ASEN 2012 Project 2 - Group Portion varied Cd and initial launch angle simulation

Table of Contents

By: Ian Faber and Jack Walsh
SID: 108577813, 10040714

Started: 12/5/21, 13:30
Finished: 12/7/21, 16:38

Runs a varied bottle rocket simulation subject to a set of initial parameters through 3 different phases of flight: water thrust, air thrust, and ballistic flight. This specific script utilizes a custom ODE45 EOM function and varies the rocket's coefficient of drag and intial launch angle, then plots them all together to see trends in trajectory and thrust.

Setup

```
% Housekeeping
clc; clear; close all;

% Get all constants from the const structure
const = getConst();

target = 85;

cDragSim = struct('Cdrag',0,'rocketX',[],'rocketZ',[]);
initialAngleSim = struct('initialAngle',0,'rocketX',[],'rocketZ',[]);
```

Vary Cd

```
Cds = 0.2:0.1:0.8;

for k = 1:length(Cds)
```

```
    const.Cdrag = Cds(k);

    fprintf("Simulation with a Cd of %.3f\n", const.Cdrag);

    % Store simulated Cd
    cDragSim(k).Cdrag = const.Cdrag;

    % Difference of Bottle and initial water volumes
    VAirInit = const.Vbottle - const.VWaterInit;

    % Need absolute pressure of air, also convert psi to Pa
    PAirInit = (const.PGageInit+const.PAmb)*6894.76;

    % Calculate rho w/ Ideal Gas EOS
    rhoAirInit = (PAirInit)/(const.R*const.TAirInit);

    % Calculate initial masses
    mAirInit = rhoAirInit*VAirInit;
    mWaterInit = const.rhoWater*const.VWaterInit;
    mRocketInit = const.mBottle + mAirInit + mWaterInit;

    % Calculate initial x and z velocities
    vx0 = const.vInit*cosd(const.thetaInit);
    vz0 = const.vInit*sind(const.thetaInit);

    % Format the initial conditions vector, and by extension variables
 to
    % integrate
    X0 = [const.xInit; const.zInit; vx0; vz0; mRocketInit; mAirInit;
 VAirInit];

    % Define events worthy of stopping integration
    options = odeset('Events',@phase);
```

```
Simulation with a Cd of 0.200

Simulation with a Cd of 0.300

Simulation with a Cd of 0.400

Simulation with a Cd of 0.500

Simulation with a Cd of 0.600

Simulation with a Cd of 0.700

Simulation with a Cd of 0.800
```

Simulation

```
    % Integrate! Solves for the trajectory of the rocket by
 integrating the
    % variables in X0 over tspan according to the derivative
 information
```

```
    % contained in rocketEOM. Also stops integration according to
"options," a
    % predefined set of stopping conditions
    [time, state, timePhases, ~, ~] =
ode45(@(t,state)rocketEOM(t,state,const), const.tspan, X0, options);

    % Extract intermediate variables from rocketEOM for debugging,
particularly
    % weight, drag, thrust, and air pressure. Found this approach on
the MATLAB
    % forums.
    [~,gravCell, dragCell, thrustCell, PairCell] =
cellfun(@(t,state)rocketEOM(t,state.',const), num2cell(time),
num2cell(state,2), 'uni', 0);

    %Allocate space for intermediate variables
    gravity = zeros(length(time),1);
    drag = zeros(length(time),1);
    thrust = zeros(length(time),1);
    Pair = zeros(length(time),1);

    % Extract intermediate variables from their cells
    for i = 1:length(time)
        gravity(i) = norm(gravCell{i});
        drag(i) = norm(dragCell{i});
        thrust(i) = norm(thrustCell{i});
        Pair(i) = norm(PairCell{i});
    end
```

Extraction

```
    % Extract variables of interest
    rocketX = state(:,1);
    rocketZ = state(:,2);
    rocketVx = state(:,3);
    rocketVz = state(:,4);
    rocketM = state(:,5);
    rocketMair = state(:,6);
    rocketV = state(:,7);

    % Find maximum values of interest
    maxRange = max(rocketX)
    maxHeight = max(rocketZ)
    maxVx = max(rocketVx)
    maxVy = max(rocketVz)
    maxThrust = max(thrust)

    % Update structure entry
    cDragSim(k).time = time;
    cDragSim(k).timePhases = timePhases;
    cDragSim(k).rocketX = rocketX;
    cDragSim(k).rocketZ = rocketZ;
    cDragSim(k).thrust = thrust;
```

```
maxRange =

   84.5941

maxHeight =

   21.2554

maxVx =

   26.2184

maxVy =

   21.0934

maxThrust =

  191.0459


maxRange =

   74.2976

maxHeight =

   19.6761

maxVx =

   26.0314

maxVy =

   20.9326

maxThrust =

  191.0459


maxRange =
```

```
    66.4193

maxHeight =

    18.3493

maxVx =

    25.8125

maxVy =

    20.7559

maxThrust =

   191.0459


maxRange =

    60.2658

maxHeight =

    17.2294

maxVx =

    25.6057

maxVy =

    20.5830

maxThrust =

   191.0459


maxRange =

    55.2425
```

```
maxHeight =

   16.2586


maxVx =

   25.3982


maxVy =

   20.4139


maxThrust =

  191.0459


maxRange =

   51.1411


maxHeight =

   15.4329


maxVx =

   25.1953


maxVy =

   20.2484


maxThrust =

  191.0459


maxRange =

   47.6927
```

```
maxHeight =

   14.7077


maxVx =

   24.9970


maxVy =

   20.0866


maxThrust =

  191.0459


end
```

Vary initial heading

```
headings = 5:5:85;

const.Cdrag = 0.5; % Reset Cd

for k = 1:length(headings)

    const.thetaInit = headings(k);

    fprintf("Simulation with an initial heading of %.2f degrees \n",
 const.thetaInit);

    % Store simulated initial angle
    initialAngleSim(k).initialAngle = const.thetaInit;

    % Difference of Bottle and initial water volumes
    VAirInit = const.Vbottle - const.VWaterInit;

    % Need absolute pressure of air, also convert psi to Pa
    PAirInit = (const.PGageInit+const.PAmb)*6894.76;

    % Calculate rho w/ Ideal Gas EOS
    rhoAirInit = (PAirInit)/(const.R*const.TAirInit);

    % Calculate initial masses
    mAirInit = rhoAirInit*VAirInit;
    mWaterInit = const.rhoWater*const.VWaterInit;
    mRocketInit = const.mBottle + mAirInit + mWaterInit;

    % Calculate initial x and z velocities
    vx0 = const.vInit*cosd(const.thetaInit);
```

```
    vz0 = const.vInit*sind(const.thetaInit);

    % Format the initial conditions vector, and by extension variables
to
    % integrate
    X0 = [const.xInit; const.zInit; vx0; vz0; mRocketInit; mAirInit;
VAirInit];

    % Define events worthy of stopping integration
    options = odeset('Events',@phase);
```

Simulation with an initial heading of 5.00 degrees

Simulation with an initial heading of 10.00 degrees

Simulation with an initial heading of 15.00 degrees

Simulation with an initial heading of 20.00 degrees

Simulation with an initial heading of 25.00 degrees

Simulation with an initial heading of 30.00 degrees

Simulation with an initial heading of 35.00 degrees

Simulation with an initial heading of 40.00 degrees

Simulation with an initial heading of 45.00 degrees

Simulation with an initial heading of 50.00 degrees

Simulation with an initial heading of 55.00 degrees

Simulation with an initial heading of 60.00 degrees

Simulation with an initial heading of 65.00 degrees

Simulation with an initial heading of 70.00 degrees

Simulation with an initial heading of 75.00 degrees

Simulation with an initial heading of 80.00 degrees

Simulation with an initial heading of 85.00 degrees

Simulation

```
    % Integrate! Solves for the trajectory of the rocket by
integrating the
    % variables in X0 over tspan according to the derivative
information
    % contained in rocketEOM. Also stops integration according to
"options," a
    % predefined set of stopping conditions
    [time, state, timePhases, ~, ~] =
ode45(@(t,state)rocketEOM(t,state,const), const.tspan, X0, options);
```

```
    % Extract intermediate variables from rocketEOM for debugging,
particularly
    % weight, drag, thrust, and air pressure. Found this approach on
the MATLAB
    % forums.
    [~,gravCell, dragCell, thrustCell, PairCell] =
cellfun(@(t,state)rocketEOM(t,state.',const), num2cell(time),
num2cell(state,2), 'uni', 0);

    %Allocate space for intermediate variables
    gravity = zeros(length(time),1);
    drag = zeros(length(time),1);
    thrust = zeros(length(time),1);
    Pair = zeros(length(time),1);

    % Extract intermediate variables from their cells
    for i = 1:length(time)
        gravity(i) = norm(gravCell{i});
        drag(i) = norm(dragCell{i});
        thrust(i) = norm(thrustCell{i});
        Pair(i) = norm(PairCell{i});
    end
```

Extraction

```
    % Extract variables of interest
    rocketX = state(:,1);
    rocketZ = state(:,2);
    rocketVx = state(:,3);
    rocketVz = state(:,4);
    rocketM = state(:,5);
    rocketMair = state(:,6);
    rocketV = state(:,7);

    % Find maximum values of interest
    maxRange = max(rocketX)
    maxHeight = max(rocketZ)
    maxVx = max(rocketVx)
    maxVy = max(rocketVz)
    maxThrust = max(thrust)

    % Update structure entry
    initialAngleSim(k).time = time;
    initialAngleSim(k).timePhases = timePhases;
    initialAngleSim(k).rocketX = rocketX;
    initialAngleSim(k).rocketZ = rocketZ;
    initialAngleSim(k).thrust = thrust;
```

```
maxRange =

    6.1186
```

```
maxHeight =

    0.2619


maxVx =

   33.9884


maxVy =

    0.1930


maxThrust =

  191.0459


maxRange =

   17.6459


maxHeight =

    0.5513


maxVx =

   33.5487


maxVy =

    1.2978


maxThrust =

  191.0459


maxRange =

   31.2750


maxHeight =
```

```
    1.6150

maxVx =

   33.8271

maxVy =

    4.2108

maxThrust =

  191.0459


maxRange =

   40.9733

maxHeight =

    3.2757

maxVx =

   32.6997

maxVy =

    7.1240

maxThrust =

  191.0459


maxRange =

   48.1205

maxHeight =

    5.4599
```

```
maxVx =

   31.5515


maxVy =

    9.9878


maxThrust =

  191.0459


maxRange =

   53.5534


maxHeight =

    8.0438


maxVx =

   30.3145


maxVy =

   12.8211


maxThrust =

  191.0459


maxRange =

   57.2735


maxHeight =

   10.9839
```

```
maxVx =

   28.8933


maxVy =

   15.6005


maxThrust =

  191.0459


maxRange =

   60.2915


maxHeight =

   14.3567


maxVx =

   27.5405


maxVy =

   18.4191


maxThrust =

  191.0459


maxRange =

   60.2658


maxHeight =

   17.2294


maxVx =
```

```
   25.6057

maxVy =

   20.5830

maxThrust =

  191.0459


maxRange =

   59.2442

maxHeight =

   20.5144

maxVx =

   23.4752

maxVy =

   22.9134

maxThrust =

  191.0459


maxRange =

   56.3814

maxHeight =

   23.9165

maxVx =

   20.9219
```

```
maxVy =

   25.1361


maxThrust =

  191.0459


maxRange =

   51.0232

maxHeight =

   26.2558

maxVx =

   18.0698

maxVy =

   26.5881

maxThrust =

  191.0459


maxRange =

   46.6467

maxHeight =

   29.7155

maxVx =

   15.6246

maxVy =
```

```
   28.6024

maxThrust =

  191.0459


maxRange =

   38.4524

maxHeight =

   31.6699

maxVx =

   12.5615

maxVy =

   29.5976

maxThrust =

  191.0459


maxRange =

   29.7251

maxHeight =

   33.4412

maxVx =

    9.4984

maxVy =

   30.5030
```

```
maxThrust =

  191.0459


maxRange =

   20.6852

maxHeight =

   35.9347

maxVx =

    6.5330

maxVy =

   31.9262

maxThrust =

  191.0459


maxRange =

   10.3505

maxHeight =

   35.5280

maxVx =

    3.1916

maxVy =

   31.5049
```

```
maxThrust =

  191.0459


end
```

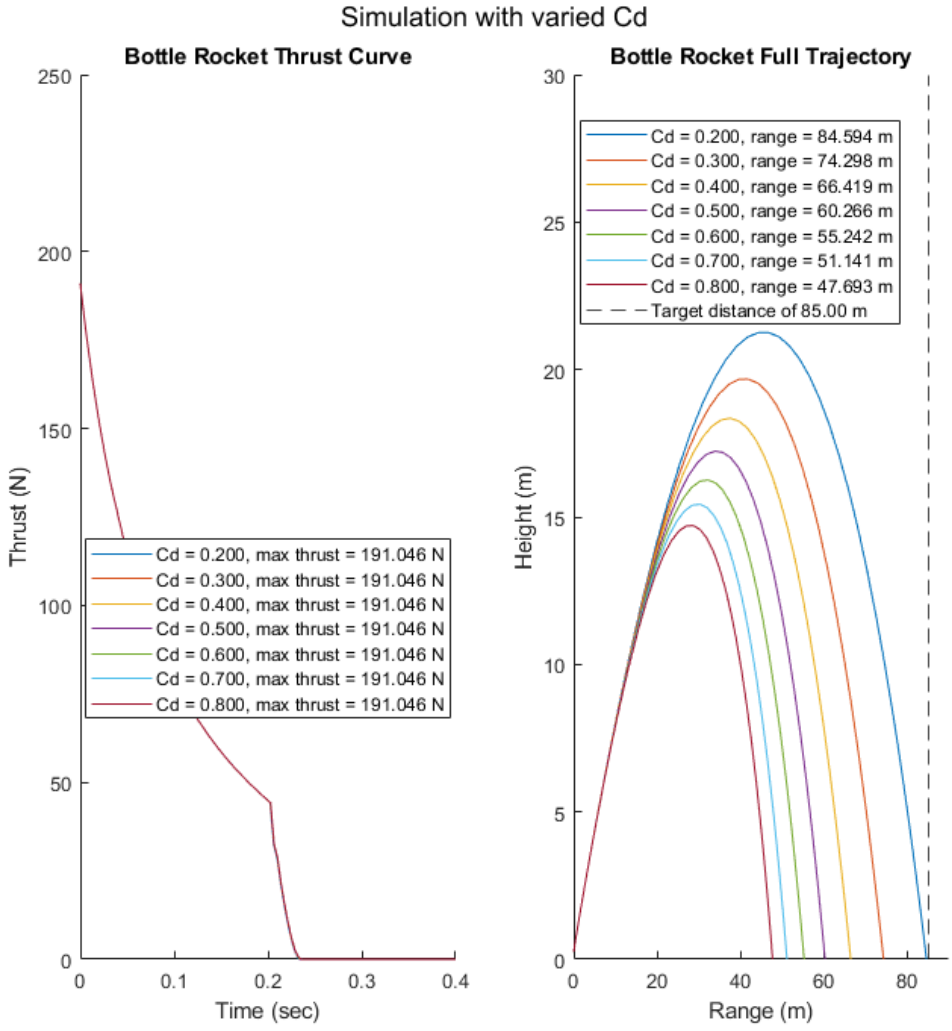
Plotting

Plot the trajectory/thrust with varied Cd

```
f = figure();
f.Position = [100 100 740 740]; % Start at (100, 100), end at (100 +
 740, 100 + 740)
sgtitle("Simulation with varied Cd");

% Thrust
subplot(1,2,1);

hold on

title("Bottle Rocket Thrust Curve")
label = strings(1,length(cDragSim) + 1);
label(length(cDragSim) + 1) = sprintf("Target distance of %.2f m",
 target);
plots = zeros(1,length(cDragSim));

for k = 1:length(cDragSim)
    plots(k) = plot(cDragSim(k).time, cDragSim(k).thrust);
    label(k) = sprintf("Cd = %.3f, max thrust = %.3f N",
 cDragSim(k).Cdrag, max(cDragSim(k).thrust));
end

xlim([0 0.4]);
ylim([0 250]);
xlabel('Time (sec)')
ylabel('Thrust (N)')
legend(plots, label, 'Location', 'best');

hold off

subplot(1,2,2);
hold on;

title("Bottle Rocket Full Trajectory")

label = strings(1,length(cDragSim) + 1);
label(length(cDragSim) + 1) = sprintf("Target distance of %.2f m",
 target);
plots = zeros(1,length(cDragSim));

% Trajectory
for k = 1:length(cDragSim)
```

```
    hold on;
    rocketX = cDragSim(k).rocketX;
    rocketZ = cDragSim(k).rocketZ;
    label(k) = sprintf("Cd = %.3f, range = %.3f m", cDragSim(k).Cdrag,
 max(rocketX));
    plots(k) = plot(rocketX, rocketZ);
end

plots(k+1) = xline(target, 'k--');

xlim([0, 90]);
ylim([0, 30]);
xlabel("Range (m)");
ylabel("Height (m)");
legend(plots, label, 'Location', 'best');
hold off;
```

Warning: Ignoring extra legend entries.

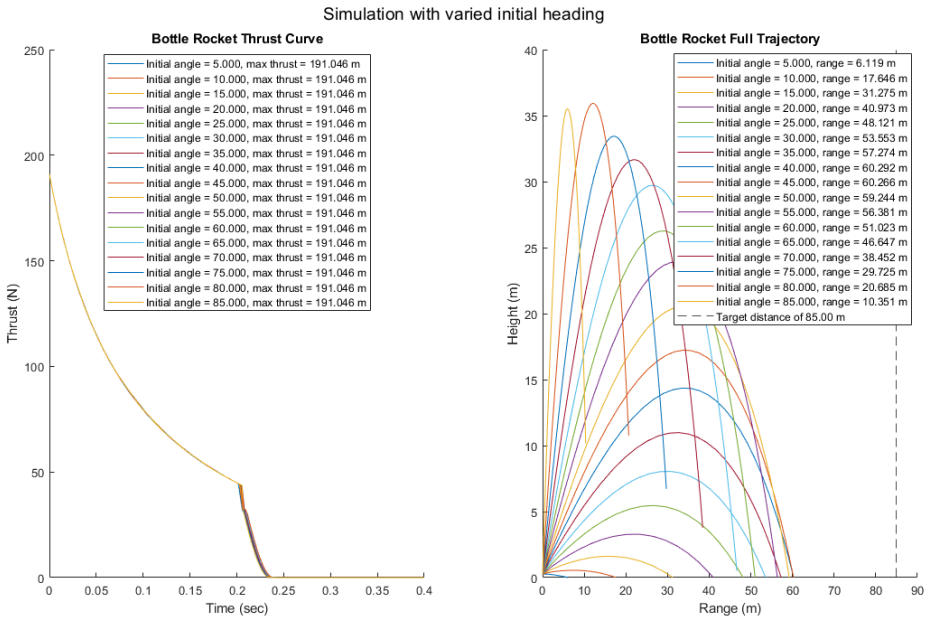


Plot the trajectory/thrust with varied initial heading

```
f = figure();
f.Position = [100 100 1240 740]; % Start at (940, 100) end at (940 +
 940, 100 + 940)
sgtitle("Simulation with varied initial heading");

label = strings(1,length(initialAngleSim) + 1);
label(length(initialAngleSim) + 1) = sprintf("Target distance of %.2f
 m", target);
plots = zeros(1,length(initialAngleSim));

% Thrust
subplot(1,2,1);

hold on

title("Bottle Rocket Thrust Curve");

for k = 1:length(initialAngleSim)
    plots(k) = plot(initialAngleSim(k).time,
 initialAngleSim(k).thrust);
    label(k) = sprintf("Initial angle = %.3f, max thrust = %.3f m",
 initialAngleSim(k).initialAngle, max(initialAngleSim(k).thrust));
end

xlabel("Time (sec)")
ylabel("Thrust (N)")
xlim([0 0.4])
ylim([0 250])
legend(plots, label, 'Location', 'best')

hold off

subplot(1,2,2);

hold on
title("Bottle Rocket Full Trajectory");

% Trajectory
for k = 1:length(initialAngleSim)
    hold on;
    rocketX = initialAngleSim(k).rocketX;
    rocketZ = initialAngleSim(k).rocketZ;
    time = initialAngleSim(k).time;
    label(k) = sprintf("Initial angle = %.3f, range = %.3f m",
 initialAngleSim(k).initialAngle, max(rocketX));
    plots(k) = plot(rocketX, rocketZ);
end

plots(k+1) = xline(target, 'k--');
```

```
xlim([0, 90]);
ylim([0, 40]);
xlabel("Range (m)");
ylabel("Height (m)");
legend(plots, label, 'Location', 'best');
hold off;
```

Warning: Ignoring extra legend entries.



Published with MATLAB® R2021a