ASEN 2012 Project 2 - Group Portion varied initial water volume and air pressure simulation

Table of Contents

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

Started: 12/5/21, 13:30
Finished: 12/7/21, 20:00

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 initial water volume and intial air pressure, 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;

volumeSim = struct('initialWaterVolume',0,'rocketX',[],'rocketZ',[]);
pressureSim = struct('initialGagePressure',0,'rocketX',[],'rocketZ',
[]);
```

Vary initial water volume

```
const.thetaInit = 45;
```

```
volumes = 0:0.0001:0.0013;

for k = 1:length(volumes)

    const.VWaterInit = volumes(k);

    fprintf("Simulation with an initial water volume of %.2f L \n",
 const.VWaterInit*1000);

    % Store simulated initial volume
    volumeSim(k).initialWaterVolume = const.VWaterInit;

    % 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 water volume of 0.00 L

Simulation with an initial water volume of 0.10 L

Simulation with an initial water volume of 0.20 L

Simulation with an initial water volume of 0.30 L

Simulation with an initial water volume of 0.40 L

Simulation with an initial water volume of 0.50 L

Simulation with an initial water volume of 0.60 L

Simulation with an initial water volume of 0.70 L
```

```
Simulation with an initial water volume of 0.80 L

Simulation with an initial water volume of 0.90 L

Simulation with an initial water volume of 1.00 L

Simulation with an initial water volume of 1.10 L

Simulation with an initial water volume of 1.20 L

Simulation with an initial water volume of 1.30 L
```

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
    volumeSim(k).time = time;
    volumeSim(k).timePhases = timePhases;
    volumeSim(k).rocketX = rocketX;
    volumeSim(k).rocketZ = rocketZ;
    volumeSim(k).thrust = thrust;


maxRange =

    6.5103


maxHeight =

    1.8112


maxVx =

    5.8105


maxVy =

    5.3059


maxThrust =

   38.2969



maxRange =

   31.4559


maxHeight =
```

```
    9.2513

maxVx =

   14.7720

maxVy =

   14.1168

maxThrust =

  191.0459


maxRange =

   46.7686

maxHeight =

   14.6842

maxVx =

   19.8871

maxVy =

   19.0595

maxThrust =

  191.0459


maxRange =

   55.7389

maxHeight =

   18.1657
```

```
maxVx =

   23.0964


maxVy =

   22.0893


maxThrust =

  191.0459


maxRange =

   60.5451


maxHeight =

   19.8746


maxVx =

   25.0047


maxVy =

   23.5569


maxThrust =

  191.0459


maxRange =

   62.9589


maxHeight =

   21.0002


maxVx =
```

```
   25.9302

maxVy =

   24.4920

maxThrust =

  191.0459


maxRange =

   64.9203

maxHeight =

   20.9932

maxVx =

   27.0781

maxVy =

   24.5129

maxThrust =

  191.0459


maxRange =

   65.2876

maxHeight =

   20.9258

maxVx =

   27.1842
```

```
maxVy =

   24.3235


maxThrust =

  191.0459


maxRange =

   66.4360

maxHeight =

   20.6342

maxVx =

   27.7855

maxVy =

   23.9633

maxThrust =

  191.0459


maxRange =

   63.1969

maxHeight =

   19.0899

maxVx =

   26.5057
```

```
maxVy =

   22.4111


maxThrust =

  191.0459


maxRange =

   60.2658

maxHeight =

   17.2294

maxVx =

   25.6057

maxVy =

   20.5830

maxThrust =

  191.0459


maxRange =

   54.0523

maxHeight =

   14.5398

maxVx =

   23.5543

maxVy =
```

```
   17.9510

maxThrust =

  191.0459


maxRange =

   47.1315

maxHeight =

   11.3477

maxVx =

   21.5278

maxVy =

   14.6600

maxThrust =

  191.0459


maxRange =

   37.4817

maxHeight =

    8.3251

maxVx =

   18.3087

maxVy =

   10.9469
```

```
maxThrust =

  191.0459


end
```

Vary initial water volume

```
const.VWaterInit = 0.001;
pressures = 0:10:80;

for k = 1:length(pressures)

    const.PGageInit = pressures(k);

    fprintf("Simulation with an initial air pressure of %.2f psi \n",
 const.PGageInit);

    % Store simulated initial volume
    pressureSim(k).initialGagePressure = const.PGageInit;

    % 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 air pressure of 0.00 psi

Simulation with an initial air pressure of 10.00 psi
```

```
Simulation with an initial air pressure of 20.00 psi

Simulation with an initial air pressure of 30.00 psi

Simulation with an initial air pressure of 40.00 psi

Simulation with an initial air pressure of 50.00 psi

Simulation with an initial air pressure of 60.00 psi

Simulation with an initial air pressure of 70.00 psi

Simulation with an initial air pressure of 80.00 psi
```

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] = 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
    pressureSim(k).time = time;
    pressureSim(k).timePhases = timePhases;
    pressureSim(k).rocketX = rocketX;
    pressureSim(k).rocketZ = rocketZ;
    pressureSim(k).thrust = thrust;
```

```
maxRange =

     0


maxHeight =

    0.2500


maxVx =

     0


maxVy =

     0


maxThrust =

     0



maxRange =

   1.3472 + 0.0011i


maxHeight =
```

```
    0.3904

maxVx =

   2.9273 + 0.0069i

maxVy =

  -2.7374 - 0.0076i

maxThrust =

   38.2092


maxRange =

   13.7911

maxHeight =

    2.4333

maxVx =

   10.7113

maxVy =

    4.2432

maxThrust =

   76.4183


maxRange =

   33.9383

maxHeight =

    7.6867
```

```
maxVx =

   17.1167


maxVy =

   11.2814


maxThrust =

  114.6275


maxRange =

   48.5004


maxHeight =

   12.5141


maxVx =

   21.7445


maxVy =

   16.2784


maxThrust =

  152.8367


maxRange =

   60.2658


maxHeight =

   17.2294
```

```
maxVx =

   25.6057


maxVy =

   20.5830


maxThrust =

  191.0459


maxRange =

   68.7916


maxHeight =

   21.2302


maxVx =

   28.6721


maxVy =

   24.1855


maxThrust =

  229.2550


maxRange =

   74.5488


maxHeight =

   24.3666


maxVx =
```

```
    31.1281


maxVy =

    27.0727


maxThrust =

   267.4642


maxRange =

    81.7437


maxHeight =

    27.7866


maxVx =

    33.9948


maxVy =

    30.0481


maxThrust =

   305.6734


end
```

Plotting

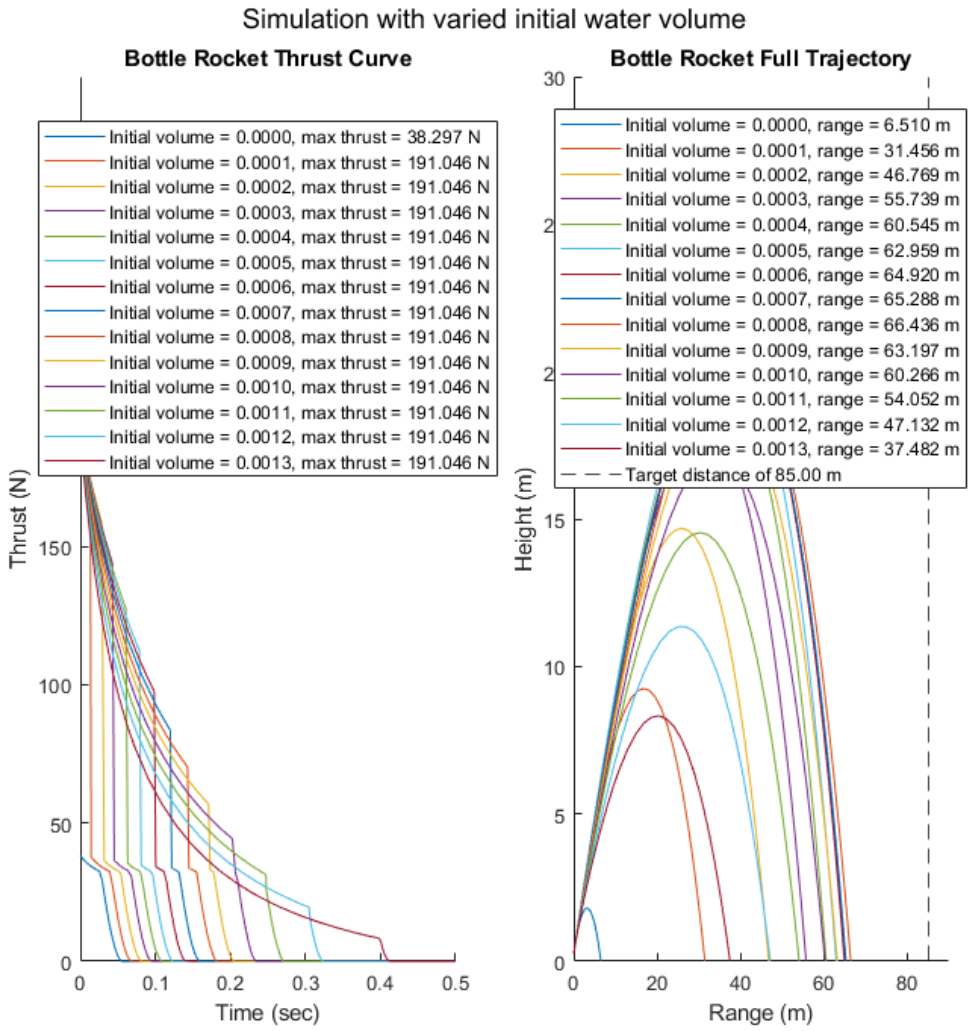
Plot the trajectory/thrust with varied initial water volume

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

```
subplot(1,2,1);

hold on

% Thrust
title("Bottle Rocket Thrust Curve");

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

for k = 1:length(volumeSim)
    plots(k) = plot(volumeSim(k).time, volumeSim(k).thrust);
    label(k) = sprintf("Initial volume = %.4f, max thrust = %.3f N",
 volumeSim(k).initialWaterVolume, max(volumeSim(k).thrust));
end

xlim([0, 0.5]);
ylim([0, 320]);
xlabel("Time (sec)");
ylabel("Thrust (N)");
legend(plots, label, 'Location', 'best');

hold off

subplot(1,2,2);

hold on;

% Trajectory
title("Bottle Rocket Full Trajectory");

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

% Trajectory
for k = 1:length(volumeSim)
    hold on;
    rocketX = volumeSim(k).rocketX;
    rocketZ = volumeSim(k).rocketZ;
    label(k) = sprintf("Initial volume = %.4f, range = %.3f m",
 volumeSim(k).initialWaterVolume, max(rocketX));
    plots(k) = plot(real(rocketX), real(rocketZ));
end

plots(k+1) = xline(85, '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.
```

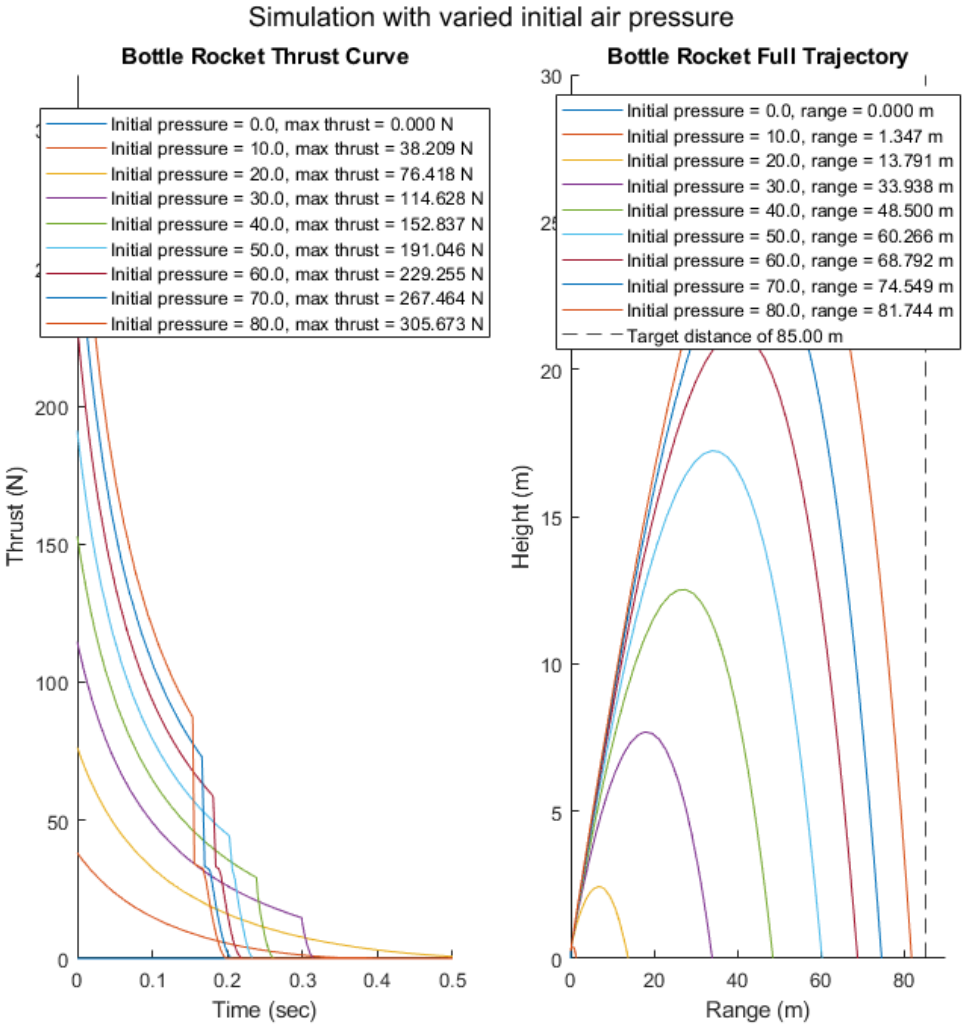


Plot the trajectory with varied initial air pressure

```
f = figure();
f.Position = [940 100 740 740]; % Start at (100, 100), end at (100 +
 740, 100 + 740)

sgtitle("Simulation with varied initial air pressure");

subplot(1,2,1);
```

```
hold on

% Thrust
title("Bottle Rocket Thrust Curve");

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

for k = 1:length(pressureSim)
    plots(k) = plot(pressureSim(k).time, pressureSim(k).thrust);
    label(k) = sprintf("Initial pressure = %.1f, max thrust = %.3f N",
 pressureSim(k).initialGagePressure, max(pressureSim(k).thrust));
end

xlim([0, 0.5]);
ylim([0, 320]);
xlabel("Time (sec)");
ylabel("Thrust (N)");
legend(plots, label, 'Location', 'best');

hold off

subplot(1,2,2);

% Trajectory
title("Bottle Rocket Full Trajectory");

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


for k = 1:length(pressureSim)
    hold on;
    rocketX = pressureSim(k).rocketX;
    rocketZ = pressureSim(k).rocketZ;
    label(k) = sprintf("Initial pressure = %.1f, range = %.3f m",
 pressureSim(k).initialGagePressure, max(rocketX));
    plots(k) = plot(real(rocketX), real(rocketZ));
end

plots(k+1) = xline(85, '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.
```



Published with MATLAB® R2021a