Recitation Problems – Week 1

440:127 – Spring 2015 – S. J. Orfanidis

Please do the following problems during your recitation session, including any additional problems given to you by your TA. Within *48 hours* of your recitation session, you must upload the complete solutions of these problems to Sakai, as instructed by your TA.

1. Consider the row vectors:

```
a = [5 2 1];
b = [1 2 4];
```

   Perform the following element-wise operations and then try to understand them by re-doing them by hand:

```
c1 = [a + 5;  b*5;  a + b;  a.*b;  a./b; a.\b]

c2 = [1./a + b;  (1./a) + b;  1./(a + b)]

c3 = [a.^2;  a.^2./a;  a.^2./a.^2;  a.^(2./a).^2]

c4 = [a.^2./b;  (a.^2)./b;  a.^(2./b);  a./b.^2;  a./(b.^2);   (a./b).^2]
```

   What are your conclusions about the precedence of the operations `+ * / ^` ?

   In doing this problem, you may wish to create a script M-file and place all of the above commands into it. This will save you time retyping commands in the command window.

2. Create a *function* M-file, say, **abc.m**, that accepts two row vectors a, b of the same length as inputs, and calculates the four matrices c_1, c_2, c_3, c_4 as in Problem 1. It must have usage:

```
[c1,c2,c3,c4] = abc(a,b)
```

   Run it on the example of Problem 1 and verify that you get the same results. See section 6.1 of the text on how to create function M-files, which must always start with the keyword **function**.

3. End-of-Chapter Problem 2.19. For both parts (a) and (b), calculate the temperatures using both the ideal gas law and the van der Waals equation. However, make the following changes: for part (a) use only five equally-spaced pressures in the range $0 \le P \le 400$, and for part (b), use nine equally-space volumes in the range $1 \le V \le 9$.

   Use the **char** and **num2str** functions to print your calculations for case (a) with a single Matlab command, as shown below, using two decimal places,

```
   P        T_i       T_vw
--------------------------
  0.00       0.00    125.04
100.00     601.36    689.74
200.00    1202.72   1254.43
300.00    1804.08   1819.12
400.00    2405.44   2383.81
```

   and for part (b), print them as follows with a single Matlab command, using three decimal places,

```
  V        T_i         T_vw
----------------------------
1.000    1322.994    1367.363
2.000    2645.989    2629.865
3.000    3968.983    3931.793
4.000    5291.978    5244.085
5.000    6614.972    6560.604
6.000    7937.966    7879.259
7.000    9260.961    9199.143
8.000   10583.955   10519.798
9.000   11906.950   11840.969
```

4. In a simplified version of Problem 4 of the week-1 homework set, consider the following table of stopping distances and stopping times for various car speeds (this version ignores the driver's reaction time), takne from the NJ Driver's Manual:

```
speed     braking      deceleration     braking
(mph)   distance (ft)   (ft/sec^2)     time (sec)
--------------------------------------------------
 10         12             9.34           1.57
 20         36            11.97           2.45
 30         72            13.54           3.25
 40        127            13.55           4.33
 50        216            12.47           5.88
 60        333            11.62           7.57
 70        493            10.68           9.61
```

We know from physics that a car with initial speed v, subject to constant deceleration a, will stop in time t, and travel distance s, given by:

$$t = \frac{v}{a}, \qquad s = \frac{1}{2}at^2$$

a. Given the column vectors of speeds (column 1) and braking times (column 4), calculate the corresponding column vectors of braking distances and decelerations (columns 2 & 3). You will need to convert the speeds from units of MPH to ft/sec. Arranged the computed values into a 7×4 matrix, $A = [v, s, a, t]$, and print it in the command window (the values shown above are rounded).

b. Using the built-in functions **char** and **num2str**, generate a table of values as shown below using a single Matlab command:

```
speed       braking     deceleration   braking
(mph)     distance (ft)  (ft/sec^2)   time (sec)
-------------------------------------------------
10.00        11.51         9.34         1.57
20.00        35.93        11.97         2.45
30.00        71.50        13.54         3.25
40.00       127.01        13.55         4.33
50.00       215.60        12.47         5.88
60.00       333.08        11.62         7.57
70.00       493.31        10.68         9.61
```

This can be done by using **char** to concatenate vertically the three header lines, and concatenate below those the entire matrix A, converted to a string using **num2str**, i.e.,

```
num2str(A,'%13.2f')
```

where the `13.2f` format specification allows for two decimal places and leaves enough blank spaces to fit the headers.