

```
1 %Aidan Chin
2 %Project 2
3 %11/14/23
4
5
6 % initialize
7
8 clear
9 clc
10
11 % Constants
12
13 g = 32.2; % acceleration of gravity in ft/s^2
14 theta = deg2rad(28); % launch angle in radians
15 vE = 116 * 5280 / 3600; % exit velocity in ft/s (converted from mph)
16 aToF = 5.3; % time of flight in seconds
17 baseballmass = 0.145; % mass of a baseball in kg
18 rho_air = .00238; % air density in slugs/ft^3
19 C = input("Enter drag coefficient (e.g. .38): ");
20
21 baseballr = .06035; %radius of baseball in ft
22 ballarea = pi * baseballr^2; %cross sectional area of baseball
23
24 % Initial conditions
25 x0 = 0; y0 = 0; % initial position
26 vx0 = vE * cos(theta); % initial x-component of velocity
27 vy0 = vE * sin(theta); % initial y-component of velocity
28
29 % Time settings
30 dt = 0.01; % time step
31 tmax = aToF; % maximum time
32 tval = 0:dt:tmax; % array of time values
33
34 % Initialize arrays to store results
35 %no drag
36 x = zeros(size(tval));
37 y = zeros(size(tval));
38 %drag
39 xdrag = zeros(size(tval));
40 ydrag = zeros(size(tval));
41
42 % Initial conditions
43 %no drag
44 x(1) = x0;
45 y(1) = y0;
46 vx = vx0;
47 vy = vy0;
48 %drag
49 xdrag(1) = x0;
50 ydrag(1) = y0;
51 vxdrag = vx0;
```

```
52 vydrag = vy0;
53
54 % Numerical computation using Euler's method with and without drag
55 for i = 2:length(tval)
56     % Acceleration components
57     % no drag
58     ax = 0; % no acceleration in x-direction
59     ay = -g; % acceleration due to gravity in y-direction
60     % drag
61     vdrag = sqrt(vxdrag^2 + vydrag^2);
62     axdrag = -.5 * C * rho_air * baseballa * vdrag * vxdrag / baseballmass;
63     aydrag = -g - .5 * C * rho_air * baseballa * vdrag * vydrag / baseballmass;
64
65     % Update velocities and positions using Euler's method
66     %no drag
67     vx = vx + ax * dt;
68     vy = vy + ay * dt;
69     x(i) = x(i - 1) + vx * dt;
70     y(i) = y(i - 1) + vy * dt;
71     %drag
72     vxdrag = vxdrag + axdrag * dt;
73     vydrag = vydrag + aydrag * dt;
74     xdrag(i) = xdrag(i - 1) + vxdrag * dt;
75     ydrag(i) = ydrag(i - 1) + vydrag * dt;
76
77     % Check for the end of the trajectory
78     if y(i) < 0
79         break;
80     end
81 end
82
83 % convert units to feet
84 x = x * 3.28084;
85 y = y * 3.28084;
86 xdrag = xdrag * 3.28084;
87 ydrag = ydrag * 3.28084;
88
89 %check function
90
91 check_x = abs(x(end) - (vx0 * tval(end)));
92 check_y = abs(y(end) - (y0 + vy0 * tval(end) - 0.5 * g * tval(end)^2));
93
94 disp(['max difference in x (with drag): ', num2str(check_x)])
95 disp(['max difference in y (with drag): ', num2str(check_y)])
96
97 % Plot trajectories
98 figure;
99 plot(x, y, '--', 'LineWidth', 1.5, 'DisplayName', 'no drag');
100 hold on;
101 plot(xdrag, ydrag, '-', 'LineWidth', 1.5, 'DisplayName', 'drag');
102 title(['Aidan Chin | ECE202 Project 2 | 12/07/23 |' ...
```

```
103     ' Baseball Trajectory with and without Air Resistance']]);
104 xlabel('Distance (feet)');
105 ylabel('Height (feet)');
106 legend('Location', 'Best');
107 grid on;
108 ax = gca;
109 ax.GridAlpha = .4;
110 ax.MinorGridAlpha = .5;
111
```

```
Enter drag coefficient (e.g. .38): .38  
max difference in x (with drag): 796.1597  
max difference in y (with drag): 28.9234  
>>
```

Aidan Chin | ECE202 Project 2 | 12/07/23 | Baseball Trajectory with and without Air Resistance

