ANALYSIS OF A BOUNCING BALL

Objective

If you drop a ball from h_0 meters above a hard, flat surface, each time the ball bounces, it rebounds a factor of r times its previous height. In this project, you will analyze the behavior of such a ball to determine rebound heights, the total vertical distance traveled by the ball, and the effect of different surfaces on these characteristics.

Restrictions

When user functions are required, they must be written as "primary" functions, saved in their own .m files. in-line functions, anonymous functions, and subfunctions are not permitted.

No "global" variables are allowed.

You can use array operations & functions, including logicals and when appropriate the "if", but NO wxplicit loops (for or while) are permitted.

Procedure

This is an individual project. **You are encouraged to <u>discuss</u> <u>verbally</u> the project with other students. You can brainstorm together solution approaches, and you can teach each other how to do things with Matlab. However, allowed collaboration ends with this verbal discussion. At no time can you copy work others have done, or have someone else do any of the work for you, or do any of the work for someone else.** Everything in the Matlab files themselves must be your work, and your work alone**. If you need more help, ask your proctor or instructor for assistance.**

Create a directory entitled **uuuuup4**, where **uuuuu** are the characters of your engineering domain username. As you work on this project, save all of the specified files in this directory. This should be set as your current working directory. When finished, you will turn in the complete directory using the dropfolders method.

Begin by creating a script file named **Main.**m. This file should begin with a "title" comment that includes your name, email address, and proctor number. This information should also be displayed in the command window when your script is run. Additional comments should briefly describe the project, the important parameters, and the required user-defined functions. The computations will be divided into the tasks described below. Preceding each task, provide the task number, e.g., "**Task 1**", followed by suitable explanatory comments. The task number and explanations should also be displayed in the command window when your script file is run. This documentation is required! In other words, both your script and your output must be easy to read with each section well delineated. Many of the tasks will require you to define functions, which will then be invoked by **Main**. Each of these functions must include appropriate comments, and must be saved in **uuuuup4**.

Be sure to put **clear, close all** and **clc** as the first executable instructions in your script.

On the course website, in the Projects/Project4 folder you will find a plain text file, zzp4data.txt. This file contains values that you will need to use in your script. Copy the file to your p4 directory. However, your solution must be kept sufficiently general so that if these values are changed the script will still execute correctly--without anybody making additional changes. We will execute your script with a different set of values! We will not provide any interactive input!

Task 1: (Read and display the data) Using the load function, input the data from the file zzp4data.txt. The data consists of values for the initial height (**hin**), initial rebound coefficient (**rin**), the initial number of bounces (**nin**), and a target height (**htar**) Store the values in the indicated variables and display them clearly labeled, assuming heights are in units of meters.

Task 2:. Define a Matlab function Height(h0,r,n) which will give the height of the ball after n bounces if the initial height is h0 and the rebound coefficient is r. Design the function to work when n is a vector. Evaluate height(15,.25,3). The result should be .23438. Define nn = [0:nin] and evaluate hnn =height(hin,rin,nn). Display the result as a two-column table with the first column being the value of n and the second column the corresponding height.

Task 3. Plot the rebound height as a function of n using the vectors nn and hnn obtained in Task 2. Designate this as Figure 1. Note that this is a plot of the maximum heights at each bounce, not a plot of the motion of the bouncing ball. To visualize the motion of the ball you would need twice as many points, since at every other point the ball hits the ground! Do this plot by constructing a new vector from hnn whose every other element is 0. For the independent coordinates, use the vector nne=[0:.5:nin+.5].This should be figure 2. Be sure your plots are properly labeled.

Task 4. Now we want to investigate the total vertical distance traveled by the ball from the initial drop. Calculate the distance the ball will have traveled from the initial drop to the apex of the 4^h bounce, assuming h_0=15 and r=.25, using your height function. Display this value, clearly labeled. You should get a value of 24.9023 meters. With this experience, define a function distance(h_0 ,r,n) which will give the total vertical distance traveled from the initial drop to the apex of the nth bounce if the initial height is h_0 and the rebound coefficient is r . Use your height function defined above. Evaluate distance(20,.3,1) which should be 26 and distance(hin,rin,100), displaying both, clearly labeled.

Task 5. Use the distance function to construct a five column table. The first column should show the number of bounces, from 0 to nin. The remaining columns should show the distance traveled after that many bounces for a ball with rebound coefficient rin if the initial heights are 5,10,15, and 20 meters, respectively.

Task 6: Now we want to analyze the motion of a different kind of ball--one whose rebound coefficient depends on how hard it strikes the ground. If the ball is dropped from 10 meters or greater, it rebounds a factor of 0.8 times its previous height. If the ball is dropped from less than 10 meters, the rebound factor is reduced to 0.6. Define a function rebound(h) which gives the rebound coefficient based on the current height h. Evaluate this function for h=5 and h=[3,10,4]. Then define a new function, vheight(h,n) that gives the vertical height of this ball on the n-th rebound if the initial height is h. Evaluate vheight(hin,nin).

Task 7: Use your function from Task 6 to produce a table and plot of the rebound heights(not the motion of the ball) for n = 0 to nin, using an initial height of hin meters. This should be figure 3

Task 8: Suppose the behavior of the ball depends on a more complicated expression for the rebound coefficient: $rc(h) = \begin{cases} .6 & if\ h < 5\ or\ h > 16 \\ .7 & if\ 5 \le h < 10 \\ .8 & if\ 10 \le h \le 16 \end{cases}$ Define the rebound function rc in *Matlab*. Plot the rebound function for h from 0 to 20. Use the axis function to use limits of 0 and 1 on the vertical axis.. This is figure 4.

Task 9. Now it seems logical to define another new height function, based on the rebound function from Task 8. But this could get tedious. A new height function, every time we change the rebound function? Not if we can pass the handle to the proper rebound function as one of the arguments! Define a new function vvheight(h,n,reb) that gives the height of a ball on the n-th rebound if the initial height is h, using the rebound function whose handle is reb. Note that this function will be identical to the one you defined in Task 6 except that it uses a different rebound function. Note that because the parameter reb is a handle, you will need to use feval in order to evaluate it. Evaluate vvheight(hin,nin,@rebound) and vvheight(hin,nin,@rc). Use vvheight to produce a table and plot of rebound heights for n = 0 to 12 if the initial height is 20 meters and the rebound function is rc. This is figure 5.

Task 10. Returning to the case of a constant rebound coefficient, if r = 1.1, how many bounces does it take for a ball dropped from hin meters to reach htar meters, where htar>hin? Use fzero and then the ceil function so that your result is an integer. Is this rebound coefficient possible? Explain your answer.

Task 11. Define a new distance function dist2(h,r,n0,n1) by modifying your distance function from Task 4 so that it computes the total vertical distance traveled from the apex of bounce n0 to the apex of bounce n1. Evaluate dist2(hin,rin,0,2) and dist2(hin,rin,2,5). Assuming h= hin and we are going to perform 100 total bounces, what rebound coefficient r produces a distance for the first 2 bounces equal to the distance for all of the remaining bounces thereafter? Hint: Plot dist2 as a function of r with n0=1 & n1=2, and dist2 as a function of r with n0=2 and n1=100 on the same graph, for r from .1 to 1. This should give you a good initial guess for r to use in fzero.

Turn In Your Project

Use the DropFolders method to turn in your project. Copy your entire uuuuup4 directory into the proper dropfolder. Remember that projects up to 24 hours late receive a 25 point penalty; projects later than 24 hours are not accepted.

No matter what problems you are having with your project, always turn in whatever you have completed by the due date. If you have strange last minute problems, you can send email to your professor explaining it. But don't miss the due date! Turn in whatever you have by the due date!!!

```
I have just posted on the project 4 web folder a "template" for vheight (called
zvheightbjb.m) that finds the vertical height using an explicit loop.  You
should be able to use it as a model for both your vheight function & your
vvheight function.
You can also use explicit loops to build the table for Task 5:
nb=[1:nin];  %first column indicates number of bounces
h4=[5:5:20]; %the heights we want to consider
for i=1:4
    h=i*5;  %generates 4 starting heights
    for n=1:nin
    htab(i,n)=distance(h,rin,n);  %for fixed height, find distance after n
bounces
end
end
You can also use an explicit loop in Task 7:
for i=0:nin
    vhtab(i+1)=vheight(hin,i);  %note subscripting must begin at 1
end
And in Task 9:
for i=0:12
    vvhtab(i+1)=vvheight(20,i,@rc); %remember subscripting must begin at 1
end

Good luck!
bjb
```

```
% Task 4:
    disp ('-----------------------------------------------------------------------
--');
    disp ('Task 4: ');



% Task 5:
    disp ('-----------------------------------------------------------------------
--');
    disp ('Task 5: ');
```

```
% Task 6:
    disp ('-----------------------------------------------------------------------
--');
    disp ('Task 6: ');


% Task 7:
    disp ('-----------------------------------------------------------------------
--');
    disp ('Task 7: ');


% Task 8:
    disp ('-----------------------------------------------------------------------
--');
    disp ('Task 8: ');


% Task 9:
    disp ('-----------------------------------------------------------------------
--');
    disp ('Task 9: ');


% Task 10:
    disp ('-----------------------------------------------------------------------
--');
    disp ('Task 10: ');
```