CS1112 Fall 2010 Project 2 due Monday 9/20 at 11pm

You must work either on your own or with one partner. You may discuss background issues and general solution strategies with others, but the project you submit must be the work of just you (and your partner). If you work with a partner, you and your partner must first register as a group in CMS and then submit your work as a group.

Objectives

Completing this project will solidify your understanding of Chapters 2 and 3 in *Insight* (`for`-loops, `while`-loops, nested loops). In problem 2 you will learn to implement a simulation that includes graphics.

Ground Rule

The use of arrays and the `break` command is *not* allowed. In fact, do not use the `break` command in this course.

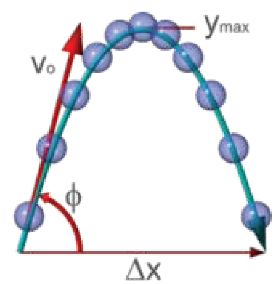
1 Verifier

Complete Problem **P2.1.7** in *Insight* (page 35). The problem asks you to verify the given inequalities for $n = 1, 2, \ldots, 100$. Your script should work through all 100 n values but produce one line of output for each value of n that is a multiple of 10. Each line of output should contain four values: n and the three values that are computed for verification for that n. The output should line up neatly in "table format." *Sanity check:* there are 10 rows of numbers in your output.

Your script should end by displaying one of these two messages: "the inequalities hold for all n values" or "the inequalities do not hold for at least one n values."

Save your script in a file `verifier.m` and submit it on CMS.

2 Trajectory of a golf ball



In your physics course you might have studied the motion of a projectile. Now you will write a simulation to plot the trajectory of a golf ball in flight, subject to air drag (resistance). Without air drag, one expects the trajectory to be a parabola due to Earth's gravity, with equal time for ascending and descending, as shown in the diagram on the right. What is the effect of air drag? You will find out!

We consider the golf ball to be a unit mass and air resistance to act in the opposite direction of motion. Further we assume that air resistance is proportional to the square of the velocity of the projectile. With these assumptions, the relevant equations are

$$
\begin{aligned}
v_x &= dx/dt \\
v_y &= dy/dt \\
dv_x/dt &= -k \cdot v_x \sqrt{v_x^2 + v_u^2} \\
dv_y/dt &= -k \cdot v_y \sqrt{v_x^2 + v_u^2} - g
\end{aligned}
$$

where v_x and v_y are the components of the velocity in the x- and y-directions, k is the coefficient of air drag, and g is the gravitational constant. The golf ball is initially at $x = 0$ and $y = 0$ and is launched with some initial velocity at an angle ϕ measured from the x-axis.

Download the file `golfTrajectory.m` from the *Projects* page. Read and run it. Since the simulation code is missing the output includes only a figure window with axes labels and "dummy values" printed to the Command Window. You will complete the simulation and replace the dummy values with the actual values calculated in the simulation.

Here are the details of our simulation:

- The constants, parameter values, and initial conditions are given in the provided code. Develop your simulation with the given values but you can (and will) experiment with different values later.
- The simulation begins with time $t = 0$ and position $x = 0$ and $y = 0$ and ends when the golf ball lands (y gets back to, or passes, zero) or the maximum allowed simulation time has been reached, whichever happens first.
- Given the initial velocity v and launch angle ϕ, the initial velocities in the x- and y-directions are $v_x = v\cos\phi$ and $v_y = v\sin\phi$
- We use *differencing* to represent the (continuous) derivatives. For example, instead of $\frac{dx}{dt}$ we consider

  $$\frac{x_{\text{new}} - x_{\text{current}}}{\triangle t}$$

  where $\triangle t$ is a discrete time step. With this discrete representation, at each time step, i.e., each step of the simulation, we can compute the new velocities and positions as follows:

  $$\begin{aligned} v_x\text{new} &= v_x - \triangle t \cdot k \cdot v_x\sqrt{v_x^2 + v_y^2} \\ v_y\text{new} &= v_y - \triangle t \cdot (k \cdot v_y\sqrt{v_x^2 + v_y^2} + g) \\ x\text{new} &= x + v_x \cdot \triangle t \\ y\text{new} &= y + v_y \cdot \triangle t \end{aligned}$$

- Starting at $t = 0$ and after each time step, plot the location of the golf ball. Recall that the command `plot(a,b,'ro')` draws a marker (red circle) at position (a,b). Choose any marker or color you like. Add a pause of 0.01 seconds (`pause(.01)`) after the plot command so that the plotting plays like a movie when the simulation is executed.

  Run the code you have so far to make sure that the simulation works! At this point you should have removed all the dummy values except perhaps those for variables `ascendTime` and `descendTime`. Does the trajectory make sense (a curve that opens down, starting and ending at around $y = 0$)? Do the displayed values in the Command Window look correct?

- Add more code to the simulation to compute the durations of ascent and descent. Do this using code based on the simulation here—do not look up equations from textbooks! Make sure all the dummy variable values are removed.
- Finally, modify the parameter values `phi`, `v`, `k`, and `maxTime` to see how the trajectory changes. Once you change the parameter values, the complete trajectory may not be shown since the axes are set to 120m wide and 100m tall and `maxTime` may be "too short," but you can use the values displayed in the Command Window to deduce the trajectory. Add a comment at the end of the script to answer these questions:

  1. Which of these launch angle results in the longest horizontal range, $\pi/3, \pi/4, \pi/5, \pi/6$ (without changing the other parameter values)?
  2. How does the flight time change with the launch angle? Answer in one sentence.
  3. What is the shape of the trajectory when $k = 0$?

  Answer the above questions simply by *running the simulation* with different parameter values. Do not write more code.

Before submitting your file `golfTrajectory.m` on CMS, change all the parameter values back to the original values given: `k=0.02`, `maxTime=10`, `phi=pi/4`, and `v=100`.

3 Diamond is for~~ever~~-loop

Complete Problem **P3.1.8** in *Insight* (page 55). Recall that `fprintf(' ')` prints one space character while `fprintf('\n')` starts a new line. Save your script as `diamond.m` and submit it on CMS.