Introductory exercises in MATLAB

Only solve these questions if you have gone through the Matlab tutorial. **Start with solving the red-colored exercises.** It is very important for future exercises (and for the exam!) that you are able to solve all of these. Solve the other exercises at home if you feel you need more practice.

Create an m-file 'Matlab_ex1.m' in which you save the solution for all exercises below. E-mail the results to yourself such that you can continue working at home, and such that you can review your solutions for the exam.

PS: some commands in these exercises may be new (use 'help' to figure out what they do, or try to figure out yourself based on their output).

Set 1: Basic Syntax and Command-Line Exercises

1. Create a vector x of the even whole numbers between 31 and 75 (do not type each number yourself).

2. For x of exercise 1:

   a. Add 16 to each element
   b. Add 3 to just the odd-index elements
   c. Compute the square root of each element
   d. Compute the square of each element

3. Let x = [3 2 6 8]' and y = [4 1 3 5]' (NB. x and y should be column vectors).

   a. Add the sum of the elements in x to y
   b. Raise each element of x to the power specified by the corresponding element in y.
   c. Divide each element of y by the corresponding element in x
   d. Multiply each element in x by the corresponding element in y, calling the result "z".
   e. Add up the elements in z and assign the result to a variable called "w".
   f. Compute x'*y - w and interpret the result

4. Evaluate the following MATLAB expressions by hand and use MATLAB to check the answers

   a. 2 / 2 * 3
   b. 6 - 2 / 5 + 7 ^ 2 - 1
   c. 10 / 2 \ 5 - 3 + 2 * 4
   d. 3 ^ 2 / 4
   e. 3 ^ 2 ^ 2
   f. 2 + round(6 / 9 + 3 * 2) / 2 - 3
   g. 2 + floor(6 / 9 + 3 * 2) / 2 - 3
   h. 2 + ceil(6 / 9 + 3 * 2) / 2 - 3

5. Create a vector x with the following elements (do not type them by hand, i.e., you should be able to continue the series for any pre-defined length).

   a. 2, 4, 6, 8, ...
   b. 10, 8, 6, 4, 2, 0, -2, -4
   c. 1, 1/2, 1/3, 1/4, 1/5, ...
   d. 0, 1/2, 2/3, 3/4, 4/5, ...

6. Create a vector x with the elements,

$$x_n = (-1)^{n+1}/(2n-1)$$

Add up the elements of the version of this vector that has 100 elements.

7. ### (omitted)

8. Given a *vector*, t, of length n, write down the MATLAB expressions that will correctly compute the following:

a. $\ln(2 + t + t^2)$
b. $e^t(1 + \cos(3t))$
c. $\cos^2(t) + \sin^2(t)$
d. $\tan^{-1}(1)$ (this is the *inverse* tangent function)
e. cot(t)

9. Plot the functions x, x^3, e^x and e^{x2} over the interval $0 < x < 4$ ...

a. on rectangular paper
b. on semilog paper (logarithm on the y-axis)
c. on log-log paper

Be sure to use an appropriate mesh of x values to get a smooth set of curves.

10. Make a good plot (i.e., a non-choppy plot) of the function

$$f(x) = \sin(1/x)$$

for $0.01 < x < 0.1$. How did you create x so that the plot looked good?

11. #### omitted

12. Plot the expression (determined in modelling the growth of the US population)

$$P(t) = 197{,}273{,}000/(1 + e^{-0.0313(t - 1913.25)})$$

where t is the date, in years AD, using t = 1790 to 2000. What population is predicted in the year 2020?

Set 2: Basic Array Syntax and Manipulations

1. Given x = [3 1 5 7 9 2 6], explain what the following commands "mean" by summarizing the net result of the command.

a. x(3)
b. x(1:7)
c. x(1:end)
d. x(1:end-1)
e. x(6:-2:1)
f. x([1 6 2 1 1])
g. sum(x)

2. Given the array A = [2 4 1 ; 6 7 2 ; 3 5 9], provide the commands needed to

a. assign the first row of A to a new vector called x1
b. assign the last 2 rows of A to a new vector called y
c. compute the sum over the columns of A
d. compute the sum over the rows of A
e. compute the standard deviation of each column of A

3. Given the arrays x = [1 4 8], y = [2 1 5] and A = [3 1 6 ; 5 2 7], determine which of the following statements will correctly execute and provide a result. If the command will not correctly execute, state why it will not. Using the command **whos** may be helpful here.

a. x + y
b. x + A
c. x' + y
d. A - [x' y']
e. [x ; y']
f. [x ; y]
g. A - 3

4. Given the array A = [2 7 9 7 ; 3 1 5 6 ; 8 1 2 5], explain the results of the following commands:

a. A'
b. A(:,[1 4])
c. A([2 3],[3 1])
d. reshape(A,2,6)
e. A(:)
f. flipud(A)
g. fliplr(A)
h. [A A(end,:)]
i. A(1:3,:)
j. [A ; A(1:2,:)]
k. sum(A)
l. sum(A')
m. sum(A,2)
k. [[A ; sum(A)] [sum(A,2) ; sum(A(:))]]

5. Given the array A from problem 4, above, provide the command that will

a. assign the even-numbered columns of A to an array called B
b. assign the odd-numbered rows to an array called C
c. convert A into a 4-by-3 array
d. compute the reciprocal of each element of A (the reciprocal of x is 1/x)
e. compute the square-root of each element of A

6. Give the following commands to create an array with random numbers called F:

```
>> randn('seed',123456789)
>> F = randn(5,10);
```

a. Compute the mean of each column and assign the results to the elements of a vector called avg.
b. Compute the standard deviation of each column and assign the results to the elements of a vector called s.

Set 3: Exercises on Relational and Logical Operations

1. Given that x = [1 5 2 8 9 0 1] and y = [5 2 2 6 0 0 2], execute and explain the results of the following commands:

a. x > y

b. y < x
c. x == y
d. x <= y
e. y >= x
f. x | y
g. x & y
h. x & (~y)
i. (x > y) | (y < x)
j. (x > y) & (y < x)

2. The exercises here show the techniques of logical-indexing (indexing with 0-1 vectors). Given x = 1:10 and y = [3 1 5 6 8 2 9 4 7 0], execute and interpret the results of the following commands:

a. (x > 3) & (x < 8)
b. x(x > 5)
c. y(x <= 4)
d. x((x < 2) | (x >= 8))
e. y((x < 2) | (x >= 8))
f. x(y < 0)

3. Given x = [3 15 9 12 -1 0 -12 9 6 1], provide the command(s) that will

a. ... set the values of x that are positive to zero
b. ... set values that are multiples of 3 to 3 (the command 'rem' will help here)
c. ... multiply the values of x that are even by 5

Set 4: Control of Flow: if-blocks

In each of the following questions, evaluate the given MATLAB code fragments with pen and paper for each of the cases indicated. Use MATLAB to check your answers.

1.
```
if n > 1
  m = n+1
else
  m = n - 1
end
```
a. n = 7 m = ?
b. n = 0 m = ?
c. n = -10 m = ?

2.
```
if z < 5
  w = 2*z
elseif z < 10
  w = 9 - z
elseif z < 100
  w = sqrt(z)
else
  w = z
end
```
a. z = 1 w = ?
b. z = 9 w = ?
c. z = 60 w = ?
d. z = 200 w = ?

3.
```
if T < 30
  h = 2*T + 1
elseif T < 10
  h = T - 2
else
  h = 0
```
a. T = 50 h = ?
b. T = 15 h = ?
c. T = 0 h = ?

```
end
```

4.
```
if 0 < x < 10
   y = 4*x
elseif 10 < x < 40
   y = 10*x
else
   y = 500
end
```

a. x = -1 y = ?
b. x = 5 y = ?
c. x = 30 y = ?
d. x = 100 y = ?

Write brief scripts to evaluate the following functions. If you start each script with a request for input (using **input**), you'll be able to test that your code provides the correct results.

5. h(T) = T - 10 when 0 < T < 100
= 0.45 T + 900 when T > 100

Test cases: a. T = 5, h = -5
b. T = 110, h = 949.5

6. f(x) = -1 if x < 0
= 0 if x = 0
= 1 if x > 0

Compare your results to the MATLAB function **sign**.

7. t(y) = 200 when y is below 10000
= 200 + 0.1 (y - 10000) when y is between 10000 and 20000
= 1200 + 0.15 (y - 20000) when y is between 2000 and 50000
= 5700 + 0.25 (y - 50000) when y is above 50000

Test cases: a. y = 5,000 t = 200
b. y = 17000 t = 900
b. y = 25000 t = 1950
c. y = 75000 t = 11950

8. Explain why the following if-block would **not** be a correct solution to the previous exercise.

```
if y < 10000
   t = 200
elseif 10000 < y < 20000
   t = 200 + 0.1*(y - 10000)
elseif 20000 < y < 50000
   t = 1200 + 0.15*(y - 20000)
elseif y > 50000
   t = 5700 + 0.25*(y - 50000)
end
```

Set 5: Loop Constructs

1. Given the vector x, create a short set of commands that will

a. Add up the values of the elements (check with **sum**.)
b. Computes the running sum (for element j, the running sum is the sum of the elements from 1 to j, inclusive. Check with **cumsum**.)
c. computes the sine of the given x-values (should be a vector)

2. Create an M-by-N array of random numbers (use **rand**). Move through the array, element by element, and set any value that is less than 0.2 to 0 and any value that is greater than (or equal to) 0.2 to 1.

3. Given x = [4 1 6] and y = [6 2 7], compute the following arrays

a. $a_{ij} = x_i y_j$
b. $b_{ij} = x_i / y_j$
c. $c_i = x_i y_i$, then add up the elements of c.
d. $d_{ij} = x_i / (2 + x_i + y_j)$
e. e_{ij} = reciprocal of the lesser of x_i and y_j

4. Write a script (in an m-file) that will use the random-number generator **rand** to determine the following (Note that, since you are dealing with random numbers, the result may be different each time you run the m-file):

a. The number of random numbers it takes to add up to 20 (or more).
b. The number of random numbers it takes before a number between 0.8 and 0.85 occurs.
c. The number of random numbers it takes before the mean of those numbers is within 0.01 of 0.5 (the mean of this random-number generator).

5. Write an m-file that asks for a temperature (in degrees Fahrenheit) and computes the equivalent temperature in degrees Celcius. The script should keep running until no number is provided to convert. [NB. the function **isempty** will be useful here.]

Set 6: Plotting

See exercise in the Matlab tutorial