commands `lookfor Absolute` vs `Help Absolute`. What is the difference?

Functions and Scripts: Examples

Solve: $\sum_{i=1}^{10} |i-5|$

script

File: `scriptNofunction.m`

```
sum=0
for i=1:10
    i
    if i-5<0
        sum=sum-1*(i-5)
    else
        sum=sum+(i-5)
    end
end
```

Function (No Helpcomments used)

File: `Absolute.m`

```
function [out] = Absolute (in)
    if in < 0
        out=-1*in
    else
        out=in
    end
end
```

File: `scriptYesfunction.m`

```
sum=0
for i=1:10
%This calls Function Absolute
    sum=sum+Abs(i-5)
end
```

Functions and Scripts: Examples

If a function has more than one output, you can save both by using an array with matrices variable in the call.

File: `MySum.m`

Function

```
function [theArray theSum] = MySum(start,final)
count=1; theSum = 0
for i=start:final
    theSum=theSum+i
    theArray(count)=i
    count=count+1
end
```

File: `lazysum.m`

script

```
%lazy to make the sum everytime?, call your function
[theArray theSum] = MySum(3,5)
```


MATLAB - plot and fplot: Definition

Syntax

`plot(X,Y)`

creates a 2-D line plot of the data in Y versus the corresponding values in X.

```
x=0:10
y=x.^2
plot(x,y)

x=0:10
y=x.^2
plot(x,y)
hold on
plot(x,x)
```

Syntax

`fplot(fun,limits)`

plots a function between specified limits. `limits` is a vector specifying the x-axis limits (`[xmin xmax]`).

```
fplot(@Abs, [-10 10])
% Check the values around 0
fplot(@Abs, [-10 10], 100)
% What happened to the plot?
```

Notice the use of our earlier `function [out] = Abs (in);`

MATLAB - plot and fplot: Examples

File: `plotgas.m`

```
R = 8.314;
T1 = 298;
T2 = 398;
T3 = 498;
% Create an array of Volume
% values from 0.01 to 0.1
%V = linspace(0.01,0.1,100);
V = 0.01:0.001:0.1;
% Fill the data arrays
P1 = Pressure(T1,V,R);
P2 = Pressure(T2,V,R);
P3 = Pressure(T3,V,R);
%continuation
plot(V,P1,V,P2,V,P3)
title('Ideal Gas Plot')
xlabel('Molar Volume')
ylabel('Pressure')
legend('T=298','T=398','T=498')
```

Exercise

*Change the value of
`V = 0.01:0.0001:0.1;`
*What does this definition do:?
`V = linspace(0.01,0.1,100);`

Notice the use of our earlier function `P = Pressure(T,V,R);`

MATLAB - plot and fplot: Examples

Compare to `plotgas.m`

File: `plotgasalt.m`

```
close all;
p1 = @(x) Pressure(298,x,8.314);
fplot(p1,[0.01 0.1],'g-')
hold on
p1 = @(x) Pressure(398,x,8.314);
fplot(p1,[0.01 0.1],'r:')
p1 = @(x) Pressure(498,x,8.314);
fplot(p1,[0.01 0.1],'b')
```

Notice the use of our earlier function `P = Pressure(T,V,R);`

Notice the use of `'g-'` and `'r:'` and `'b'`

The first letter in the code sets the color of the line.

Symbol	Color
g	green
b	blue
r	red
c	cyan
m	magenta
y	yellow
k	black
w	white

MATLAB - plot and fplot: Examples

The next symbol is the Line Style

Symbol	Line Style
-	solid line
--	dashed line
:	dotted line
-.	dash-dot line

You can add other symbols to your plot, called marker for the line.

Symbol	Marker Style	Symbol	Marker Style
.	point	v	triangle (down)
x	cross	^	triangle (up)
+	plus	<	triangle (left)
s	square	>	triangle (right)
d	diamond	*	asterisk
p	pentagram	h	hexagram

MATLAB - subplot: Definition

Syntax:
`subplot(m,n,p)`
divides the current figure into an `m`-by-`n` grid and creates an axes for a subplot in the position specified by `p`.

```
x = linspace(0,10);

figure(1)
subplot(2,1,1);
p1 = @(x) Pressure(298,x,8.314);
fplot(p1,[0.01 0.1],'g-x')

subplot(2,1,2);
p1 = @(x) Pressure(398,x,8.314);
fplot(p1,[0.01 0.1],'r:d')
```

File: `plotsubplot.m`

```
clc; clear; close all;

x=0:10;
y=x.^2;
subplot(2,2,1)
plot(x,y)

subplot(2,2,2)
plot(x,y)

subplot(2,2,[3 4])
fplot(@Abs, [-10 10])
```

Other commands to remember

Type this in the command window

```
>> help func     %What does this command do?
>> who           %What does this command do?
>> whos          %What is the difference with >>who ?
>> zeros (3,4)   %What does this command do?
>> ones (5,2)    %What does this command do?
>> linspace(1,10,100) %Compare to >> 1:0.1:10
>> logspace(1,10,100)
>> M=[[1,2,3]'[4,5,6]'[7,8,9]']  %What does command do?
>> log(M)                        %What does command do?
>> length(M)                     %What does command do?
```

Really look up for what these command do

`length(), logspace(), linspace(), hold on`

Other commands to remember

Code 1: Concatenating text

```
1 n='Dark '
2 l='lord'
3 donotsayit = [n l]
4 isaidno = ['Dark ' 'lord'] %compare to donotsayit
```

Code 2: Making Tables

```
1 a=[20; 21; 22];
2 b={'Eng'; 'Sci'; 'Sci'} %Notice the {} instead of []
3 X=table(a, b, 'VariableNames',{'age','Major'})
```

Test these commands/script in your computer

See you next class

"Just as there is not royal road to geometry, there is no royal road to programming".- Euclid and J. V. Guttag

The computer will do what you TELL them to do NOT what you WANT them to do.- Someone in the Internet (Perhaps)

Think twice, code once.- Anonymous

The sooner you start to code, the longer the program will take.- R. Carlson

Any fool can write code that a computer can understand. Good programmers write code that humans can understand.- M. Fowler

Simplicity is the soul of efficiency.- A. Freeman

If you cannot grok the overall structure of a program while taking a shower, you are not ready to code it.- R. Pattis

Appendix: Scripts included

Code 3: exercise_inputtypes.m

```
clc; clear;
%n = input ('text here: ')
name = input ('what is your name: ','s');
dis = (name) %value can be a number, string, expression
```

Try these commands in your own workstation, i.e. have the lectures on one half side of your screen and Matlab/Octave-GUI on the other half. This is the best approach to learning this.

Check the scripts/functions under the directory for this note number (X): /NX_Notes_directory