Simulation Project

Scott Kobos

Introduction/Purpose

The most commonly used example throughout the computational methods course for physics is the kinematic equations; calculating range, time of flight, maximum height for example. While these equations are quality approximations and generally accepted as accurate, they do not account for every factor of projectile motion. One of the key missing components of an objects flight through air is drag, or air resistance. Drag is defined as the resistance due to air that opposes the flight of an object, that increases as the speed of the object decreases. With that definition in mind, an elementary hypothesis can be stated. It can be expected that the trajectory of a projectile, in this case, a batted ball, when calculated without drag, will fly farther than if drag was accounted for. We can also expect to see a greater difference in trajectory when the ball is launched, or hit, with a greater velocity. When implementing the program, we can see that these hypotheses hold true. The purpose of this simulation project is the write a procedural program that generates an animation of the motion of a batted ball both with and without drag, using the initial velocity and launch angle as inputs.

Theory

The key concepts applied to this simulation are the basic kinematic equations, without drag, for x-position and y-position as shown below:

$$x = x0 + v\,\text{Cos}[\theta\,\text{Degree}]\,t;$$

$$y = y0 + v\,\text{Sin}[\theta\,\text{Degree}]\,t - \frac{1}{2}\,g\,t^2$$

The variables in the first equation, for x-position, are standard, x0 is the initial x position, in this simulation the initial x position is always zero. Following the initial x-position is vCos[θ] which is the the x-component of the initial velocity. In the equation shown above you see the angle, θ, specified in degrees. This is done because the Mathematica program would otherwise assume the angle is in radians, which is not commonly used for projectile motion. The final variable in the equation is t, which is the time of flight. The second equation for y-position uses the same variable, t, for time. It also uses y0 for initial y-position, and the y-component of the initial velocity, given by vSin[θ]. The only difference is the inclusion of g, which is gravity which is 9.8 m/s^2.

The next two equations are the more advanced projectile motion equations that include drag and are as follows:

$$x = \left(\left(\frac{v\,\mathrm{Cos}[\theta\ \mathrm{Degree}]}{k}\right)\right)\left(1 - e^{-k\,t}\right)$$

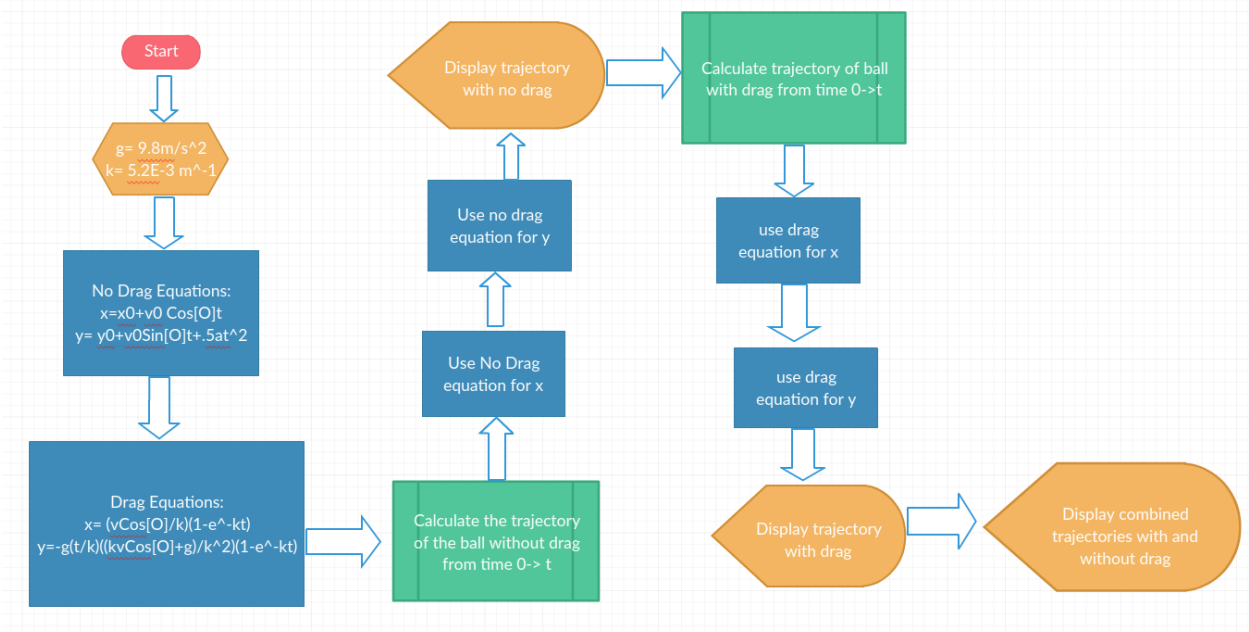
$$y = -g\left(\frac{t}{k}\right) + \frac{((k\ (v\,\mathrm{Sin}[\theta\ \mathrm{Degree}]\ ))\ + g)}{k^2}\left(1 - e^{-k\,t}\right)$$

These equations come from the fifth edition of the Thornton and Marion Classical Dynamics textbook and follow the same general idea as the basic kinematic equations but have the added component of drag. The variable, k, is for the drag coefficient, which is $5.2 \times 10^{-3} m^{-1}$. We once again have the initial x-component in the first equation, along with the initial y-component in the second equation. The most noticeable difference in the equations is the inclusion of the (1-exp(-kt)), which is part of the factoring in of drag.

Methodology

Before the implementation of the kinematic equations with and without drag could begin, a flowchart and pseudocode were written to plan out the code that would produce the required animations. The flowchart and pseudocode are as follows:

Flowchart:



Pseudocode:

START
READ g= 9.8 m/s^2, k= $5.2*10^{-3}$
SET $y=y_i+v\sin\theta t+1/(2\ at^2)$
SET $x=x_i+v\cos\theta t$
SET $y= -g\ (t/k)\ (k\ v\sin\theta + g)/k^2\ (1-E^{-kt})$
SET $x=v\cos\theta/k\ (1-E^{-kt})$
CALCULATE trajectory of ball without drag
IF y<0
PRINT {x,0}
END IF
PRINT trajectory of ball without drag
CALCULATE trajectory of ball with drag
IF y<0
PRINT {x,0}
END IF
PRINT trajectory of ball with drag
PRINT combined graph of trajectory of ball with and without drag
END

The first step in the implementation of this planning was to define a function for the motion without drag, as it would be easier to check if my code was producing the correct result. I named this function "nodrag", with the arguments of initial x-position, initial y-position, initial velocity, and time. This function would be executed with Mathematica's Module function. Within the Module, I defined the local variables x and y, and designated g, the value of 9.8. Following the definition of my local variables, I defined the equations to be utilized as the x and y without drag equations. Next was perhaps the most important inclusion in the Module; and If statement that prevented the code from printing a negative value for y, as a batted ball, and most projectiles, will not continue to fall through the ground. The If statement ordered that if the output of the y equation was less than 0, that the Module produced an answer that included the x-position, with a y-position of zero, as if the ball had hit the ground and started rolling. With the "nodrag" function complete, the next step was to produce a list of coordinates to be plotted as the trajectory of the ball without drag. To do this, the Module function was once again used, this function being named "zerodrag." "Zerodrag" was written to produce data points at different increments of time, using the "nodrag" function, so that the position of the ball would be recorded and stored in a list every tenth of a second, for a number, Np, of seconds. With both the "nodrag" and "zerodrag" functions complete and operating as expected, the next step was using the ListPlot function in Mathematica to plot the coordinates produced by the "zerodrag" function. The plot produced should have, and did, appear parabolic. Following the successful plotting of the "zerodrag" function, the final part of the task could be completed, the animation of the function with input arguments of initial velocity and launch angle. To do this, the Manipulate function of Mathematica was used. The Manipulate function allows for the manipulation of desired variables, which would provide an animation of the trajectory as either variable changes. The input arguments of initial velocity and launch angle were limited to an initial velocity of 50m/s, or 111mph (an elite major league exit velocity), and 90 degrees, as it is desired to keep the direction of the batted ball in the positive x direction. The exact same process was carried out for the trajectory of a ball with drag, with the functions "yesdrag" and "morethanzerodrag." The only additional step was including both the trajectories with and without drag in the same Manipulate function to see a direct comparison of the two trajectories. An important note is that the neither the function for drag nor the function with drag include an initial y-position. Choosing to start the initial position of the batted ball at {0,0} was done because of the natural variances in initial y-position that come from batter height, pitch location, and pitch type, which were not included in the scope of this simulation.

Analysis

To begin to analyze this simulation, it must be considered that no one, specific calculation is to be produced by a program specifically designed to give a range of values given two different argument variables. While the accuracy of one specific calculation is not to be computed, the validity of the program can be verified by observation and comparison to an established model. The first method of verifying the accuracy of this program is to see that the graph for the trajectory with drag is shorter than the trajectory of the ball without drag. It makes sense that the ball with drag flies a shorter distance because drag is a combatant force (opposes the flight of the ball). The established model to verify the validity of the simulation comes from the Wolfram Demonstrations Project, written by Enrique Zeleny (link to the WDP will be included at the conclusion of this analysis section). The Projectile with Air Drag function in the WDP is similar to the animation produced by this simulation, only with additional argument variables. One aspect of this WDP code to note is that it takes terminal velocity into account. The simulation written for this project does not include terminal velocity, so when comparing the animations of the WDP by Zeleny, and the simulation produced within the scope of this project, you must maximize the limit of the terminal velocity. It is also necessary to note that the input angle is in radians, and the procedural code in this project is written in degrees. When accounting for slight differences in scope of the code and units, you see that the simulations are identical, verifying and validating the simulation produced in this project code.

Projectile with Air Drag WDP: http://demonstrations.wolfram.com/ProjectileWithAirDrag/

Conclusion

The purpose of this simulation was to provide an animation for the trajectory of a batted ball with and without drag, with the manipulatable arguments being the initial velocity and launch angle. It was hypothesized that the flight of a batted ball with drag would be shorter than that of a batted ball without drag, due to the nature of the combatant drag force, and that the difference in flight would be greater at greater velocities. As seen in both the simulation produced by this project, and the simulation performed and published in the Wolfram Demonstrations Project, both hypotheses are proven to be correct. To put these results in context, it shows how the elementary kinematic equations are quite accurate for projectile motion, even without accounting for every factor involved in the flight of an object, which is why at lower levels of physics, the elementary equations are sufficient in learning the concepts of projectile motion.