MATLAB Tutorial - Week 1

NOTE: at any point, typing `help` *`functionname`* in the Command window will give you a description and examples for the specified function. This is very useful and should not be used in moderation!

1 Define/change variables: working from the 'Command Window'

NOTE: adding a semicolon at the end of a command prevent the result from being displayed in the command window.

- They are several data "class". The main ones are integers and strings. String refers to a sentence and is created using apostrophes:

`A='I already love matlab'` $\Rightarrow$ A is a string

`A=2` $\Rightarrow$ A is an integer

- Create a 3x3 matrix called `A`

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

- Display `A`'s first column and last row (*hint*: use colons and `end`)
- Replace the fifth value of A by a NaN
- Create a 20x5 matrix, `B`

$$B = \begin{bmatrix} 1 & ... & 5 \\ 6 & ... & 10 \\ ... & ... & ... \\ 96 & ... & 100 \end{bmatrix};$$

hint: Create an array first, then use `reshape()`. You do not have to type every components. Colons can be used as a shortcut: `B=1:1:100;` means "From 1 to 100, with an increment of 1"

- Clear your workspace

2 Arithmetic and operation

- Create two 2x2 matrices, A and B
- Compare the results of the following commands:
  - `A*B` with `A.*B`
  - `A/B` with `A./B`
  - `A*A` with Aˆ2 and $A.$ˆ2

You will note that operations in MATLAB follow the same rules as classical operation rules for matrices. The “dot” is used to specify that you want an element-wise operation.

- Clear your workspace

3 Useful commands and missing values

- Create a random array (5x1) A, using `rand()`
- Calculate the mean, maximum and minimum of A (Use `min, max, mean`)
- Replace one of A’s value by a NaN
- Calculate the mean, maximum and minimum of A

For data analysis, self-explanatory commands can be used: `min, max, mean, sum, var,...`. If your matrix contains NaN(s), these commands will return NaNs. To avoid this issue (data are often missing), you can use the equivalent commands that ignore NaNs: `nanmin, nanmean, ...`

4 Plotting

- Create the array `t`

```
t=-2*pi:0.1:2*pi;
```

Don't worry, Matlab know what `pi` is

- Plot sine of t versus t, using the `sin()` and `plot()` command. You can add arguments to the `plot` function to change the color, line width, line style, etc of your plot. Experiment different combinations:

`plot(t,sin(t))` classic

`plot(t,sin(t),'r')` red curve

`plot(t,sin(t),'--g')` dashed, green curve

`plot(t,sin(t),'--gs')` dashed, green curve with square markers

If you want to change a lot of parameters, you can use a "handle". A handle is a variable that contains all the specifications associated with your curve. Try:

```
h=plot(t,sin(t));get(h)
```

The `get()` command will list all the specifications you can modify. Use the `set()` command to modify your curve:

```
set(h,'linewidth',2,'marker','x','color','r','linestyle','--','markersize',4)
```

NOTE: the `sin(), cos(), tan(),...` commands are in radians. If you are working with degrees, you need to convert to radians! Either you know the formula, either you can never remember it (like me) and can use the `deg2rad()` and `rad2deg()` built-in commands

- Change the limits of the x axis to [-2*pi 2*pi] and y-axis to [-2 2]. You can use both `xlim()` and `ylim()` commands, or just the one `axis()` command

- Label your plot's axis using the following commands: `xlabel(), ylabel(), title()`

- You can use the `hold on` command to hold on to your plot, and super-imposed any future plots onto it. Try plotting cos(t) on top of sin(t), in 2 different colors. You can add a legend using the `legend` command:

```
legend('sin(t)','cos(t)');
```

- If you want to plot 2 graphs in 2 different windows, `figure` can be used to create a new figure window. Try plotting cos(t) and sin(t), in 2 different windows.

- Want 2 separated plots in the same window? Have a look at subplot! (`help subplot`)