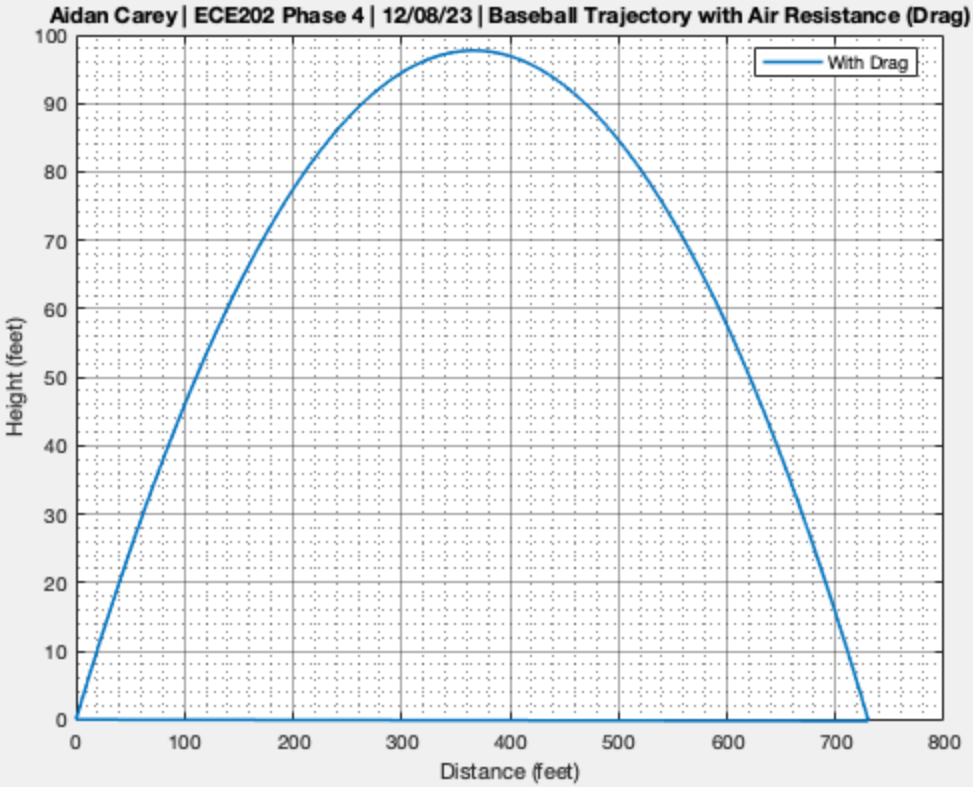
```
%Aidan Carey
%12/08/23
% project 2 phase 4

% description - in this phase, we will be doing the same as phase 3,
% calculating the time of flight, max height, and range of the ball in
% MATLAB versus excel.


% Constants
g = 32.2;  % acceleration due to gravity in ft/s^2
launch_angle = deg2rad(28);  % launch angle in radians
exit_velocity = 116 * 5280 / 3600;  % exit velocity in ft/s
% (converted from mph)
analytic_time_of_flight = 5.3;  % time of flight in seconds

% Additional constants
mass_of_baseball = 0.145;  % mass of a baseball in kg
rho_air = 0.00238;  % air density in slugs/ft^3

% User input for drag coefficient
C = 0.38;

% Cross-sectional area of a baseball (approximated as a sphere)
radius_of_baseball = 0.06035;  % in feet
A = pi * radius_of_baseball^2;  % cross-sectional area

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

% Time settings
dt = 0.01;  % time step
t_max = analytic_time_of_flight;  % maximum time
t_values = 0:dt:t_max;

% Initialize arrays to store results
x_values_with_drag = zeros(size(t_values));
y_values_with_drag = zeros(size(t_values));

% Initial conditions
x_values_with_drag(1) = x0;
y_values_with_drag(1) = y0;
vx_with_drag = vx0;
vy_with_drag = vy0;

% Initialize variables to store information
time_of_flight = 0;
max_height = 0;
range_ft = 0;
final_speed = 0;

% Numerical computation using Euler's method with drag
for i = 2:length(t_values)
    % With drag
    v_with_drag = sqrt(vx_with_drag^2 + vy_with_drag^2);
    ax_with_drag = -0.5 * C * rho_air * A * v_with_drag * vx_with_drag ...
        / mass_of_baseball;
```

```
    ay_with_drag = -g - 0.5 * C * rho_air * A * v_with_drag *...
        vy_with_drag / mass_of_baseball;

    % Update velocities and positions using Euler's method
    vx_with_drag = vx_with_drag + ax_with_drag * dt;
    vy_with_drag = vy_with_drag + ay_with_drag * dt;
    x_values_with_drag(i) = x_values_with_drag(i - 1) + vx_with_drag * dt;
    y_values_with_drag(i) = y_values_with_drag(i - 1) + vy_with_drag * dt;

    % Check for the end of the trajectory
    if y_values_with_drag(i) < 0
        % Grab information when the ball hits the ground
        time_of_flight = t_values(i);
        max_height = max(y_values_with_drag);
        range_ft = x_values_with_drag(i);
        final_speed = sqrt(vx_with_drag^2 + vy_with_drag^2);
        break;
    end
end


% Display results
disp(['Time of Flight: ', num2str(time_of_flight), ' seconds']);
disp(['Maximum Height: ', num2str(max_height), ' feet']);
disp(['Range: ', num2str(range_ft), ' feet']);
disp(['Final Speed: ', num2str(final_speed), ' ft/s']);

% Energy calculations
initial_kinetic_energy = 0.5 * mass_of_baseball * (exit_velocity)^2;
final_kinetic_energy = 0.5 * mass_of_baseball * final_speed^2;
energy_lost_J = initial_kinetic_energy - final_kinetic_energy;

disp(['Energy Lost due to Air Resistance: ', num2str(energy_lost_J), ...
    ' joules']);

% Plot trajectory with drag
figure;
plot(x_values_with_drag, y_values_with_drag, '-', 'LineWidth', 1.5, ...
    'DisplayName', 'With Drag');
title(['Aidan Carey | ECE202 Phase 4 | 12/08/23 | Baseball Trajectory ' ...
    'with Air Resistance (Drag)']);
xlabel('Distance (feet)');
ylabel('Height (feet)');
legend('Location', 'Best');
grid on;
grid minor;  % Add minor grid lines
ax = gca;
ax.GridAlpha = 0.4;  % Adjust grid alpha for visibility
ax.MinorGridAlpha = 0.5;  % Adjust minor grid alpha for visibility
```

```
>> project2phase4
Time of Flight: 4.93 seconds
Maximum Height: 97.7248 feet
Range: 730.5084 feet
Final Speed: 166.1077 ft/s
Energy Lost due to Air Resistance: 98.1356 joules
>>
```


Aidan Carey | ECE202 Phase 4 | 12/08/23 | Baseball Trajectory with Air Resistance (Drag)
With Drag
Height (feet)
Distance (feet)
0
10
20
30
40
50
60
70
80
90
100
0
100
200
300
400
500
600
700
800