Practical 1: Submit solutions in Matlab Grader before Fri 9th August at 5pm.

Bring your lecture notes with you for reference.

The aim of this practical is to revise some of the basic commands within MATLAB and to solve your first optimisation problem. You are expected to know the basics already. If you do not know any MATLAB, you should look at the MATLAB notes available on my web page.

1. Start MATLAB using the start menu (bottom left)
   - You can start by typing commands in the command window.
   - Typing `help` <*command*> will give you more information.
   - You will want to create a `.m` script file in which to keep your commands.
     - To create this file use the `NEW->Script` menu button.
     - MATLAB will bring up an editor, and you can use this to write commands in the file, save it, and do other tasks much as with Word, or other editors.
     - Make a note of where the file is saved. MATLAB searches this directory for files to use (put downloaded files here for instance).
     - You can call the file by typing its name minus the `.m` at the command prompt.
2. MATLAB is very good for performing matrix and vector arithmetic, but its rules aren't exactly the same as the rules you may have learnt elsewhere in maths. This question is aimed at teaching you some of these rules.
   (a) Write a script, called `vec_and_mat.m`, that inputs the following vectors and matrices into MATLAB.
   $$A = \begin{pmatrix} 0 & 1 \\ 3 & 2 \end{pmatrix}, \quad B = \begin{pmatrix} -1 & 0 & 1 \\ 0 & -2 & 0 \end{pmatrix}, \quad x = \begin{pmatrix} 1 \\ 2 \end{pmatrix}, \quad y = \begin{pmatrix} 3 & 4 \end{pmatrix},$$
   $$z = (1, 2, 3, \ldots, 10), \quad Z = (20, 18, 16, \ldots, 2)$$
   Write (with pen and paper) the size and type of each variable (*don't use* MATLAB *yet*).
   (b) With a pen and paper (*don't use* MATLAB), determine what results you expect from the following calculations:

(i)	`a1 = 2 * A`	(vii)	`a7 = x .* y'`
(ii)	`a2 = A + 4`	(viii)	`a8 = A.^2`
(iii)	`a3 = A * B`	(ix)	`a9 = A * x+1`
(iv)	`a4 = B(:,2)+x`	(x)	`a10 = z(1:2:5) < Z(end-2:end)`
(v)	`a5 = x * y`	(xi)	`a11 = A/x'`
(vi)	`a6 = x'* y'`	(xii)	`a12 = 9:-3:0`

   (c) Add these calculations to your script, run it, and then compare the results to what you wrote down. Also use the command `whos` to find the size and type of the variables. Make sure you understand if there are any differences between the results and your estimates. Use `help ops` if you have problems.
   (d) **Check your results with Matlab Grader (see below).**
3. Take the summation notation $z_{ij} = \sum_{k=1}^{n} x_{ik} y_{kj}$ and express it in MATLAB, presuming $X = [x_{ik}]$ and $Y = [y_{kj}]$ are $m \times n$ and $n \times \ell$ matrices, respectively.

4. We will be using MATLAB "GRADER" to

   (a) help you get feedback on your MATLAB coding; and

   (b) let me check how much of the practicals you finish.

   To use MATLAB GRADER, you need to create an account on Mathworks. Check your email – there should be an invitation to join.

   (a) Click "View Invitation" in the email message. You should be directed to the invitation on the Matlab Grader website.

   (b) If you don't already have a MathWorks Account you will have to create one (it's free). Then you can login.

   (c) Accept the terms and conditions and click Enroll to complete the enrolment.

   If you do not enroll in time, the invitation will lapse!

   Once you have joined you should be able to see our course. Go to this course, and then select "Practical 1". You should see several questions, corresponding to this sheet. This week we will start by checking questions 2(b), 2(c), and 5 in Grader to make we can make everything work. Do each in turn, and **make sure to submit your results before the due date.**

   You can work out your solutions with MATLAB, and then cut and paste them into the Grader Solution box, but the Solution box will contain a skeleton of the results, which might give you a hint.

   You can hit the "Run Pretest" button to get some feedback about your result. When you are happy with it, hit "Submit". You can submit as often as you like by returning to the problem and clicking "Try Again". Matlab Grader saves all of your submissions. The solution that passes the most tests and has the smallest size will be used.

   If you need extra help with Matlab Grader, then look at the MATLAB page `https://au.mathworks.com/help/matlabgrader/for-students-enroll-in-courses-and-solve-problems.html` which has more detailed instructions on how to use Grader.

   Note that although Grader "scores" each persons solution based on program size, we won't be using this score in this course (personally, I don't think 'size' is a very good metric for writing good code).

5. Enter and solve the optimisation problem from Lecture 1.

   - Type `help linprog` to learn how to use MATLAB's linear programming function. You may find additional material at
     - `http://au.mathworks.com/help/optim/ug/linprog.html`
     - `http://au.mathworks.com/matlabcentral/newsreader/view_thread/173489`
   - Enter an appropriate A, $\mathbf{b}$, and $\mathbf{c}$, and use `linprog` to solve the optimisation.
   - Hints
     - Remember the problem in the lecture is a maximisation problem, but linprog solves minimisation!
     - Remember to include the non-negativity bounds!
     - MATLAB can output more than just the solution $\mathbf{x}$. It can output the objective value, and exit flags that tell you whether the problem was feasible, etc.
   - **Check your answer with Grader.**