

```
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 baseballa = 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     % Check for the end of the trajectory
77     if y(i) < 0
78         %get info
79         flighttime = tval(i);
80         ymax = max(ydrag);
81         range = xdrag(i);
82         Vfinal = sqrt(vxdrag^2 + vydrag^2);
83         break;
84     end
85 end
86
87 % convert units to feet
88 x = x * 3.28084;
89 y = y * 3.28084;
90 xdrag = xdrag * 3.28084;
91 ydrag = ydrag * 3.28084;
92
93 %check function
94
95 check_x = abs(x(end) - (vx0 * tval(end)));
96 check_y = abs(y(end) - (y0 + vy0 * tval(end) - 0.5 * g * tval(end)^2));
97
98 disp(['max difference in x (with drag): ', num2str(check_x)])
99 disp(['max difference in y (with drag): ', num2str(check_y)])
100
101 % final stats
102 disp(['Time of Flight: ', num2str(flighttime), ' seconds']);
```

```
103
104 %flight time is identical
105
106 disp(['Maximum Height: ', num2str(ymax), ' feet']);
107 disp(['Range: ', num2str(range), ' feet']);
108 disp(['Final Speed: ', num2str(Vfinal), ' ft/s']);
109
110
111 % Plot trajectories
112 figure;
113 plot(x, y, '--','LineWidth', 1.5, 'DisplayName', 'no drag');
114 hold on;
115 plot(xdrag, ydrag, '-', 'LineWidth', 1.5, 'DisplayName', 'drag');
116 title(['Aidan Chin | ECE202 Project 2 | 12/07/23 | ' ...
117       ' Baseball Trajectory with and without Air Resistance']);
118 xlabel('Distance (feet)');
119 ylabel('Height (feet)');
120 legend('no drag', ['drag (C = ' num2str(C) ')'], 'Location', 'Best');
121 grid on;
122 ax = gca;
123 ax.GridAlpha = .4;
124 ax.MinorGridAlpha = .5;
125
126 % save to excel
127
128 % create a matrix of data
129 dataMatrix = [tval',xdrag', ydrag'];
130
131 % export the data to a CSV
132 filename = 'baseball_trajectory_data.csv';
133 writematrix(dataMatrix, filename);
134
135 % display filename
136 disp(['Data exported to ' filename]);
137
138
```