```
%Aidan Chin
%Project 2
%11/14/23


% initialize

clear
clc

% Constants

g = 32.2; % acceleration of gravity in ft/s^2
theta = deg2rad(28); % launch angle in radians
vE = 116 * 5280 / 3600; % exit velocity in ft/s (converted from mph)
aToF = 5.3; % time of flight in seconds
baseballmass = 0.145; % mass of a baseball in kg
rho_air = .00238; % air density in slugs/ft^3
C = input("Enter drag coefficient (e.g. .38): ");

baseballr = .06035; %radius of baseball in ft
baseballa = pi * baseballr^2; %cross sectional area of baseball

% Initial conditions
x0 = 0; y0 = 0; % initial position
vx0 = vE * cos(theta); % initial x-component of velocity
vy0 = vE * sin(theta); % initial y-component of velocity

% Time settings
dt = 0.01; % time step
tmax = aToF; % maximum time
tval = 0:dt:tmax; % array of time values

% Initialize arrays to store results
%no drag
x = zeros(size(tval));
y = zeros(size(tval));
%drag
xdrag = zeros(size(tval));
ydrag = zeros(size(tval));

% Initial conditions
%no drag
x(1) = x0;
y(1) = y0;
vx = vx0;
vy = vy0;
%drag
xdrag(1) = x0;
ydrag(1) = y0;
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
112 % Compute percent errors
113 known_range = 463; % known range from mlb.statcast
114 percent_error_range = ((range - known_range) / known_range) * 100;
115
116 known_ymax = 100; % known max height from mlb.statcast
117 percent_error_ymax = ((ymax - known_ymax) / known_ymax) * 100;
118
119 disp(['Percent Error in Range: ', num2str(percent_error_range), '%']);
120 disp(['Percent Error in Max Height: ', num2str(percent_error_ymax), '%']);
121
122
123 %Calculate initial kinetic energy
124 initial_speed = vE; % Assuming the initial speed is the
125 % exit velocity
126 KE_initial = 0.5 * baseballmass * initial_speed^2;
127
128 % Calculate final kinetic energy (after the FOR loop)
129 final_speed = sqrt(vxdrag(end)^2 + vydrag(end)^2);
130 KE_final = 0.5 * baseballmass * final_speed^2;
131
132 % Calculate energy lost
133 delta_KE = KE_initial - KE_final;
134
135 % Display the results
136 disp(['Initial Kinetic Energy: ', num2str(KE_initial), ' J']);
137 disp(['Final Kinetic Energy: ', num2str(KE_final), ' J']);
138 disp(['Energy Lost: ', num2str(delta_KE), ' J']);
139
140 % Plot trajectories
141 figure;
142 plot(x, y, '--','LineWidth', 1.5, 'DisplayName', 'no drag');
143 hold on;
144 plot(xdrag, ydrag, '-', 'LineWidth', 1.5, 'DisplayName', 'drag');
145 title(['Aidan Chin | ECE202 Project 2 | 12/07/23 |' ...
146     ' Baseball Trajectory with and  without Air Resistance']);
147 xlabel('Distance (feet)');
148 ylabel('Height (feet)');
149 legend('no drag', ['drag (C = ' num2str(C) ')'], 'Location', 'Best');
150 grid on;
151 ax = gca;
152 ax.GridAlpha = .4;
153 ax.MinorGridAlpha = .5;
```

```
154
155 % save to excel
156
157 % create a matrix of data
158 dataMatrix = [tval',xdrag', ydrag'];
159
160 % export the data to a CSV
161 filename = 'baseball_trajectory_data.csv';
162 writematrix(dataMatrix, filename);
163
164 % display filename
165 disp(['Data exported to ' filename]);
166
167
```

```
Enter drag coefficient (e.g. .38): 19
max difference in x (with drag): 796.1597
max difference in y (with drag): 28.9234
Time of Flight: 4.96 seconds
Maximum Height: 69.7537 feet
Range: 462.8212 feet
Final Speed: 97.2301 ft/s
Percent Error in Range: -0.038615%
Percent Error in Max Height: -30.2463%
Initial Kinetic Energy: 2098.538 J
Final Kinetic Energy: 685.3926 J
Energy Lost: 1413.1454 J
Data exported to baseball_trajectory_data.csv
>>
```