CS5810 Assessed Coursework 1

This assignment must be submitted by October 30th, 2017 at 12:00 (noon).
Feedback will be provided within ten working days of the submission deadline.

Learning outcomes assessed

This coursework will test some basic concepts of Matlab programming for data analysis, including writing short scripts, simple functions and basic plotting.

Instructions

Put your marking ID on your solutions. You can obtain your marking ID by running the command `getmarkingid` while logged into *linux.cim.rhul.ac.uk*. Please do **not** put your name on your solutions.

Please make a copy of your solutions before submitting. Your solutions will not be returned together with the feedback.

You will submit your work electronically by means of the submission script `submitCoursework` which can be found on *linux.cim.rhul.ac.uk*. You may resubmit your script any number of times, though only the last submission will be kept.

The submission will occur on *linux.cim.rhul.ac.uk* and the protocol is:

```
submitCoursework <your directory>
```

For example, assuming the directory with my solutions is assignment1 then I would submit by typing in the following command:

```
submitCoursework assignment1
```

and then follow the instructions accordingly.

The files you submit cannot be overwritten by anyone else, and they cannot be read by any other student. You can, however, overwrite your submission as often as you like, by resubmitting, though only the last version submitted will be kept. Submission after the deadline will be accepted but it will automatically be recorded as being late and is subject to College Regulations on late submissions. Please note that all your submissions will be graded anonymously.

IMPORTANT: In this assignment, exercises 1, 5 and 6 require you to write scripts, while exercises 2, 3 and 4 require you to write functions. You will have to submit a total of 5 files:

- A file containing all the scripts for exercises 1, 5 and 6. A template for this file, called Assignment1_scripts is provided on the course page on Moodle. You need to insert your code for each exercise where indicated.
- A file containing function *calc_crown_area* required for exercise 2
- A file containing function *flipvector* required for exercise 3
- A file containing function *evenodd* required for exercise 4
- The file *salesMetformin.dat* that you will use in exercise 1

Templates for functions *calc_crown_area*, *flipvector* and *evenodd* are also provided on the course page on Moodle. You need to insert your code where indicated.

Please do not change the above file names in your submission.

All the work you submit should be solely your own work. Coursework submissions are routinely checked for this.

EXERCISE 1 (2 marks)

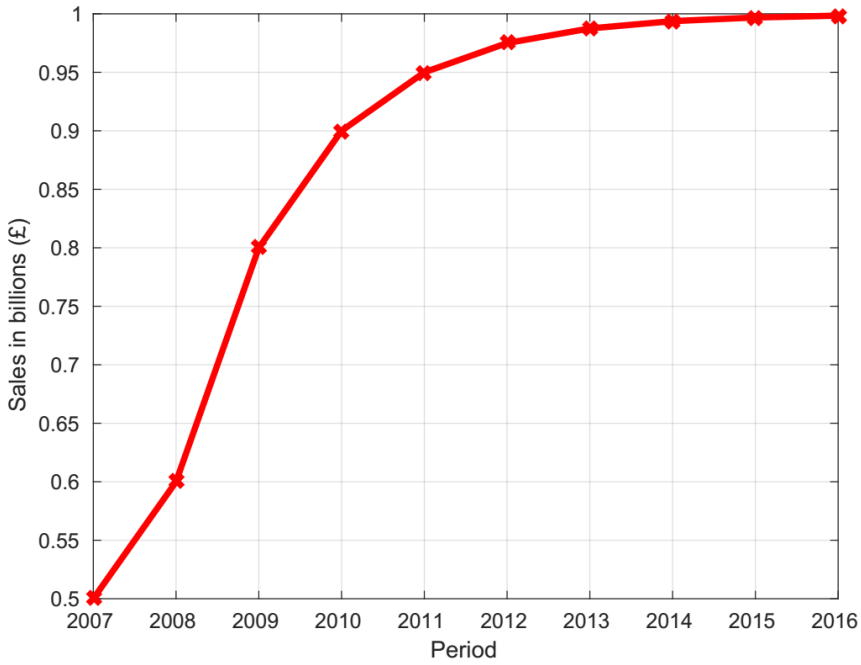
The sales per year (in £ billions) for the anti-diabetic drug Metformin for the years 2007 to 2016 was approximately as follows:

YEAR	2007	2008	2009	2010	2011	2012	2013	2014	2015	2016
SALES	0.50	0.60	0.80	0.90	0.95	0.97	0.98	0.99	0.99	1.00

First, create a text file that contains the above data on two rows, one for the *year* and one for the *sales* (you can use any editor, and then save it as "salesMetformin.dat").

Then, write a script that will:

- Load the data from the file into a matrix
- Create the plot seen in the figure below (which uses red lines and red stars as the plot symbols). Don't forget to label the axis as in the figure below.



EXERCISE 2 (3 marks)

The area of the circular crown is given by the following formula:

$$\pi * (r_2^2 - r_1^2)$$

where r_1 and r_2 are the inner and outer radius, respectively.

Write a function called *calc_crown_area* that will receive pairs of lengths of radii as input arguments, and will return the areas of the crowns. The function should work when:

- the inputs are a pair of single values, i.e. one scalar for the inner radius and one scalar for the outer radius. In this case, the function will return one single output for the area.
- the inputs are a pair of vectors, i.e. one vector of inner radii (***R1***) and a corresponding vector of outer radii (***R2***), of the same length *n*. In this case the function will return *n* values for the areas where each is calculated as $\pi * (\boldsymbol{R2}_i^2 - \boldsymbol{R1}_i^2)$ for each $i \leq n$. For this case, your code needs to handle the user error when the two vectors provided as inputs do not have the same length.
- your code needs to handle the user error when $r_1 > r_2$ for both scalar and vector inputs.

EXERCISE 3 (3 marks)

Write a function *flipvector* that will receive one input argument.

- If the input argument is a row vector, the function will reverse the order and return a new row vector.
- If the input argument is a column vector, the function will reverse the order and return a new column vector.
- If the input argument is a matrix or a scalar, the function will return the input argument unchanged.

EXERCISE 4 (3 marks)

Write a function called *evenodd* that will take in input a natural number *n* and a value *e* in the set {1,2} and then:

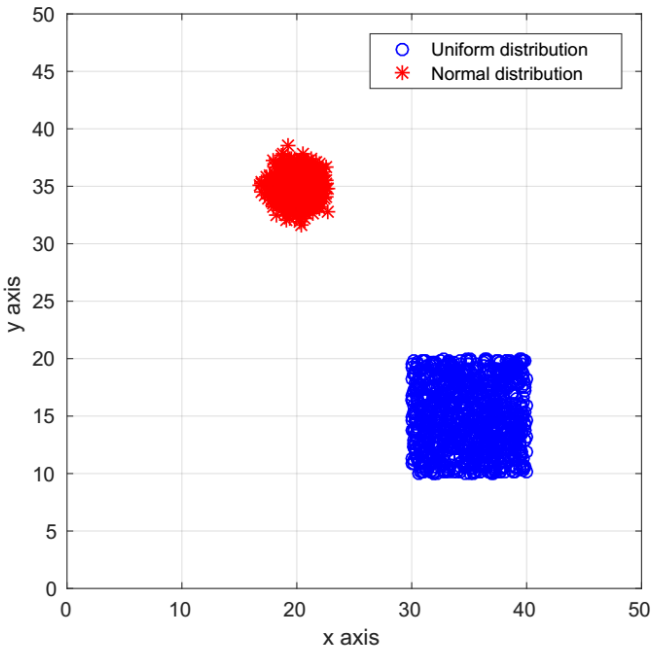
- Create a vector *v* of length *n* of random integers in the range [0 .. 30]
- If *e* is 1, return only the elements of *v* which have odd values and are placed at odd positions in *v*.
- If *e* is 2, return only the elements of *v* which have even values and are placed at even positions in *v*.

EXERCISE 5 (4 marks)

Create a script that will:

- Generate 1000 points, in 2 dimensions, which are uniformly distributed in the range: xmin=30, xmax=40, ymin=10, ymin=20.
- Plot the points, as blue circles, in a figure whose axis are set in the range xmin=0, xmax=50, ymin=0, ymin=50
- Add to the same figure 1000 points, in 2 dimensions, which are normally distributed with a mean value of (20, 35) and unit variance. These points should be denoted by red stars.
- Add the legend and the axis labels (as shown in the figure below).

Your figure should look something like the one given below.

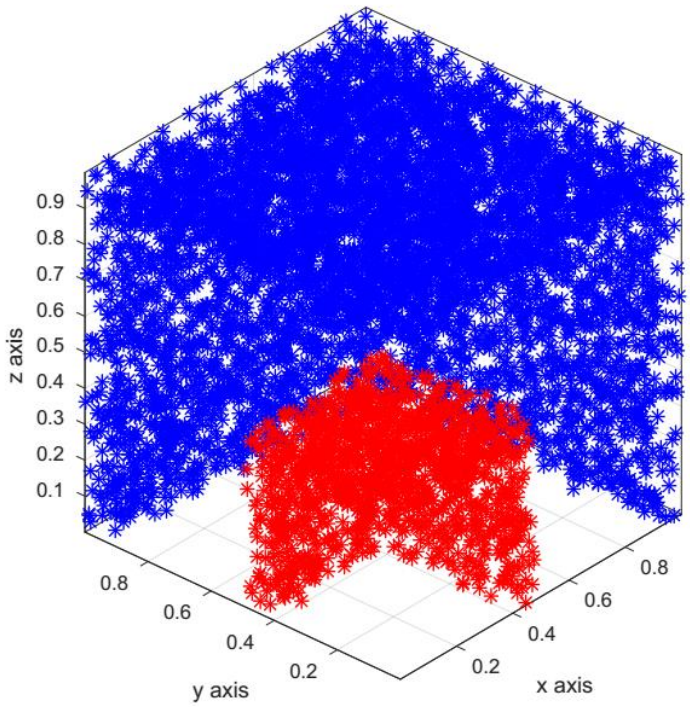


EXERCISE 6 (5 marks)

Create a script that will:

- Generate 20000 points, in 3 dimensions, uniformly distributed in the interval (0, 1).
- Extract those points for which at least one of the 3 dimensions is bigger than 0.9 and then plot them as blue stars in 3 dimensions.
- Extract those points for which both conditions are met:
  (1) at least one of the 3 dimensions is bigger than 0.4
  (2) all the dimensions are smaller than 0.5
  and then plot them as red stars in 3 dimensions.
- Add the axis labels (as shown in the figure below).

Your figure should look something like the one given below.



Marking Criteria

This coursework is assessed and mandatory and is worth 20% of your total final grade for this course.

In order to obtain full marks for each question, you must answer it correctly but also completely, based on the contents taught in this course.

Marks will be given for writing compact, vectorised code and avoiding the use of "loops" (*for* or *while* loops).