

Estimated Landing Zone (x,y): (71,2) [m]

Rocket Equation

Table of Contents

.....	1
House Keeping	1
Monte Carlo loop	1
Uncertainties in Values	1
Values pulled from static test stand report	2
Initial Values	2
ODE45 Start	3
Landing Points	3
Plot	3

```
%Rocket Equation calculations for a Rocket
%Zak Reichenbach
%4/5/2021
```

House Keeping

```
clc;
clear all;
close all;
```

Monte Carlo loop

```
%How many simulations?
simulations = 100;

for k = 1:simulations
```

Uncertainties in Values

```
SigmaLA = -1:.25:1; %Launch Angle [deg]
SigmaWater = -.0005:.000075:.0005; %Water Mass [kg]
SigmaDrag = -.1:.025:.1; %Drag Coeff.
SigmaDir = -1:.25:1; %XY launch angle (40 deg from
    north to west with 3deg variation)

%Random Value Selection
Picker = randi([1 length(SigmaLA)],1,1);
DeltaLA = SigmaLA(Picker);
Picker = randi([1 length(SigmaWater)],1,1);
DeltaWater = SigmaWater(Picker);
Picker = randi([1 length(SigmaDrag)],1,1);
DeltaDrag = SigmaDrag(Picker);
Picker = randi([1 length(SigmaDir)],1,1);
DeltaDir = SigmaDir(Picker);
```

Values pulled from static test stand report

```
% ISP = [1.421 1.866 1.187 1.582 1.338 1.401 1.511 1.251 1.473 1.457 1.531 1.680 1.352 1.783 1.593  
1.740 1.682]; % ISP_Avg = mean(ISP); % ISP_Sigma = std(ISP);
```

```
ISP_Avg = 2.346; %Averaged from my and Roberts ISP Calculation for  
600g.  
ISP_Sigma = .1;
```

```
%Error in ISP?  
INeg = round((ISP_Avg - ISP_Sigma),1);  
IPos = round((ISP_Avg + ISP_Sigma),1);  
%Random Value Selection  
ISP_Range = linspace(INeg,IPos);  
Picker = randi([1 100],1,1);  
ISP_Avg = ISP_Range(Picker);
```

Initial Values

```
BottleDiameter = 10.5; %cm  
mdry = .16; %[kg]  
mp = .6 + DeltaWater; %[kg] Optimized Case  
mwet = mdry+mp; %[kg]  
Cd = 0.3 + DeltaDrag;  
LaunchAngle = 45 + DeltaLA; %[deg]  
% % p0 = 40; %[psi]  
% % Temp = 63; %[deg]  
g = 9.81; %[m/s^2]  
  
%Passable Constant Struct  
Values =  
    num2cell([g;0.961;1000;2.1;10.5;mdry;Cd;0.0;LaunchAngle;0.0;0.25;0.5]);  
Names  
    =["g","RoeAmb","RoeWat",'Dt','Db','MBot','CD','V0','Theta','X0','Z0','L'];  
C = cell2struct(Values,Names,1); %Create Struct of Constants  
  
%Bottle Area for Drag  
Ab = pi/4*((C.Db*10^-2))^2;  
  
%Initial velocity  
Delta_V = ISP_Avg*g*log(mwet/mdry);  
  
%Initial velocity componentsm  
VX = Delta_V*cosd(LaunchAngle)*cosd(DeltaDir);  
VZ = Delta_V*sind(LaunchAngle);  
VY = Delta_V*sind(DeltaDir);  
  
%Wind Factor  
theta = -22.25:.5:22.25;  
theta0 =355; %[deg]  
theta = theta+theta0;
```

```
w =randi([1,length(theta)],1,1);
WindX =3.57632*cosd(theta(w));
WindY =3.57632*sind(theta(w));
```

ODE45 Start

```
% Intial State of the rocket to enter into ODE45
Properties0 = [C.Z0 C.X0 0 VZ VX VY];

tspan = [0 10];

terminalCond = odeset('Events', @hitGround);

[t,Properties] = ode45(@(t,s) Rocket(t,s,C,Ab,WindX,WindY), tspan,
    Properties0, terminalCond);

Q{k} = Properties;

end
```

Landing Points

```
for i =1:simulations
    Zs{i} = Q{i}(:,1);
    Xs{i} = Q{i}(:,2);
    Ys{i} = Q{i}(:,3);
end

for j = 1:simulations

    Xf(j) = Xs{j}(end)';
    Yf(j) = Ys{j}(end)';

end
```

Plot

```
figure(1)
plot3(Properties(:,2),Properties(:,3),Properties(:,1))
grid on
title('A Bottle Rocket Trajectory')
xlabel('X [m]')
ylabel('Y [m]')
zlabel('Z [m]')

save('landingPoints.mat','Xf','Yf','simulations')

figure(2)
for i = 1:simulations
    hold on
    plot3(Q{i}(:,2),Q{i}(:,3),Q{i}(:,1))
```

```

end
title('Simulated Trajectories')
ylabel('Y Cross Range [m]')
xlabel('X Down Range [m]')
grid on
hold off

function state = Rocket(t,state_i,C,AreaBottle,WindX,WindY)

    %State Vector
    state = zeros(4,1);
    Z = state_i(1);           %Z position [m]
    X = state_i(2);           %X position [m]
    Y = state_i(3);
    V_Z = state_i(4);         %Velocity [m/s]
    V_X = state_i(5);         %Velocity [m/s]
    V_Y = state_i(6);

    %Heading Calculations
    Velocity = sqrt(V_Z^2+V_X^2+V_Y^2);           %[m/s]
    %Heading position
    if (sqrt((X-C.X0)^2+(Z-C.Z0)^2) < C.L)       %Launch angle 45 deg
        hz = sind(C.Theta);
        hx = cosd(C.Theta);
        hy = 0;
    else
        hz = V_Z/Velocity;                       %sin(theta)
        hx = V_X/Velocity;                       %cos(theta)
        hy = V_Y/Velocity;
    end

    q_z = (1/2)*C.RoeAmb*V_Z^2;                   %Dynamic Pressure
    q_x = (1/2)*C.RoeAmb*V_X^2;
    q_y = (1/2)*C.RoeAmb*V_Y^2;
    Drag_Z = q_z*C.CD*AreaBottle;                 %Drag
    Drag_X = q_x*C.CD*AreaBottle;
    Drag_Y = q_y*C.CD*AreaBottle;
    %%%%%%%%%%CALCULATE THETA_DOT HERE

    Sum_FX = - Drag_X*hx;
    Sum_FY = - Drag_Y*hy;
    Sum_FZ = - Drag_Z*hz - C.MBot*C.g;
    %Calculate the X and Y accelerations
    Accel_X = Sum_FX /C.MBot;
    Accel_Y = Sum_FY /C.MBot;
    Accel_Z = Sum_FZ /C.MBot;

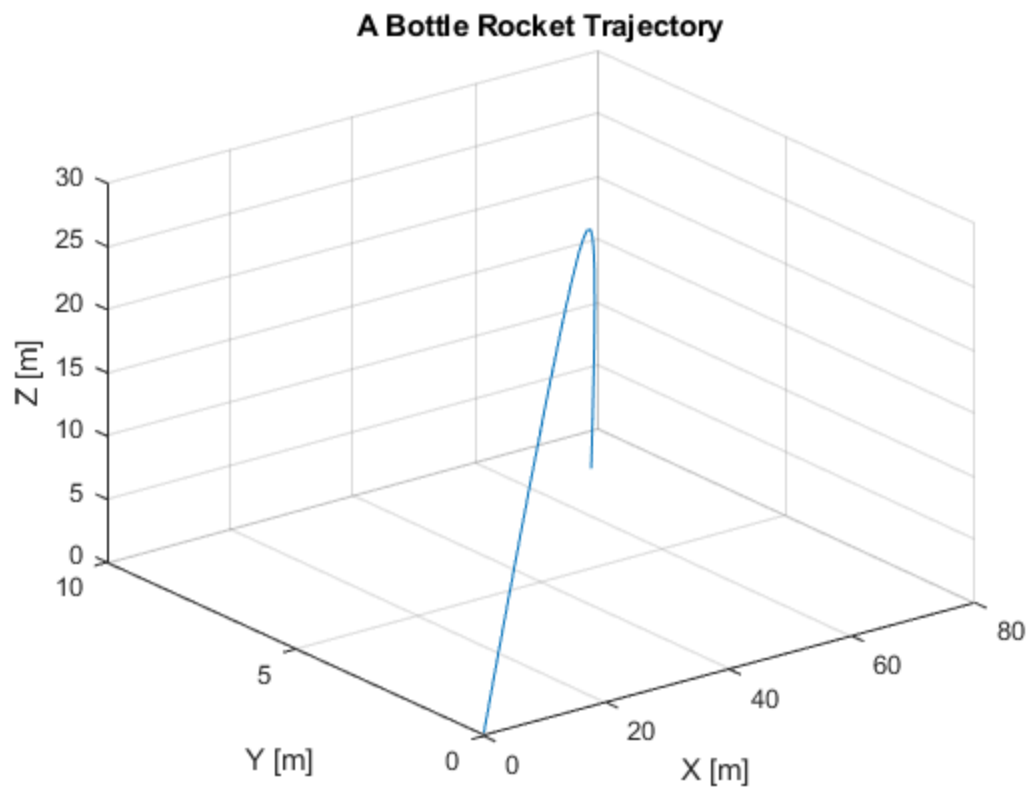
    %Outputing the derivative of things on top of our state vectors
    state(1) = V_Z;                               %[m/s]
    state(2) = V_X - WindX;                       %[m/s]
    state(3) = V_Y - WindY;

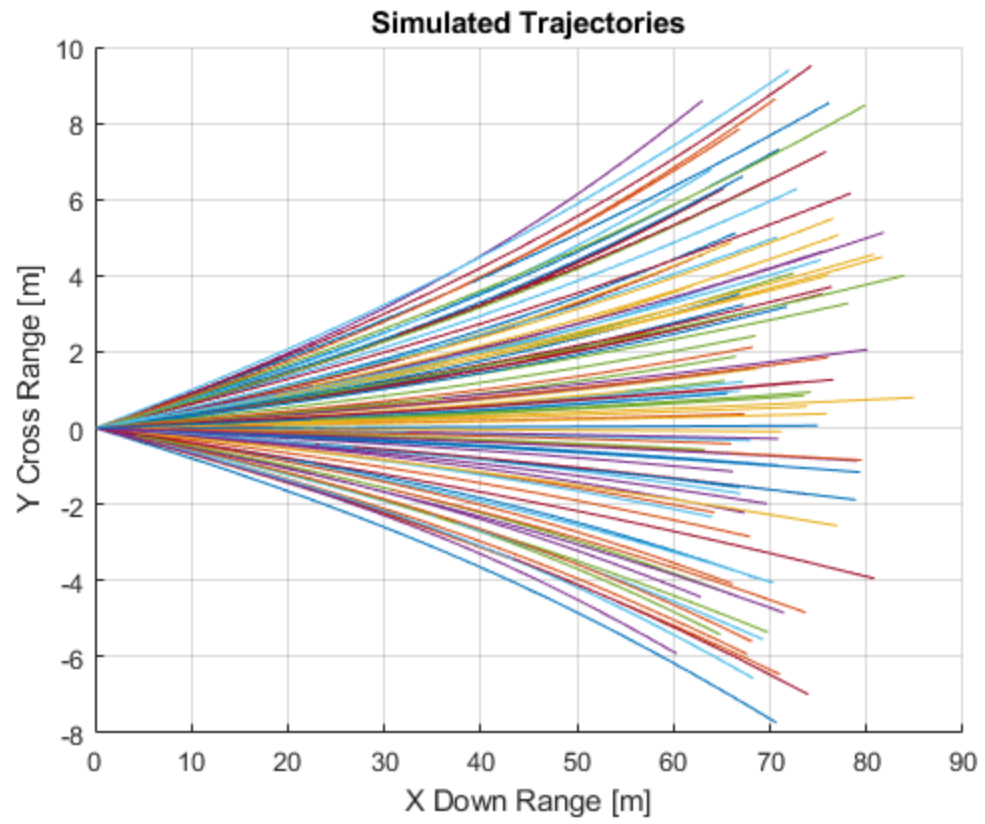
```

```
state(4) = Accel_Z;           %[m/s^2]
state(5) = Accel_X;           %[m/s^2]
state(6) = Accel_Y;
```

```
end
```

```
%This stops the ODE45 call when the rocket would hit the ground
function [value, isterminal, direction] = hitGround(t,state)
value  =(state(1) <= 0);
isterminal = 1;
direction = 0;
end
```





Published with MATLAB® R2019a

```
clc; clear all; close all;
```

Replace this section of code with your real data

Simulate and plot 100 data points, (YOU SHOULD USE REAL DATA HERE!)

```
load('landingPoints.mat')

N = simulations; % Number of points to simulate
x = Xf; % Data From ISP Rocket Model
y = Yf;

figure; plot(x,y,'k.','markersize',6)
axis equal; grid on; xlabel('X Down Range [m]'); ylabel('Y Cross Range [m]'); title('Landing Ellipses'); hold on;

% Calculate covariance matrix
P = cov(x,y)
mean_x = mean(x);
mean_y = mean(y);

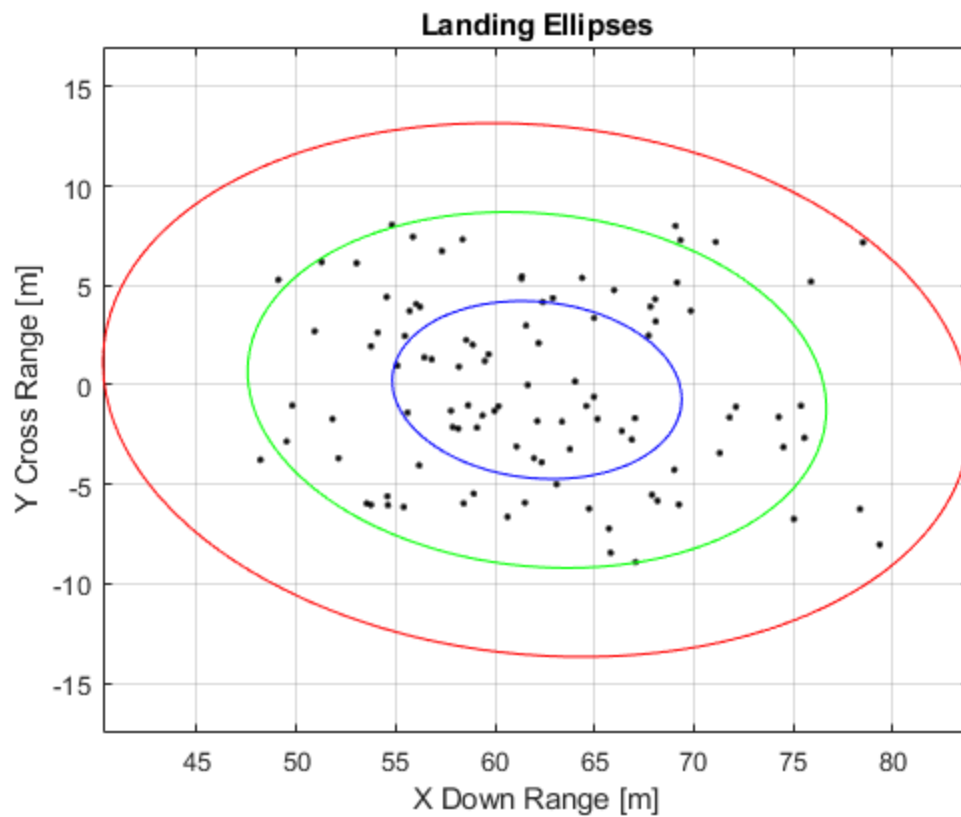
% Calculate the define the error ellipses
n=N; % Number of points around ellipse
p=0:pi/n:2*pi; % angles around a circle

[eigvec,eigval] = eig(P); % Compute eigen-stuff
xy_vect = [cos(p'),sin(p')] * sqrt(eigval) * eigvec'; % Transformation
x_vect = xy_vect(:,1);
y_vect = xy_vect(:,2);

% Plot the error ellipses overlaid on the same figure
plot(1*x_vect+mean_x, 1*y_vect+mean_y, 'b')
plot(2*x_vect+mean_x, 2*y_vect+mean_y, 'g')
plot(3*x_vect+mean_x, 3*y_vect+mean_y, 'r')

P =

    52.9444    -3.3892
   -3.3892    20.0208
```

Published with MATLAB® R2019a