**LAB #5 Name:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Dropping a Stone from a Bridge:**

* We are able to solve a large number of problems using pencil and paper with relatively little effort, while other problems are literally impossible to solve. These unsolvable problems can be approximated using numerical techniques. In this lab we will investigate a solvable problem numerically and compare it to the exact solution. This will help give us an idea as to the limitations of the numerical method.

Q1) Using Euler’s method, write a script in MatLab to plot the position and velocity of a stone dropped from a 25 m tall bridge. Include a printout of your script and labeled plot to turn in.

 Strategy:

Step #1 (velocity)

1. Beginning with your program from lab 4 redefine your time step as an independent, fixed value of dt = 0.1 seconds.
2. Your initial time will begin the moment you drop the stone at *t(1)* = 0 seconds.
3. In this case, you do not know your final time when the stone hits the ground. So delete any mention of a final time.
4. You will still need numerical steps as you calculate the velocity and height of the stone. We don’t know how many we’ll need but we can always fix that later. For now, we will begin with an Ansatz of N = 100.
5. How do you define your time vector now? If it is not possible or not desirable to predefine memory for your vector outside of your ‘for loop’ then it must be done inside the ‘for loop’. This can be done a number of ways. For example, define *t(i+1)* in terms of *t(i)* plus a small incremental increase. Each iteration of the ‘for loop’ will add a new value to the vector increasing its length by one with each iteration.
6. Now modify your original ‘for loop’ to solve for *v(t)* from the first order differential equation:
7. Running your program and plot the result. You should see the familiar linear decrease toward larger negative velocities.

Step #2 (height)

1. Now, solve for *y(t)* by adding an additional line inside your ‘for loop’. Essentially, you will apply Euler’s method twice in the ‘for loop’ instead of once.
2. Be sure to set your initial height as 25 meters.
3. Plot your height as a function of time. You should see a familiar parabolic curve, but also notice your stone continues to fall through the earth by about half a kilometer. This is not usually what happens to stones falling from bridges.
4. To stop the stone from falling we must include a ‘break’ statement to prematurely end the ‘for loop’ when *y >= 0*.
5. Examine your variables. Do you still have a slew of zeros at the end of your position or velocity vectors? This happens because your ‘for loop’ ends before rewriting the original zeros. When your vector will be of an unknown length then it is no longer desirable to allocate memory outside the ‘for loop’. Instead we allow the ‘for loop’ to build these vectors one element at a time.

Q2) According to your numerical results, how long does it take for the stone to fall? How did you determine this time?

Q3) Repeat your numerical calculations using at least ten different time steps (they must span at least two orders of magnitude). Use these values to create a plot of the numerical solution as a function of time step. Include a printout of your final plot.

Q4) As you decrease the size of your time step how does the solution change?

Q5) Compare your numerical results to the exact solution found using pen and paper. Show your work below.