CS 251: Simple Science: Outlab

- Due: 7/30 11:55 am. Released 7/26 8:00 pm
- Please write (only if true) the honor code. If you used any source (person or thing) explicitly state it. You can find the honor code on the web page.

Overview

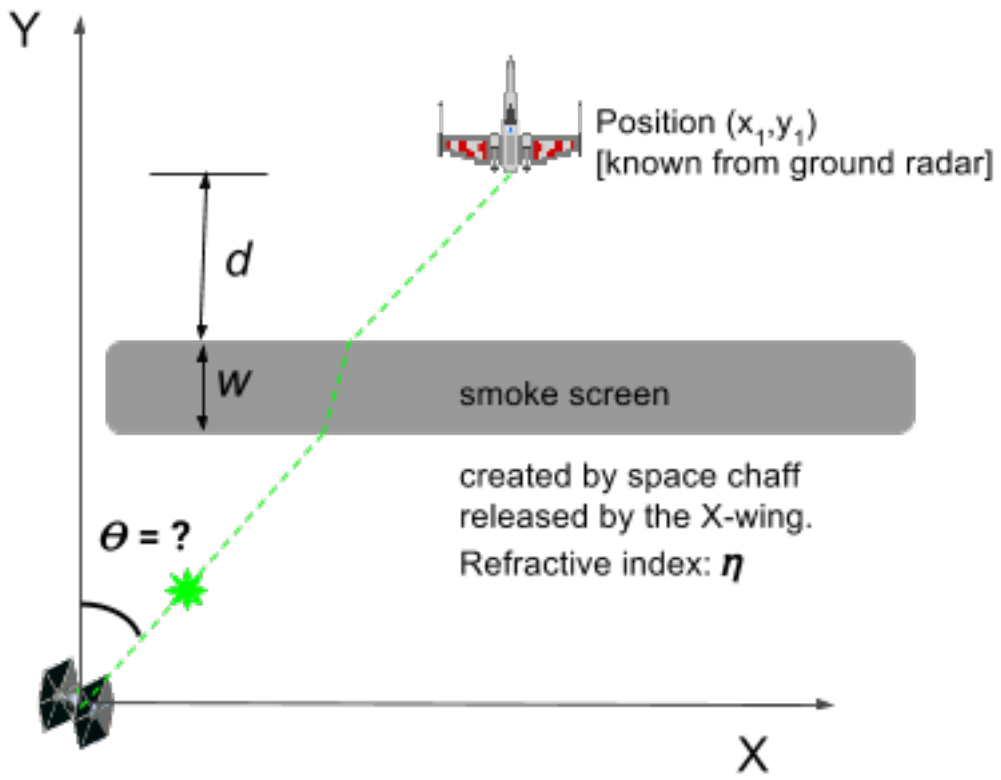
The goal of this assignment is to make you familiar with a scientific computing environment like Matlab/Octave. Octave is well documented at http://www.gnu.org/software/octave/octave.pdf. Another interesting tutorial for Octave can be found at http://www.dm.unibo.it/~lenci/teaching/14/maa/octavetut.pdf

The Empire Strikes Back!

There are *two space fighters*, a rebel *X-wing* and an imperial *TIE* fighter which are involved in a dogfight while in orbit around the Deathstar.

The *X-wing* happens to be in front of the *TIE* and is vulnerable to being shot from behind. One of the evasive measures available to the *X-wing* is space 'chaff'. Unlike chaff used by today's fighters which are essentially flares to confuse heat seeking missiles, space chaff on being released creates a thick disk shaped smoke screen behind the fighter, which obscures its position. Since the *TIE* can no longer see the *X-wing* directly, it has to rely on ground (Deathstar) based radar tracking to locate the *X-wing*. An additional complexity is that the smoke screen disk refracts the laser bursts that are fired. So aiming directly at the position provided by the radar will fail.

In this lab, we will use a **simplified model** of the above scenario as shown below:



It takes three hits to destroy the *X-wing*. Two to disable the shields and the third to destroy the unshielded hull. Note that

1. The units of distance is km, units of speed is km/s and the unit of angle is degrees.
2. The Octave code for each of the tasks is to be submitted in files named as `main_task_{number}.m` where number is the number of the task.

1. **Task A1:** Assume both the fighters are moving with identical velocity v in a direction parallel to **Y** axis. The *X-wing* is located at (x_1, y_1) relative to the *TIE* fighter. The smoke screen is at a distance d from the *X-wing*. It has a thickness of w and refractive index η. In this task, the smoke screen is also moving with the same velocity v. Determine the angle in degrees at which the *TIE* has to aim its laser cannon so that it strikes the *X-wing*. Answer is acceptable if it is within ± 0.1.

Input:	Read from the file: `input_outlab_task_A1.txt`. Position of *X-wing*: (x_1, y_1), Distance: d, Width of smoke screen: w, Refractive index of smoke screen: η (All numbers are separated by a single space)
Output:	Write to file: `output_outlab_task_A1.txt` Line 1: Angle of shooting w.r.t Y axis

Example:

Input:	180.0 100 0.05 5 3.5
Output:	62.0

2. **Task A2: (Extra Credit)** The *X-wing* accelerates for an extremely short time, and begins to move with a different speed v_2 relative to the *TIE* along the direction parallel to **Y** axis starting from (x_{X-wing}, y_{X-wing}). The smoke screen is at a distance d from the *X-wing*. The smoke screen continues to move at its original velocity v (as mentioned in previous task) and remains at the same position relative to the *TIE* fighter. Determine the angle *TIE* has to aim its laser weapon to disable the *X-wing*'s shields completely. This assumes you have succeeded in Task A1. Also determine the nearest possible position relative to *TIE* fighter at which *X-wing* will get

hit. Answer is acceptable if it is within ±0.1units. Note that the *TIE* can fire its laser cannons only every t_1 seconds starting from $t = t_1$ and that the laser has a velocity v_{laser} (comparable in magnitude to v_2).

Input:	Read from the file: `input_outlab_task_A2.txt`. Line 1: Initial Position of *X-wing*: (x_{X-wing}, y_{X-wing}), Velocity v_2, Velocity v_{laser}, Distance: d, Width of smoke screen: w, Refractive index of smoke screen: η, Time: t_1 (All numbers are separated by single space)
Output:	Write to file: `output_outlab_task_A2.txt` Line 1: Angle of shooting wrt Y axis, Position(x_f, y_f) at which *X-wing* will get hit relative to TIE (All numbers to be separated by a single space)

Example:

Input:	0.0 0.0 76.63 27 0.027 0.0555 7 5 3.5 5
Output:	41.1

3. **Task A3: (Challenge Question)** This time, to avoid being counter attacked, the *TIE* fighter accelerates again and begins to move along the circle with radius r and center of the circle (x_c, y_c) starting from (x_{TIE}, y_{TIE}) at a constant speed of v_{TIE} away from *X-wing*. *X-wing* continues to move at velocity v_{X-wing} parallel to **Y** axis starting from (x_{X-wing}, y_{X-wing}). But this time the smoke screen is fixed at a location of d from the *X-wing* and has width w. Again the *TIE*'s laser cannon can be fired only every t_1 seconds starting from $t = t_1$. Now determine the angle the *TIE* has to aim its laser cannon to finish off the *X-wing* and also the nearest possible location relative to *TIE* fighter at which the *X-wing* will get hit. Answer is acceptable if it is within ±0.1units.

Input:	Read from the file: `input_outlab_task_A3.txt`. Initial Position of *TIE*: (x_{TIE}, y_{TIE}), Initial Position of *X-wing*: (x_{X-wing}, y_{X-wing}), Speed of *TIE* fighter: v_{TIE}, Speed of *X-wing*: v_{X-wing}, Speed of laser: v_{laser} Distance: d, Width of smoke screen: w, Refractive index of smoke screen: η, Time: t_1, Center of circle: x_c, y_c (All numbers are separated by single space)
Output:	Write to file: `output_outlab_task_A3.txt` Line 1: Angle of shooting wrt Y axis, Position(x_f, y_f) at which *X-wing* will get hit (All numbers to be separated by a single space)

Example:

Input:	0.0 0.0 76.63 27 0.1 0.027 0.0555 7 10 3.5 5 3.0 0.0
Output:	39.1 76.6 101.5

How We will Grade You

The number of points per task appears below

1. Relevant comments in the code, honor code, reflection essay: 20 points.
2. Correct answer 80 points
3. Extra credit is 10%. Extra credit will be graded only if the basic task is completed correctly.
4. Challenge question carries no points to the final grade

Lights Out!

This task is related to the Light's Out puzzle that you have been introduced to during the inlab. The inlab task would have acquainted you with the mathematics underpinnings of how the puzzle works. With that in mind, we are now ready to tackle a more general version of the problem, viz., generating the solution given an initial configuration. This can be done using matrix algebra based on

the mathematical description of the puzzle. In these parts we shall **no longer** rely on lookup tables in order to solve the problem.

- **Task B1:** Write an Octave function `backward_solve_3` which takes a single argument (a 3×3 matrix representing the initial state of the board. It returns a 3×3 matrix denoting the actions to be taken to achieve lights out.

  Example

$$\text{Input} = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\texttt{backward\_solve\_3}(\text{Input}) = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}$$

Submission:	Submit the file: `backward_solve_3.m` and any other function files you may require. The function must return the appropriate value when called on the input matrix.

- **Task B2:** Write a function `backward_solve` which takes a single argument which is a matrix of any arbitrary size representing the initial state of the board and returns a equi-dimensional matrix denoting the actions to be taken to achieve lights out.
  You may assume that the cases on which your code will be tested will permit solutions (we are dealing with invertible matrices). **Do not** use external packages or toolboxes. Your code will be tested on bare installs of Octave, so stick to the primitives!

  Example

$$\text{Input} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 & 0 & 1 \end{bmatrix}$$

$$\texttt{backward\_solve}(\text{Input}) = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Submission:	Submit the file: `backward_solve.m` and any other function files you may require. The function must return the appropriate value when called on the input matrix.

How We will Grade You

The number of points per task appears below

1. Relevant comments in the code, honor code, reflection essay: 20 points.
2. Correct answer: 80 points

Submission Guidelines

1. When you submit, please document individual percentages such as Student 1: 80%, Student 2:100%, Student 3:10%. In this example, the second student will get full marks (10/10) and the first student will receive 8/10.

2. Do include a readme.txt (telling me whatever you want to tell me). Do include group members (name, roll number), group number honour code, citations etc.

3. The folder and its compressed version should both be named `lab01_groupXY_final` for example folder should be named `lab01_group07_final` and the related `tar.gz` should be named `lab01_group07_final.tar.gz`