Matlab or Octave

We'll be primarily using Matlab in class, although Octave is an open source clone. For Spring 2021, we'll be trying out Octave Online so that you won't need to actually install Octave on your home machine (although you certainly may if you want). Matlab was originally set up as a front end to linear algebra numerical routines, and so arrays form its basic structure and make the syntax fairly straightforward.

Matlab physically resides on each of the computers in the Olin Hall labs. See your instructor if you need an account on these machines.

If you are going to be going into engineering at some point, Matlab is available to download from the Mathworks website (student version). Its a good deal- $99 for Matlab and 10 of the most used toolboxes (some of which we don't have).

1 Getting Things Set Up

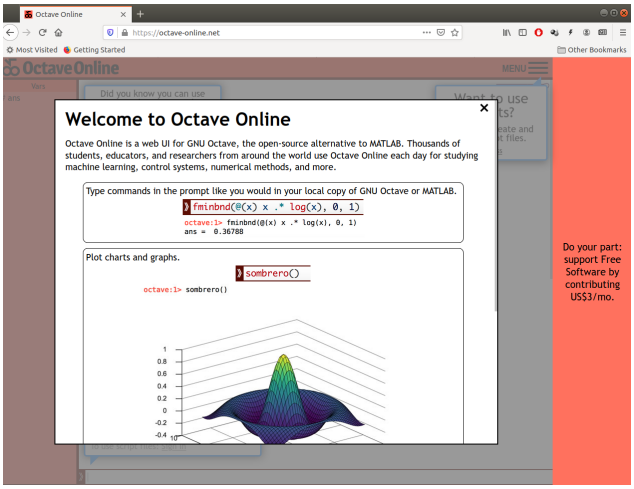
Octave Online

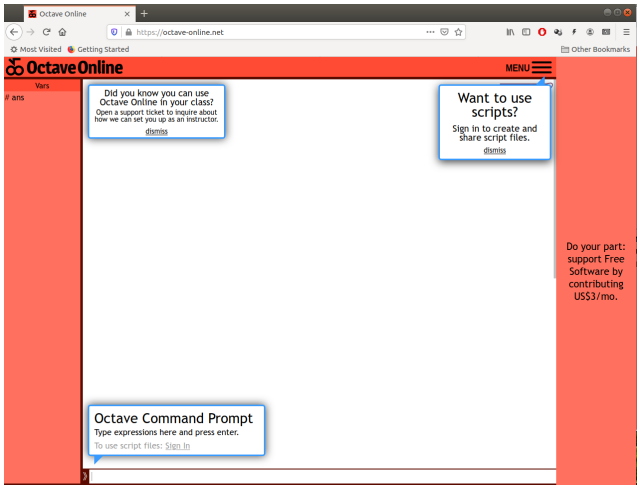
Go to the following website to get registered- you should probably use your Whitman email.

`https://octave-online.net/`

After you have registered, sign in. You may type commands now, but you should "enroll" in our class- This will give us a chance to collaborate using Octave (share code). In the command window, type the following (only for Spring semester, 2021):

```
enroll("WHITMAN_MATH/CS350_2021Sp")
```





Once registered, you'll see a panel on the left. This panel will act as an interface to uploading and downloading into a file space. You may also create directories and subdirectories there (more on that later). Downloading files is done from the icon menu, and uploading can be done by mouse ('Drag files here to upload').

1.1 How does Matlab (or Octave) work?

You can make Matlab do computations three different ways:

- Type commands directly into the keyboard in the command window.

- Have your Matlab commands typed into a separate text file (called a **script file**), and then have Matlab read these commands in. This is very nice- it gives you documentation and allows you to run similar computations several times without having to re-type the commands. We will typically use script files to do homework problems.

  To turn in homework, you will "publish" your script, then upload the PDF to your CLEO dropbox (See our course website for more information).

- Define your own functions by typing a separate text file (called an **m-file**).

1.2 Moving Files Around- Octave Online

You can create a script file using any text editor, then upload the file to Octave online (drag-and-drop to the left panel). You can also press the "Create an Empty File" button on the upper left icon on the left panel, then create your file.

There are Matlab commands to create folders and move files around, and these are available on Octave online.

`mkdir('MyDirectory')`	Makes a local directory named `MyDirectory`. Please do not put white space in your file names or folder names!!

Now create an m-file named `my_sample.m`. It doesn't need to be much, just something for us to work with below. When you're finished, proceed.

`movefile my_sample.m MyDirectory/`	Moves the file `my_sample.m` to the directory named.

`cd MyDirectory`
`ls`
`cd`
	`cd` is the command to "Change Directory". Further, the command `ls` (that's small case L, then s) will list the contents of the directory. In the last case, using `cd` by itself will move you to your 'home' directory.

`remove filename.m` `rmdir DirectoryName`	In the first line, we remove `filename.m` from our current directory. In the second case, we remove the *empty* directory `DirectoryName`.

2 Introductory Commands

On the left, we show the commands you can type in the command window, and to the right we give some commentary. Matlab uses the standard mathematical operations:

2.1 Numbers

`2+3;` `123.3*sin(3.2)`	If you leave the semicolon off the end of the line, Matlab prints the result (this is OK when typing live, but usually not good for scripts- We'll see that later).

`2^5;`
`exp(4);`
`cos(1.3);`
`log(3)`
	The function `exp(x)` is used for e^x. Sine and cosine assume the input is in radians. The logarithm assumes natural log.

`i^2`
`(1+i)*(2-i)`
`pi`
`exp(1)`
	The number $i = \sqrt{-1}$ is defined in Matlab- But only if it hasn't been defined by you to be anything else. The constant π uses lowercase p (unlike Maple). To get the constant e, use `exp(1)`.

```
5/0
0/0
eps
```

In the first case, you'll see `Inf`, which represents "infinity". This is actually incorrect- It should be $\pm\infty$, since we don't know how the denominator is reaching zero. In the second case, Matlab gives `NaN`, which means "Not a Number". Do you know why this would be? (think limits). The last value is a very small number below which we typically would think of a quantity as "zero".

2.2 Helpful Administrative Commands

The following commands are useful as you begin to use Matlab more and more:

```
clear;
clc;
whos;
help
% Help for a specific command
help sin
doc
demo
```

The up arrow key is useful (try it!). Here we first clear the workspace memory of all variables, and secondly we clear the command window. Give the others a try to see what they do. Notice that we can use a percent sign to put in a *comment* (Comments are not processed by Matlab).

3 Assigning values to variables

To assign data to a variable name, we would use the equal sign. For example, $x = 3$ assigns the value of 3 to the variable x. The array (or matrix) is the basic data type in Matlab. Data entry is straightforward.

```
A=[1 2 3 4; 5 6 7 8];
A=[1 2 3 4
5 6 7 8];
A(2,3)
```

A is an array with two rows and 4 columns and can be entered in either manner. Inside an array, the semicolon indicates the end of a row. The $(2, 3)$ element (2d row, 3d column) of A is accessed as this.

```
x=[1;0;1]; y=[-1;2;3];
dot(x,y)
cross(x,y)
```

The dot product and cross product are available.

3.0.1 The dot

If we want to take an array of numbers and perform some operation to every value, we can use the dot. Here are some examples of how this works:

```
x=[1,4,7,10];
x.^2
sin(x)
y=[2 3 4 5];
x./y
```

The array x is first formed, then we take the square of each element. Similarly, the third line takes the sine of each element. If we define y as another array of the same size, we can "divide" x by y, where the first element will be $1/2 = 0.5$ and the last element is $10/5 = 2$.

3.0.2 Special Commands: The colon operator

- `a:b`

  Produces a vector of integers from a to b in a row.

- `a:b:c`

  Produces the numbers from a to c by adding b each time" $a, a + b, a + 2b, \ldots$, up until the last number is biggest that is less than or equal to c.

- `linspace(a,b,c)`

  Produces c numbers evenly spaced from the number a to the number b (inclusive).

```
x=2:9;
y=1:2:8
z=10:-1.3:3
w=linspace(2,5,40);
w=linspace(2,5)
```

First, x is an array with the integers from 2 to 9. Secondly, y is array with integers $1, 3, 5, 7$. Can you predict what z will be? The last line produces 40 numbers evenly spaced beginning with 2 and ending with 5. If we leave the third number off, we get 100 values as the default.

Matlab commands associated with Arrays

- Random arrays (handy if you just need some quick data!)

  `A=rand(m,n)` Produces an $m \times n$ array of random numbers (uniformly distributed) between 0 and 1. If you just want a single random number between 0 and 1, just type `rand`

  `A=randn(m,n)` produces an $m \times n$ array of random numbers (with a normal distribution) with zero mean and unit variance. If you want a single random number (with a normal distribution), just type `randn`

- `A=zeros(m,n)` Produces an $m \times n$ array of zeros.

- `A=ones(m,n)` Produces an $m \times n$ array of ones.

- `A=eye(n)` Produces an $n \times n$ identity matrix.

- `A=repmat(B,m,n)` Matrix A is constructed from matrix (or vector) B by replicating B m times down and n times across.

  Example: Let $B = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$. Then `A=repmat(B,2,3)` creates the array:

```
A =
     1     2     1     2     1     2
     3     4     3     4     3     4
     1     2     1     2     1     2
     3     4     3     4     3     4
```

Matrix Arithmetic

- Transposition is denoted by the single quote character `'`. That is, `A'` $= A^T$. (CAUTION: If A contains complex numbers, then `A'` is the *conjugate transpose* of A, sometimes denoted as $A^* = \bar{A}^T$)

- Matrix addition and subtraction is performed automatically and is only defined for matrices of the same size.

- Scalar addition. If we want to add a constant c to every item in an array A, type: `A+c`

- Scalar Multiplication: We can multiply every number in the array by a constant: If A is the array and c is the constant, we would write: `B=c*A`

- Matrix Multiplication: Use the regular multiplication sign for standard matrix multiplication. If A is $m \times n$ and B is $n \times p$, then `A*B` is an $m \times p$ matrix, as we did in linear algebra.

- Elementwise Multiplication. We can multiply and divide the elements of an array A and an array B *elementwise* by `A.*B` and `A./B`

  Exponentiation is done in a similar way. To square every element of an array A, we would write: `A.^2` This is the same as saying `A.*A`

- Functions applied to arrays: Matlab will automatically apply a given function to each element of the array. For example, `sin(A)` will apply the sine function to each element of the array A, and `exp(A)` will apply e^x to each element of the array. If you write your own functions, you should always decide ahead of time how you want the function to operate on a matrix.

Exercise Set I

1. What is the Matlab command to create the array x which holds the integers: $2, 5, 8, 11, \ldots 89$
2. (Referring to the array above) What would the Matlab command be that zeros out the even-numbered indices (That is, $x(2), x(4), x(6), \ldots$)?
3. What is the difference in Matlab between typing: `x=[1 2 3]` and `x=[1,2,3]` and `x=[1;2;3]`? What happens if you type a semicolon at the end of the commands (i.e., `x=[1 2 3];`)?
4. (Referring to the last question) For each of those, what happens if you type `x.^2+3`? What happens if you forget the period (e.g., `x^2+3`)
5. What do the following commands do: `x=2;3;6;`, `x=2:3:6;`, `a=2.3:0.5:3.5;`
6. Describe the output for each of the following Matlab commands. Recall that typing a semicolon at the end of the line suppresses Matlab output- to see the results, leave off the semicolon.

```
A=rand(3,4);
A([1,2],3)=zeros(2,1);
B=sin(A);
C=B+6;
D=2*B';
E=A./2;
F=sum(A.*A);
```

7. What will Matlab do if you type in:

```
A=rand(3,4);
A(:)
A(7)
```

   NOTE: This is very bad programming style! Don't do it unless you know what you're doing!!

8. What is the Matlab command to perform the following:
   (a) Given an array x, add 3 to each of its values.
   (b) Given an array A, remove its first column and assign the result to a new array B.
9. What will the following code fragment do?

```
a=1:10;
for k=1:10
  h=ceil(length(a)*rand);
  b(k)=a(h);
  a(h)=[];
end
```

   Compare this with `a=ceil(10*rand(10,1))` and `a=randperm(10)`

10. Use the Quick Summary sheet to help you write a code fragment that takes a random matrix X and re-sorts the columns so that the first column has the smallest size and the last column has the greatest size.

Exercise Set II

1. Let x be a row. What happens if you type `plot(x)`?
2. Write a Matlab script file to plot $y = \sin(x)$ in red, $y = \sin(2x)$ in black, and $y = \sin(3x)$ in green, all on the same plot. You can assume that $x \in [-4, 8]$.
3. When we compute with numbers, some errors can occur. Try typing each of the following into Matlab, and see what happens:
   - `eps` (This is machine epsilon)
   - `1/0` (Think about what this means before trying it!)
   - `-1/0`
   - `0/0`
   - `1/Inf`
   - `Inf+Inf, Inf-Inf, Inf/Inf, Inf/0`
4. Try to reason out what you think Matlab will do with each of the following, then type it in and record what you get:

```
x=[1 3 2 1 3];
max(x)
[vals, idx]=sort(x)
find(x==max(x))
mean(x)
sum(x)
```

5. Write a Matlab function that will take in two matrices A, B of the same dimensions, and will output a matrix C so that
$$C(i,j) = \max\{A(i,j), B(i,j)\}$$
The function will use the built-in `max` function- See Matlab's help file by typing (in the command window) `doc max`
6. What does this function do (think about what each line does individually; use the help features in Matlab)? The input B is an $m \times n$ matrix.

```
function A=mystery(B)

[m,n]=size(B);
Temp=sqrt(sum(B.*B));
A=B./repmat(Temp,m,1);
```

(Hint: It is often very useful to have a matrix where each column has unit length).