Indian Institute of Space Science and Technology

AE 734 Design and Modelling of Rocket Propulsion

Assignment - 1

Author : Mohd Babar Malik (SC23M048)
M.Tech : Structures and Design

May 3, 2024

1.Determine the trajectory of a sounding rocket

a. Based on RH 300 details

The aim of this problem is to obtain the trajectory of a rocket vehicle subject to given constraints of a specific mission. The following equations discussed here give the basic method of determining the trajectory of the vehicle.

$$\frac{dv}{dt} = \frac{T}{m} - \frac{D}{m} - g\sin\gamma$$
$$v\frac{d\gamma}{dt} = -\left(g - \frac{v^2}{R_E + h}\right)\cos\gamma$$
$$\frac{dx}{dt} = \frac{R_E}{R_E + h}v\cos\gamma$$
$$\frac{dh}{dt} = v\sin\gamma$$

I am using MATLAB to numerically solve the above-mentioned four equations for the given conditions of RH − 300 sounding rocket. As a designer i want the rocket altitude variations with time. Subsequently optimisations can be made to reach the mission constraints. the codes are given in APPENDIX .
Trajectory of Sounding Rocket RH300

Parameter	Value
Initial mass m_p	563 kg
Propellant mass m_p	329.6 kg
Specific Impulse I_{sp}	260.5 kg
Burn time T_b	21 s
I	842kNs
γ	82°
Cd	06 (subsonic), 1 (transonic), 0.55 (supersonic, M = 5)
Payload mass m_{pl}	80 kg

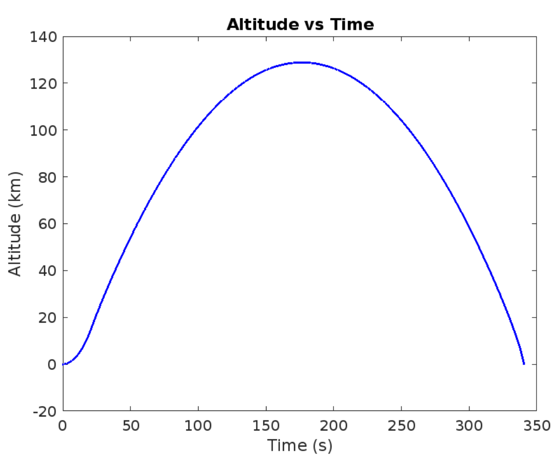


Figure 1: Time Vs Altitude

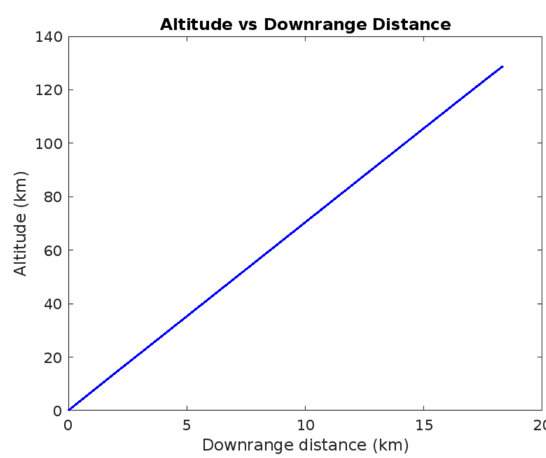


Figure 2: Down Range Vs Altitude

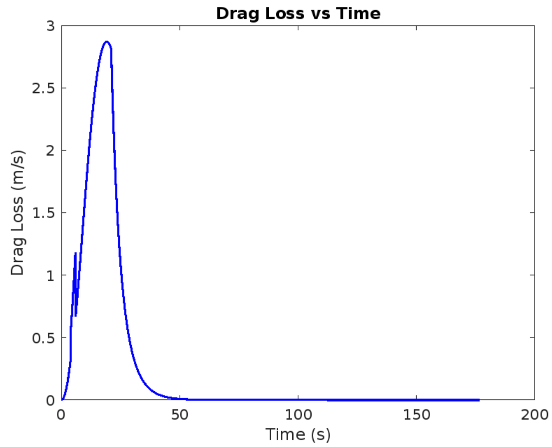


Figure 3: Drag loss Vs Time

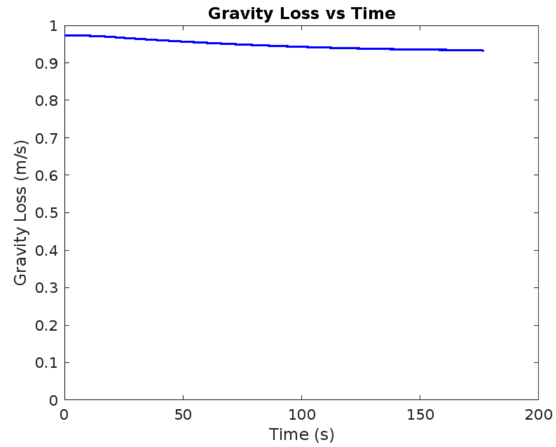


Figure 4: Gravity loss Vs Time

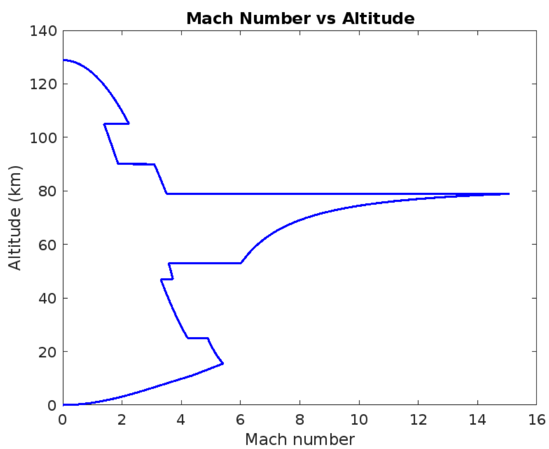


Figure 5: Mach number Vs Altitude

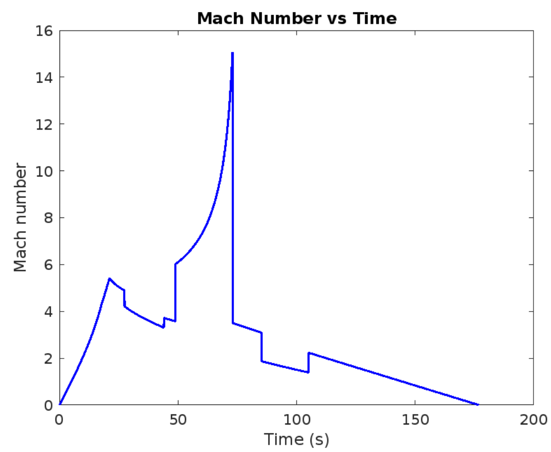


Figure 6: Mach number Vs Time

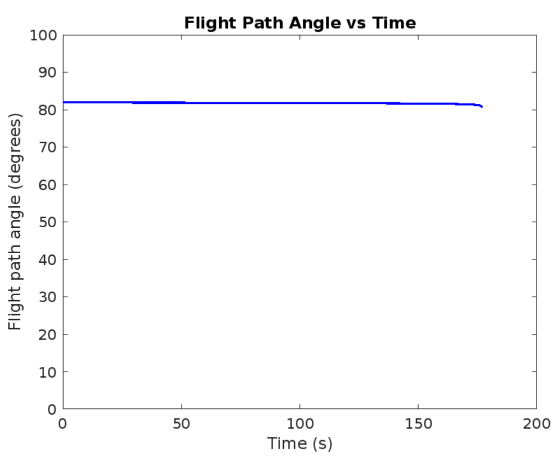


Figure 7: Fight path angle Vs Time

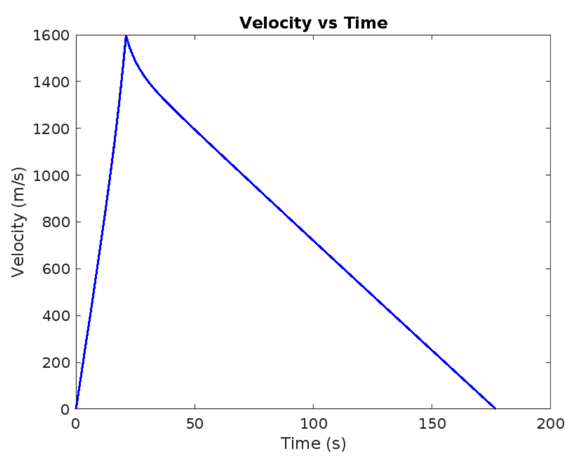


Figure 8: Velocity Vs Time

Results :

Data obtained from the MATLAB code.

Total drag loss = 0.4670 km/s

Total gravity loss = 1.6780 km/s

Burnout time = 21.2 s

Velocity at Burnout = 1.5973 km/s

Altitude at Burnout = 15.5043 km

Downrange distance at Burnout = 2.1852 km

Maximum Altitude = 128.8436 km

Time for Max Altitude = 179.8000 s

Downrange distance at max altitude = 18.3452 km

MATLAB CODE:

```
clear all
clc

% Rocket parameters
m0 = 563+60; % Initial mass
mp = 329.6; % Propellent mass
Isp = 260.5; % Specific Impulse
tburn = 21; % Burnout time
TI=842000; % Total Impulse
d=0.31; % Diameter of Rocket
Y0=82; % Flight path abgle

% Constants
G=6.67408e-11; % Gravitational Constant
M=5.9776e+24; % Mass of Earth
R=6371e+3; % Radius of Earth
h0=7500;
rho0=1.225; % Density of air at sea level
T0=288.16; % Temperature at sea level

g0=G*M/(R^2); % acceleration due to gravity
Tr =TI/tburn; % Thrust
me=mp/tburn; % mass flow rate
A=(pi* d^2)/4; % Cross Section area
mf=m0-mp; % Final mass

n=4000; % Total number of iterations
dt=0.1; % increment (s)
t=0:1:n-1; % Time range

%initialization
g=zeros(1,n);
h=zeros(1,n);
Y=zeros(1,n);
V=zeros(1,n);
D=zeros(1,n);
x=zeros(1,n);
rho=zeros(1,n);
m=zeros(1,n);
T=zeros(1,n);
C=zeros(1,n);
Mach=zeros(1,n);
Vdl=zeros(1,n);

%t=0 values
g(1)=g0;
Y(1)=Y0;
rho(1)= rho0;
m(1)=m0;
T(1)=T0;

% 0<=t<=tmax values
for i=2:1:n
```

```
    if h(i-1)>=0
        if m(i-1)>mf
            m(i)=m(i-1)-me*dt;
            V(i) = V(i-1) + (((Tr/m(i)) - D(i-1)/m(i) - g(i)*sind(Y(i-1)))
                *dt);
            tb=i;
        else
            m(i-1)=mf;
            V(i)= V(i-1)-(((g(i-1)*sind(Y(i-1)))+D(i-1)/m(i-1))*dt);
        end
        h(i) = h(i-1) + ((V(i)*sind(Y(i-1)))*dt);
        g(i) = (G*M)/((R+h(i))^2);
        Y(i)=Y(i-1)-(cosd(Y(i-1))*(g(i)-((V(i)^2)/(R+h(i))))*dt/V(i));
        rho(i)=rho(1)*exp(-h(i)/h0);
        if V(i-1)>=0
            x(i)=x(i-1)+(cosd(Y(i-1))*V(i-1)*R*dt/(R+h(i)));
        else
            x(i)=x(i-1)-(cosd(Y(i-1))*V(i-1)*R*dt/(R+h(i)));
        end
        if h<=11000
            T(i)=288.16-(0.0065*(h(i)));
        elseif h<=25000
            T(i)=216.66;
        elseif h<=47000
            T(i)=216.66+(0.003*(h(i)));
        elseif h<=53000
            T(i)=282.66;
        elseif h<=79000
            T(i)=282.66-(0.00345*(h(i)));
        elseif h<=90000
            T(i)=193;
        elseif h<=105000
            T(i)=193+(0.0037*(h(i)));
        else
            T(i)=225.66;
        end
        C(i)=sqrt(1.4*287*T(i));
        Mach(i)=abs(V(i))/C(i);
        if Mach(i)<0.8
            Cd=0.6;
        elseif Mach(i)>1.2
            Cd=0.55;
        else
            Cd=1;
        end
        D(i) = 0.5* rho(i)* V(i)^2 *Cd*A;
    else
        t_max=(i-1);
        break;
    end
end

[hmax, t_hmax] = max(h);
```

```
% Drag loss
Vdl=D*dt./m;
l1=sum(Vdl(1:t_hmax));

% Gravity loss
Vgl=(g.*sind(Y))*dt;
l2=sum(Vgl(1:t_hmax));


fprintf('Total drag loss = %f km/s',0.001*l1)
fprintf('Total gravity loss = %f km/s',0.001*l2)
fprintf('Burnout time = %f s',0.1*tb)
fprintf('Velocity at Burnout = %f km/s',0.001*V(tb))
fprintf('Altitude at Burnout = %f km',0.001*h(tb))
fprintf('Downrange distance at Burnout = %f km',0.001*x(tb))
fprintf('Maximum Altitude = %f km',0.001*h(t_hmax))
fprintf(['Time for Max Altitude = %f s'],0.1*t(t_hmax))
fprintf('Downrange distance at max altitude = %f km',0.001*x(t_hmax))

% Plot 1: Altitude vs Time
figure;
plot(0.1*t(1:t_max),0.001*h(1:t_max), 'LineWidth', 1.5, 'Color', 'blue')
xlabel("Time (s)")
ylabel("Altitude (km)")
title("Altitude vs Time")
saveas(gcf, 'altitude_vs_time.png'); % Save the plot as a PNG image

% Plot 2: Altitude vs Downrange Distance
figure;
plot(0.001*x(1:t_hmax),0.001*h(1:t_hmax), 'LineWidth', 1.5, 'Color', 'blue
    ')
xlabel("Downrange distance (km)")
ylabel("Altitude (km)")
title("Altitude vs Downrange Distance")
saveas(gcf, 'altitude_vs_downrange.png'); % Save the plot as a PNG image

% Plot 3: Mach Number vs Altitude
figure;
plot(Mach(1:t_hmax),0.001*h(1:t_hmax), 'LineWidth', 1.5, 'Color', 'blue')
xlabel("Mach number")
ylabel("Altitude (km)")
title("Mach Number vs Altitude")
saveas(gcf, 'mach_vs_altitude.png'); % Save the plot as a PNG image

% Plot 4: Drag Loss vs Time
figure;
plot(0.1*t(1:t_hmax),Vdl(1:t_hmax), 'LineWidth', 1.5, 'Color', 'blue')
xlabel("Time (s)")
ylabel("Drag Loss (m/s)")
title("Drag Loss vs Time")
saveas(gcf, 'drag_loss_vs_time.png'); % Save the plot as a PNG image

% Plot 5: Gravity Loss vs Time
figure;
```

```
plot(0.1*t(1:t_hmax),Vgl(1:t_hmax), 'LineWidth', 1.5, 'Color', 'blue')
ylim([0,1])
xlabel("Time (s)")
ylabel("Gravity Loss (m/s)")
title("Gravity Loss vs Time")
saveas(gcf, 'gravity_loss_vs_time.png'); % Save the plot as a PNG image

% Plot 6: Velocity vs Time
figure;
plot(0.1*t(1:t_hmax),V(1:t_hmax), 'LineWidth', 1.5, 'Color', 'blue')
xlabel("Time (s)")
ylabel("Velocity (m/s)")
title("Velocity vs Time")
saveas(gcf, 'velocity_vs_time.png'); % Save the plot as a PNG image

% Plot 7: Flight Path Angle vs Time
figure;
plot(0.1*t(1:t_hmax),Y(1:t_hmax), 'LineWidth', 1.5, 'Color', 'blue')
ylim([0,100])
xlabel("Time (s)")
ylabel("Flight path angle (degrees)")
title("Flight Path Angle vs Time")
saveas(gcf, 'flight_path_angle_vs_time.png'); % Save the plot as a PNG
   image

% Plot 8: Mach Number vs Time
figure;
plot(0.1*t(1:t_hmax),Mach(1:t_hmax), 'LineWidth', 1.5, 'Color', 'blue')
xlabel("Time (s)")
ylabel("Mach number")
title("Mach Number vs Time")
saveas(gcf, 'mach_vs_time.png'); % Save the plot as a PNG image
```

(b). Based on own design of sounding rocket

Own design of Sounding Rocket

RVT 4 (also known as Vanguard 4) was the fourth flight of the Vanguard family of American sounding rockets. It launched on March 27, 1959, and reached an altitude of 25 miles (40 kilometers). It was originally planned to have a satellite payload but encountered an anomaly early in the flight. RVT 4 is significant as it was the last successful Vanguard sounding rocket and marked a period of transition to more advanced satellite launches through the Explorer and Jupiter missions.

The aim of this problem is to obtain the trajectory of an own design of sounding rocket vehicle subject to given constraints of a specific mission. The following equations discussed here give the basic method of determining the trajectory of the vehicle.

$$\frac{dv}{dt} = \frac{T}{m} - \frac{D}{m} - g\sin\gamma$$

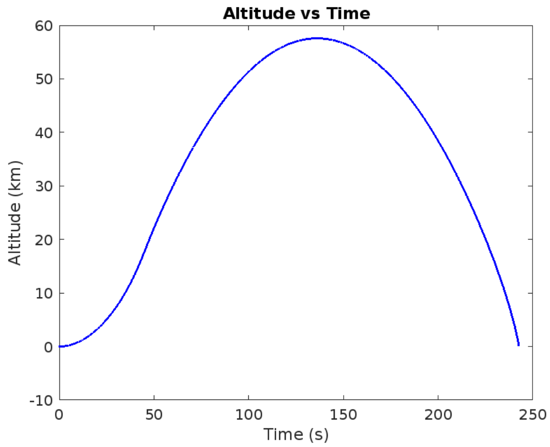
$$v\frac{d\gamma}{dt} = -\left(g - \frac{v^2}{R_E + h}\right)\cos\gamma$$

$$\frac{dx}{dt} = \frac{R_E}{R_E + h} v\cos\gamma$$

$$\frac{dh}{dt} = v\sin\gamma$$

I am using MATLAB to numerically solve the above-mentioned four equations for the given conditions for an own design of sounding rocket. As a designer i want the rocket altitude variations with time. Subsequently optimisations can be made to reach the mission constraints. the codes are given in APPENDIX .
Trajectory of an own design sounding rocket

Parameter	Value
Initial mass $\mathrm{m_p}$	10750 kg
Propellant mass $\mathrm{m_p}$	6936 kg
Specific Impulse $\mathrm{I_{sp}}$	320 kg
Burn time $\mathrm{T_b}$	45 s
I	7380kNs
γ	82°
Cd	06 (subsonic), 1 (transonic), 0.55 (supersonic, M = 5)
Payload mass $\mathrm{m_{pl}}$	100 kg



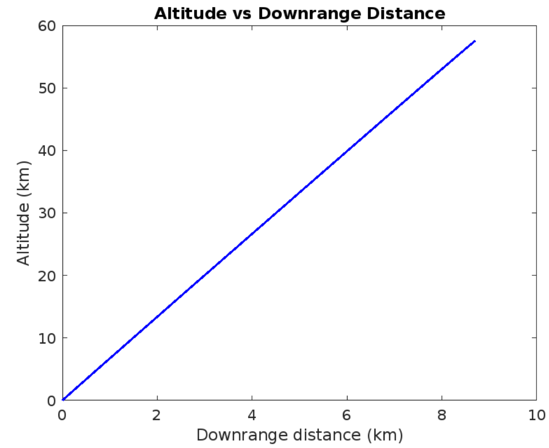
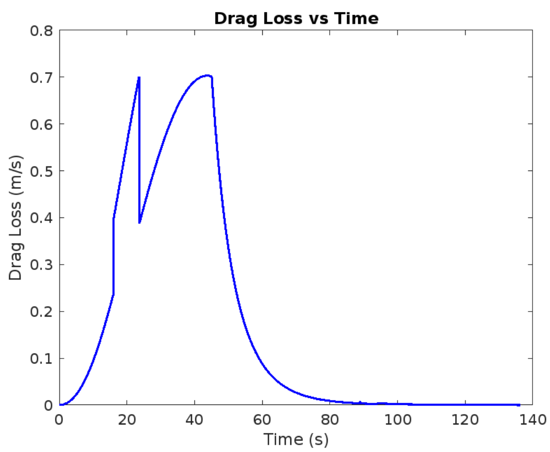


Figure 9: Time Vs Altitude

Figure 10: Down Range Vs Altitude



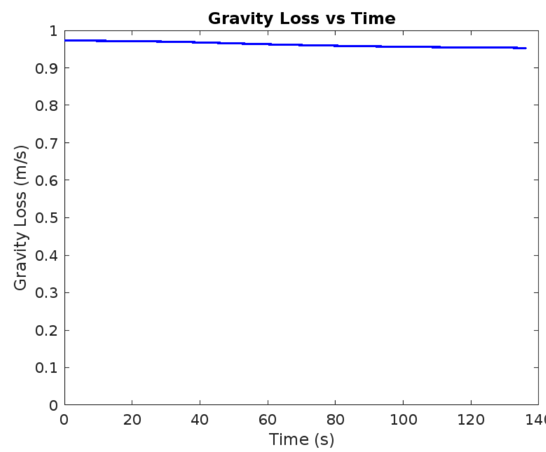
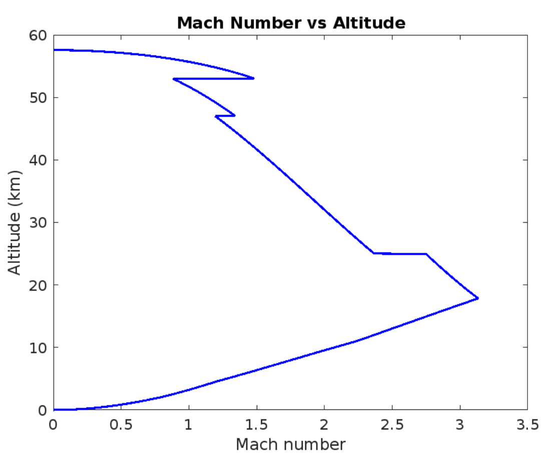


Figure 11: Drag loss Vs Time

Figure 12: Gravity loss Vs Time



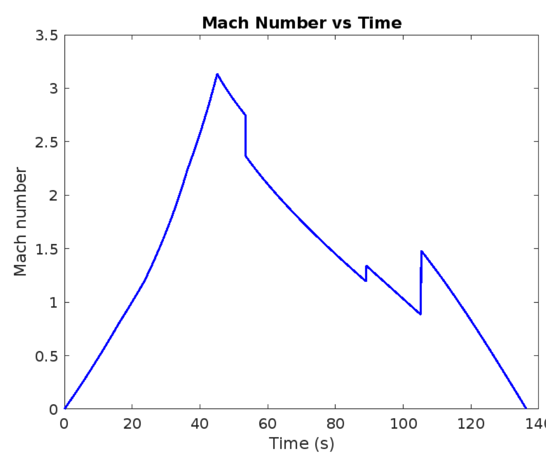


Figure 13: Mach number Vs Altitude

Figure 14: Mach number Vs Time

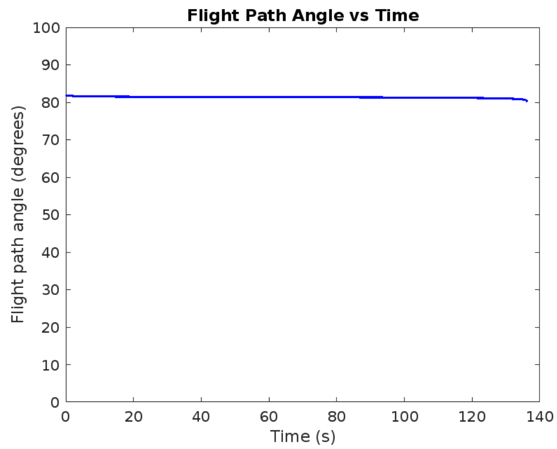


Figure 15: Fight path angle Vs Time

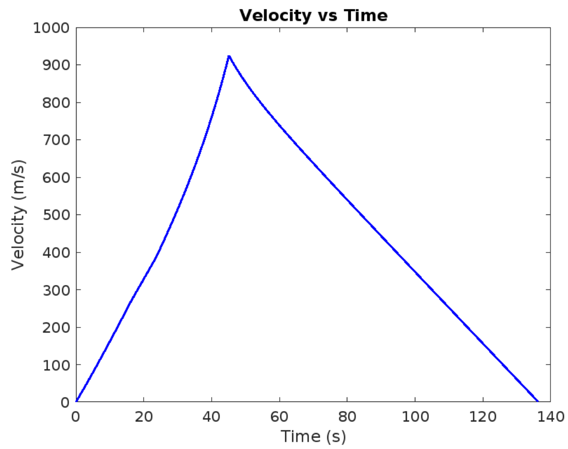


Figure 16: Velocity Vs Time

Results :
Data obtained from the MATLAB code.

Total drag loss = 0.23343 km/s

Total gravity loss = 1.3123 km/s

Burnout time = 45.2 s

Velocity at Burnout = 0.9247 km/s

Altitude at Burnout = 17.8536 km

Downrange distance at Burnout = 2.6703 km

Maximum Altitude = 57.5575 km

Time for Max Altitude = 136.30 s

Downrange distance at max altitude = 8..7061 km

MATLAB CODE:

```
% Rocket parameters
%RVT 4 (also known as Vanguard 4)
%Payload mass is 100 kg
m0 = 10750; % Initial mass
mp = 6936; % Propellant mass
Isp = 320; % Specific Impulse
tburn = 45; % Burnout time
%TI=842000; % Total Impulse
d=0.31; % Diameter of Rocket
Y0=82; % Flight path angle

% Constants
G=6.67408e-11; % Gravitational Constant
M=5.9776e+24; % Mass of Earth
R=6371e+3; % Radius of Earth
h0=7500;
rho0=1.225; % Density of air at sea level
T0=288.16; % Temperature at sea level

g0=G*M/(R^2); % acceleration due to gravity
Tr =164000; % Thrust
me=mp/tburn; % mass flow rate
A=(pi* 3^2)/4; % Cross Section area
mf=m0-mp; % Final mass

n=4000; % Total number of iterations
dt=0.1; % increment (s)
t=0:1:n-1; % Time range

% Initialization
g=zeros(1,n);
h=zeros(1,n);
Y=zeros(1,n);
V=zeros(1,n);
D=zeros(1,n);
x=zeros(1,n);
rho=zeros(1,n);
m=zeros(1,n);
T=zeros(1,n);
C=zeros(1,n);
Mach=zeros(1,n);
Vdl=zeros(1,n);

% t=0 values
g(1)=g0;
Y(1)=Y0;
rho(1)= rho0;
m(1)=m0;
T(1)=T0;

% 0<=t<=tmax values
for i=2:1:n
```

```
    if h(i-1)>=0
        if m(i-1)>mf
            m(i)=m(i-1)-me*dt;
            V(i) = V(i-1) + (((Tr/m(i)) - D(i-1)/m(i) - g(i)*sind(Y(i-1)))
                *dt);
            tb=i;
        else
            m(i-1)=mf;
            V(i)= V(i-1)-(((g(i-1)*sind(Y(i-1)))+D(i-1)/m(i-1))*dt);
        end
        h(i) = h(i-1) + ((V(i)*sind(Y(i-1)))*dt);
        g(i) = (G*M)/((R+h(i))^2);
        Y(i)=Y(i-1)-(cosd(Y(i-1))*(g(i)-((V(i)^2)/(R+h(i))))*dt/V(i));
        rho(i)=rho(1)*exp(-h(i)/h0);
        if V(i-1)>=0
            x(i)=x(i-1)+(cosd(Y(i-1))*V(i-1)*R*dt/(R+h(i)));
        else
            x(i)=x(i-1)-(cosd(Y(i-1))*V(i-1)*R*dt/(R+h(i)));
        end
        if h<=11000
            T(i)=288.16-(0.0065*(h(i)));
        elseif h<=25000
            T(i)=216.66;
        elseif h<=47000
            T(i)=216.66+(0.003*(h(i)));
        elseif h<=53000
            T(i)=282.66;
        elseif h<=79000
            T(i)=282.66-(0.00345*(h(i)));
        elseif h<=90000
            T(i)=193;
        elseif h<=105000
            T(i)=193+(0.0037*(h(i)));
        else
            T(i)=225.66;
        end
        C(i)=sqrt(1.4*287*T(i));
        Mach(i)=abs(V(i))/C(i);
        if Mach(i)<0.8
            Cd=0.6;
        elseif Mach(i)>1.2
            Cd=0.55;
        else
            Cd=1;
        end
        D(i) = 0.5* rho(i)* V(i)^2 *Cd;
    else
        t_max=(i-1);
        break;
    end
end

[hmax, t_hmax] = max(h);
```

```
% Drag loss
Vdl=D*dt./m;
l1=sum(Vdl(1:t_hmax));

% Gravity loss
Vgl=(g.*sind(Y))*dt;
l2=sum(Vgl(1:t_hmax));

fprintf('Total drag loss = %f km/s\n',0.001*l1)
fprintf('Total gravity loss = %f km/s\n',0.001*l2)
fprintf('Burnout time = %f s\n',0.1*tb)
fprintf('Velocity at Burnout = %f km/s\n',0.001*V(tb))
fprintf('Altitude at Burnout = %f km\n',0.001*h(tb))
fprintf('Downrange distance at Burnout = %f km\n',0.001*x(tb))
fprintf('Maximum Altitude = %f km\n',0.001*h(t_hmax))
fprintf('Time for Max Altitude = %f s\n',0.1*t(t_hmax))
fprintf('Downrange distance at max altitude = %f km\n',0.001*x(t_hmax))
% Plot 1: Altitude vs Time
figure;
plot(0.1*t(1:t_max),0.001*h(1:t_max), 'LineWidth', 1.5, 'Color', 'blue')
xlabel("Time (s)")
ylabel("Altitude (km)")
title("Altitude vs Time")
saveas(gcf, 'altitude_vs_time.png'); % Save the plot as a PNG image

% Plot 2: Altitude vs Downrange Distance
figure;
plot(0.001*x(1:t_hmax),0.001*h(1:t_hmax), 'LineWidth', 1.5, 'Color', 'blue
    ')
xlabel("Downrange distance (km)")
ylabel("Altitude (km)")
title("Altitude vs Downrange Distance")
saveas(gcf, 'altitude_vs_downrange.png'); % Save the plot as a PNG image

% Plot 3: Mach Number vs Altitude
figure;
plot(Mach(1:t_hmax),0.001*h(1:t_hmax), 'LineWidth', 1.5, 'Color', 'blue')
xlabel("Mach number")
ylabel("Altitude (km)")
title("Mach Number vs Altitude")
saveas(gcf, 'mach_vs_altitude.png'); % Save the plot as a PNG image

% Plot 4: Drag Loss vs Time
figure;
plot(0.1*t(1:t_hmax),Vdl(1:t_hmax), 'LineWidth', 1.5, 'Color', 'blue')
xlabel("Time (s)")
ylabel("Drag Loss (m/s)")
title("Drag Loss vs Time")
saveas(gcf, 'drag_loss_vs_time.png'); % Save the plot as a PNG image

% Plot 5: Gravity Loss vs Time
figure;
plot(0.1*t(1:t_hmax),Vgl(1:t_hmax), 'LineWidth', 1.5, 'Color', 'blue')
ylim([0,1])
```

```
xlabel("Time (s)")
ylabel("Gravity Loss (m/s)")
title("Gravity Loss vs Time")
saveas(gcf, 'gravity_loss_vs_time.png'); % Save the plot as a PNG image

% Plot 6: Velocity vs Time
figure;
plot(0.1*t(1:t_hmax),V(1:t_hmax), 'LineWidth', 1.5, 'Color', 'blue')
xlabel("Time (s)")
ylabel("Velocity (m/s)")
title("Velocity vs Time")
saveas(gcf, 'velocity_vs_time.png'); % Save the plot as a PNG image

% Plot 7: Flight Path Angle vs Time
figure;
plot(0.1*t(1:t_hmax),Y(1:t_hmax), 'LineWidth', 1.5, 'Color', 'blue')
ylim([0,100])
xlabel("Time (s)")
ylabel("Flight path angle (degrees)")
title("Flight Path Angle vs Time")
saveas(gcf, 'flight_path_angle_vs_time.png'); % Save the plot as a PNG
    image

% Plot 8: Mach Number vs Time
figure;
plot(0.1*t(1:t_hmax),Mach(1:t_hmax), 'LineWidth', 1.5, 'Color', 'blue')
xlabel("Time (s)")
ylabel("Mach number")
title("Mach Number vs Time")
saveas(gcf, 'mach_vs_time.png'); % Save the plot as a PNG image
```

2) Determine the mass distribution for each stage including propellant mass and empty mass (m_p, m_E) for

1. GSLV (LVM3) with payload mass of 3500 kg and delta-V of 12400s. Use the respective ISP and Structural mass ratio for each stage. Compare your calculations with actual data

Solution :

Given Data

$$\mathrm{m_{pl}} : 3500 \text{ kg}$$

$$\Delta \mathrm{V_{mission:}} \; : 12392 \text{ m/s}$$

	$\mathrm{V_{ei}}$	ε_i
Stage 1	2706.6	0.134
Stage 2	2873.7	0.124
Stage 3	4334.54	0.152

Following the same steps as mentioned above
(i) Firstly we have to find out to the Lagrange multiplier from the total V_{bo} equation

$$V_{bo} = \sum_{j=1}^{N} V_{ei} \ln\left(V_{ei}\eta - 1\right) - \ln\eta \sum_{i=1}^{N} V_{ei} - \sum_{i=1}^{N} V_{ei} \ln\left(V_{ei}\varepsilon_i\right)$$

We can substitute for $\mathrm{V_{bo}}, \mathrm{V_{ei}}, \varepsilon_\mathrm{i}$ from the data for the 3 stages and solve for η iteratively, here i have used MATLAB for calculating the same and i am getting

$$\eta = 0.589$$

(ii). Now we use the value of η to find out optimum mass ratios using the formula

$$n_i = \frac{V_{ei}\eta - 1}{V_{ei}\eta\varepsilon_i}$$

and i am getting,

$$n_1 = 2.8509, n_2 = 3.3705, n_3 = 4.040$$

(iii) Next we obtain the step masses of each stage, beginning with stage N and working our way down the stack to stage 1 using the relation

$$m_N = \frac{n_N - 1}{1 - n_N\varepsilon_N} m_{pl}$$

$$\mathrm{m_1} = 477225.164 \text{ kg}, \; \mathrm{m_2} = 126582.237 \text{ kg}, \; \mathrm{m_3} = 27580.661\text{Kg}$$

Stages/Mass	First stage (kg)	Second stage (kg)	Third stage (kg)
n	2.8509	3.3705	4.040
Mi	477225.164	126582.337	27580.661
Mp	408946.992	110886.127	23388.401
Ms	63278.171	15696.209	4192.260
Total mass of the vehicle in kg = 629888.162 kg			
Overall payload ratio =0.005557			

(iv) Similarly we can find out the empty mass and propellant mass at each stage

$$m_{Ei} = \epsilon_i * M_i, m_{pi} = m_i - m_{Ei}$$

Lagrangian multiplier

$$\eta = 0.589.$$

Conclusion :

The mass analysis of GSLV Mk III has been calculated using MATLAB, yielding a lift-off mass estimation of approximately 630 tons using the Lagrangian multiplier method. However, the actual mass of the vehicle is around 640 tons. The result closely approximates the actual vehicle mass.

MATLAB CODE:

```
syms x;

% Given parameters
mu3 = 3500; % kg
h = 500 * 1000; % m
G = 6.67 * 10^(-11); % m^3/(kg*s^2)
Re = 6378.1 * 1000; % m
Me = 5.97 * 10^24; % kg

ve1 = 2.706;   % km/s
ve2 = 2.873; % km/s
ve3 = 4.334; % km/s

sf1 = 0.134; % structural factor of first stage
sf2 = 0.124; % structural factor of second stage
sf3 = 0.152; % structural factor of third stage



Vt=12.392;

% Define the function
equation = ve1*log((ve1*x-1)/(ve1*sf1*x)) + ve2*log((ve2*x-1)/(ve2*sf2*x))
   + ve3*log((ve3*x-1)/(ve3*sf3*x)) - Vt;

% Solve using vpasolve
x_solution = vpasolve(equation == 0, x);

% Display the solution
disp('Solution:');
fprintf('%.6f\n', double(x_solution));

x = 0.598

% Calculate n1, n2, and n3
n1 = (ve1 * x - 1) / (ve1 * x * sf1);
n2 = (ve2 * x - 1) / (ve2 * x * sf2);
n3 = (ve3 * x - 1) / (ve3 * x * sf3);
disp('n1:');
fprintf('%.6f\n', double(n1));
disp('n2:');
fprintf('%.6f\n', double(n2));
disp('n3:');
fprintf('%.6f\n', double(n3));

% Calculate step mass
mi3 = ((n3 - 1) / (1 - n3 * sf3)) * mu3;
mi2 = ((n2 - 1) / (1 - n2 * sf2)) * (mu3 + mi3);
mi1 = ((n1 - 1) / (1 - n1 * sf1)) * (mu3 + mi2 + mi3);

disp('Step mass in each stage in kg:');
disp('mi1:');
```

```
fprintf('%.6f\n', double(mi1));
disp('mi2:');
fprintf('%.6f\n', double(mi2));
disp('mi3:');
fprintf('%.6f\n', double(mi3));

% Calculate empty mass or structural mass in each stage
ms1 = sf1 * mi1;
ms2 = sf2 * mi2;
ms3 = sf3 * mi3;


disp('Empty or structural mass in each stage in kg:');
disp('ms1:');
fprintf('%.6f\n', double(ms1));
disp('ms2:');
fprintf('%.6f\n', double(ms2));
disp('ms3:');
fprintf('%.6f\n', double(ms3));

% Calculate propellant mass of each stage
mp1 = mi1 - ms1;
mp2 = mi2 - ms2;
mp3 = mi3 - ms3;
disp('propellent mass of each stage in kg:');
disp('mp1:');
fprintf('%.6f\n', double(mp1));
disp('mp2:');
fprintf('%.6f\n', double(mp2));
disp('mp3:');
fprintf('%.6f\n', double(mp3));

% Total mass of the vehicle
m0 = mi1 + mi2 + mi3 + mu3;
disp('Total mass of the vehicle (m0) in kg:');
fprintf('%.6f\n', double(m0));

% Overall payload ratio (alpha)
alpha = mu3 / m0;
disp('Overall payload ratio (alpha):');
fprintf('%.6f\n', double(alpha));
```

b. A vehicle design of your own choice that will deliver a payload (kg) in a circular orbit (km)

I am planning to design a three-stage rocket to launch a payload with a mass of 500 kg to an altitude of 500 km. The state-of-the-art structural mass ratios for each stage are 0.134, 0.124 and 0.152, respectively. The specific impulses are 210 m/s, 240 m/s, and 290 m/s. Since this design is similar to the SSLV, I will compare my data with actual SSLV data.

(i). Firstly we need to know the total ideal velocity required:

$$V = \sqrt{\frac{\mu}{r}}; = 8.1835 \text{ km/s} \sim 8.18 \text{ km/s}$$

$$\mathrm{V_t} = \Delta \mathrm{V}_i \,\mathrm{km/sec} + \Delta \mathrm{V_{drag}} + \Delta \mathrm{V_{gravity}} = 9.6703\,\mathrm{km/sec}$$

(ii). Specific impulse, (I_{sp}) and structural coefficient (ϵ_i) are take as follows,

	$\mathrm{I_{sp}}$ (m/s)	$\mathrm{E_i}$
Stage 1	210	0.134
Stage 2	240	0.124
Stage 3	290	0.152

(iii). We have to find out the Lagrange multiplier (η) from the total V_{t} equation

$$V_{bo} = \sum_{j=1}^{N} V_{ei} \ln \left(V_{ei}\eta - 1\right) - \ln \eta \sum_{\mathrm{i=1}}^{N} V_{\mathrm{ei}} - \sum_{\mathrm{i=1}}^{N} V_{\mathrm{ei}} \ln \left(V_{\mathrm{ei}}\varepsilon_i\right)$$

Values are substituted and η is found out iteratively and we got,

$$\eta = 0.8296$$

(iv). Now the value of η will be used to find out the empty mass and propellant mass of the vehicle, by using the formula below we will get the mass ratio (n)

$$n_i = \frac{V_{ti}\eta - 1}{V_{ti}\eta\varepsilon_i}$$

we got the mass ratios of each stage,

$$n_1 = 3.1410, n_2 = 3.9781, n_3 = 3.820$$

(v). Step masses of each stage are found using the relation,

$$m_N = \frac{n_N - 1}{1 - n_N\varepsilon_N} m_{p^l}$$

so, we got the step masses of each stage as,

$$m_1 = 98214.93, m_2 = 22701.64 \text{ kg}, m_3 = 3362.50 \text{ kg}$$

(vi). Empty mass and propellant mass of each stage are found out using $m_{ei} = \epsilon_i m_i, m_{pi} = m_i - m_{Ei}$ and tabulated

Conclusion :

Total velocity loss due to drag $(Vd) = 0.09857$ km/s

Total velocity loss due to gravity $(Vg) = 1.3881$ km/s

Total Ideal velocity required $(Vi) = 8.1835$ km/s

Total velocity required with including losses $(Vt) = 9.6703$ km/s

Lagrangian Multiplier

$$\eta = 0.8296$$

Stage/mass	First stage (kg)	second stage (kg)	Third stage (kg)
n	3.1410	3.9781	3.8200
Mi	98214.93	22701.64	3362.50
Ms	13160.80	2815.00	511.100
Mp	85051.13	19886.64	2851.40
Total mass of the vehicle in kg = 124779.082			
Overall payload ratio = 0.004007			

A vehicle design that will deliver a payload of 500 kg in a circular orbit at an altitude of 500 km is compared with an SSLV launch vehicle. The mass of the SSLV vehicle is around 120 tons, while the design of my vehicle comes in at around 125 tons. The Matlab code above yields an almost equal answer to that of the SSLV launch vehicle.

MATLAB CODE:

```
% Calculation total velocity of the vehicle including gavity and drag
   losses
clear all
clc

Isp1 = 203.87;
Isp2 = 214.67;
Isp3 = 234.45;

%me2=;
mo1 = 113000;
%mo2=;
g0 = 9.8289;

%burnout time
tb01 = 94;
tb02 = 113;
tb03 = 106;
tb0 = 314;

%Propellant mass
mp1 = 80006.53;
mp2 = 15404.11;
mp3 = 2229.16;

%Exit mass flow rate
me1 = mp1 / tb01;
me2 = mp2 / tb02;
me3 = mp3 / tb03;
%me1=481471/tb01;

h0 = 7500;
G = 6.67408e-11;
M = 5.9776e+24;
R = 6371e+3;
tbo = 531;
A1 = pi * (2 ^ 2) / 4;
%A2=pi*(4^2)/4;
A2 = 0;
Cd = 0.6;

% First stage
dV1 = zeros(1, tb01);
dV2 = zeros(1, tb01);
dV3 = zeros(1, tb01);
TV1 = zeros(1, tb01);
mf1 = zeros(1, tb01);
h1 = zeros(1, tb01);
Y1 = zeros(1, tb01);
rho1 = zeros(1, tb01);
g1 = zeros(1, tb01);
```

```matlab
D1 = zeros(1, tb01);
dydt = 90 / tb0;
mf1(1) = 113000;
dt = 1;
Y1(1) = 90;
g1(1) = G * M / (R ^ 2);
rho1(1) = 1.225;
for i = 2:1:tb01
    mf1(i) = mf1(i - 1) - me1 * dt;
    dV1(i) = (Isp1 * g0 * log(mo1 / mf1(i)));
    dV2(i) = (D1(i - 1) / mf1(i)) * dt;
    dV3(i) = g1(i - 1) * sind(Y1(i - 1)) * dt;
    TV1(i) = dV1(i) - dV2(i) - dV3(i);
    h1(i) = h1(i - 1) + ((TV1(i) * sind(Y1(i - 1))) * dt);
    g1(i) = (G * M) / ((R + h1(i)) ^ 2);
    Y1(i) = Y1(i - 1) - dydt;
    rho1(i) = rho1(1) * exp(-h1(i) / h0);
    D1(i) = 0.5 * rho1(i) * TV1(i) ^ 2 * Cd * (2 * A1 + A2);
end
drag1 = sum(dV2);
gravity_velocity_loss1 = sum(dV3);
v1 = dV1(end);

% Second stage
dV1 = zeros(1, tb02);
dV2 = zeros(1, tb02);
dV3 = zeros(1, tb02);
TV1 = zeros(1, tb02);
mf1 = zeros(1, tb02);
h1 = zeros(1, tb02);
Y1 = zeros(1, tb02);
rho1 = zeros(1, tb02);
g1 = zeros(1, tb02);
D1 = zeros(1, tb02);
dydt = 90 / tb0;
mf1(1) = 3.3845e+04;
dt = 1;
Y1(1) = 63.3439;
%g1(1)=G*M/(R^2);
g1(1) = 9.5706;
dV1(1) = 2.4158e+03;
rho1(1) = 1.3942e-05;
D1(1) = 152.2832;
h1(1) = 8.5377e+04;
for i = 2:1:tb02
    mf1(i) = mf1(i - 1) - me2 * dt;
    dV1(i) = dV1(i - 1) + (Isp2 * g0 * log(mo1 / mf1(i)));
    dV2(i) = (D1(i - 1) / mf1(i)) * dt;
    dV3(i) = g1(i - 1) * sind(Y1(i - 1)) * dt;
    TV1(i) = dV1(i) - dV2(i) - dV3(i);
    h1(i) = h1(i - 1) + ((TV1(i) * sind(Y1(i - 1))) * dt);
    g1(i) = (G * M) / ((R + h1(i)) ^ 2);
    Y1(i) = Y1(i - 1) - dydt;
    rho1(i) = rho1(1) * exp(-h1(i) / h0);
```

```
    D1(i) = 0.5 * rho1(i) * TV1(i) ^ 2 * Cd * (2 * A1 + A2);
end
drag2 = sum(dV2);
gravity_velocity_loss2 = sum(dV3);
v2 = dV1(end);

% Third stage
dV1 = zeros(1, tb03);
dV2 = zeros(1, tb03);
dV3 = zeros(1, tb03);
TV1 = zeros(1, tb03);
mf1 = zeros(1, tb03);
h1 = zeros(1, tb03);
Y1 = zeros(1, tb03);
rho1 = zeros(1, tb03);
g1 = zeros(1, tb03);
D1 = zeros(1, tb03);
dydt = 90 / tb0;
mf1(1) = 1.85777e+04;
dt = 1;
Y1(1) = 31.2420;
%g1(1)=G*M/(R^2);
g1(1) = 1.2146;
dV1(1) = 3.3421e+05;
rho1(1) = 0;
D1(1) = 152.2832;
h1(1) = 8.5377e+04;
for i = 2:1:tb03
    mf1(i) = mf1(i - 1) - me3 * dt;
    dV1(i) = dV1(i - 1) + (Isp3 * g0 * log(mo1 / mf1(i)));
    dV2(i) = (D1(i - 1) / mf1(i)) * dt;
    dV3(i) = g1(i - 1) * sind(Y1(i - 1)) * dt;
    TV1(i) = dV1(i) - dV2(i) - dV3(i);
    h1(i) = h1(i - 1) + ((TV1(i) * sind(Y1(i - 1))) * dt);
    g1(i) = (G * M) / ((R + h1(i)) ^ 2);
    Y1(i) = Y1(i - 1) - dydt;
    rho1(i) = rho1(1) * exp(-h1(i) / h0);
    D1(i) = 0.5 * rho1(i) * TV1(i) ^ 2 * Cd * (2 * A1 + A2);
end
drag3 = sum(dV2);
gravity_velocity_loss3 = sum(dV3);
v3 = dV1(end);

% Total drag and gravity loss
total_drag = (drag1 + drag2 + drag3)/1000;
disp('Total velocity loss due to drag (Vd) in km/s:');
fprintf('%.6f\n', total_drag); % Display with 6 decimal places

total_gravity_loss = (gravity_velocity_loss1 + gravity_velocity_loss2 +
   gravity_velocity_loss3)/1000;
disp('Total velocity loss due to gravity (Vg) in km/s:');
fprintf('%.6f\n', total_gravity_loss ); % Display with 6 decimal places
```

```
% calculation mass of each stage


syms x;

% Given parameters
mu3 = 500; % kg
h = 500 * 1000; % m
G = 6.67 * 10^(-11); % m^3/(kg*s^2)
Re = 6378.1 * 1000; % m
Me = 5.97 * 10^24; % kg

ve1 = 2.1;    % km/s
ve2 = 2.4; % km/s
ve3 = 2.9; % km/s

sf1 = 0.134; % structural factor of first stage
sf2 = 0.124; % structural factor of second stage
sf3 = 0.152; % structural factor of third stage


% Total Ideal velocity increment for the mission (Vt)
Vt1 = sqrt(G * Me * (Re + 2 * h) / (Re * (Re + h)))/1000;
disp('Total velocity increment (Vt) in km/s:');
fprintf('%.6f\n', Vt1 ); % Display with 6 decimal places




% Total velocity
Vt = (Vt1  + total_drag + total_gravity_loss);
disp('Total velocity increment (Vt) in km/s:');
fprintf('%.6f\n', Vt ); % Display with 6 decimal places





% Define the function
equation = ve1*log((ve1*x-1)/(ve1*sf1*x)) + ve2*log((ve2*x-1)/(ve2*sf2*x))
   + ve3*log((ve3*x-1)/(ve3*sf3*x)) - Vt;

% Solve using vpasolve
x_solution = vpasolve(equation == 0, x);

% Display the solution
disp('Solution:');
fprintf('%.6f\n', double(x_solution));

x = 0.8223

% Calculate n1, n2, and n3
```

```
n1 = (ve1 * x - 1) / (ve1 * x * sf1);
n2 = (ve2 * x - 1) / (ve2 * x * sf2);
n3 = (ve3 * x - 1) / (ve3 * x * sf3);
disp('n1:');
fprintf('%.6f\n', double(n1));
disp('n2:');
fprintf('%.6f\n', double(n2));
disp('n3:');
fprintf('%.6f\n', double(n3));

% Calculate step mass
mi3 = ((n3 - 1) / (1 - n3 * sf3)) * mu3;
mi2 = ((n2 - 1) / (1 - n2 * sf2)) * (mu3 + mi3);
mi1 = ((n1 - 1) / (1 - n1 * sf1)) * (mu3 + mi2 + mi3);

disp('Step mass in each stage in kg:');
disp('mi1:');
fprintf('%.6f\n', double(mi1));
disp('mi2:');
fprintf('%.6f\n', double(mi2));
disp('mi3:');
fprintf('%.6f\n', double(mi3));

% Calculate empty mass or structural mass in each stage
ms1 = sf1 * mi1;
ms2 = sf2 * mi2;
ms3 = sf3 * mi3;


disp('Empty or structural mass in each stage in kg:');
disp('ms1:');
fprintf('%.6f\n', double(ms1));
disp('ms2:');
fprintf('%.6f\n', double(ms2));
disp('ms3:');
fprintf('%.6f\n', double(ms3));

% Calculate propellant mass of each stage
mp1 = mi1 - ms1;
mp2 = mi2 - ms2;
mp3 = mi3 - ms3;
disp('propellent mass of each stage in kg:');
disp('mp1:');
fprintf('%.6f\n', double(mp1));
disp('mp2:');
fprintf('%.6f\n', double(mp2));
disp('mp3:');
fprintf('%.6f\n', double(mp3));


% Total mass of the vehicle
m0 = mi1 + mi2 + mi3 + mu3;
disp('Total mass of the vehicle (m0) in kg:');
fprintf('%.6f\n', double(m0));
```

```
% Overall payload ratio (alpha)
alpha = mu3 / m0;
disp('Overall payload ratio (alpha):');
fprintf('%.6f\n', double(alpha));
```

3) Design of a Rocket Propulsion System

Design a rocket propulsion system by suitably selecting a thrust value. You can consider a Hydrogen-Oxygen system, Kerosene-Oxygen system, Methane-Oxygen system, or any other choice of interest. You are expected to perform a first-level calculation. Also, size the thrust chamber and the nozzle based on a parabolic approximation. Once the preliminary design and sizing are complete, perform a performance analysis by considering chemical equilibrium conditions at the thrust chamber and nozzle exit. You are expected to conduct a comprehensive analysis of the design and performance with the help of graphs/plots.

Critically compare the results obtained with those using NASA CEA/RPA software tools.

Solution :

The primary objective of this design practice is to:

- Select a suitable value of thrust as per the mission requirements.
- Select a suitable fuel and oxidizer combination as per the mission requirements.
- Get all the necessary parameters and size the combustion chamber and nozzle.
- Do the performance analysis of propellants.

Designing

(a) Calculating chamber and Throat conditions:

- Thrust requirement of my design is 100 kN
- The propellants I want to use are liquid oxygen and liquid hydrogen, by choosing these propellants I am fixing these parameters,
  - Mixture ratio $(Mr) = 4.02$
  - Density $(\rho) = 0.28$
  - Molecular weight $(Mw) = 10.0$
  - Specific Heat Ratio $(\gamma) = 1.26$
  - Specific Impulse $(I_{sp}) = 390$
  - Characteristics velocity $(C^*) = 2432$ m/s
  - As per the molecular weight, we can get the specific gas constant $(R) = 0.8314$ kJ/kg·K

  These values are taken from the book "Aerospace Propulsion System by Thomas A. Ward, Appendix D: Rocket Propellant Table". Now the jet velocity of the vehicle can be given as,

$$V_e = I_{sp} \times g = 3825.9\text{m/s}$$

- Chamber temperature (T_c) can be calculated from the relation,

$$c^* = \sqrt{\frac{R \times T_c}{\Gamma}}$$

  where $\Gamma = \sqrt{\frac{\gamma}{2}\frac{\gamma+1}{\gamma-1}}$, $\Gamma = 0.6599$, and $T_c = 3097.94$ K

- Mass Flow rate (Assumption $P_e = P_c$),

$$\text{Thrust}(F) = \dot{m} \times V_e, \quad \text{Mass flow rate}(\dot{m}) = \frac{F}{V_e}, \quad \dot{m} = 26.1376\text{kg/s}$$

- Thrust coefficient, $C_f = \frac{V_e}{c^*}$, $\quad C_f = 1.5731$
- Chamber Pressure,

$$V_c = \sqrt{\frac{\nu}{2rRT_c}\frac{r-1}{1-\left(\frac{P_c}{P_c}\right)^{(r-1)/r}}}, \quad \text{where} \quad P_c = 7342.9313\text{kPa}$$

- Calculating Various areas and area ratios,
  - Throat area (A_t), we know that

$$\dot{m} = \frac{P_c \times A_t}{C^*}, \quad A_t = \frac{mC^*}{P_c}, \quad A_t = 8.657410^{-3}\text{m}^2$$

  now the throat diameter,

$$d_t = 0.1049\text{m} = 10.4990\text{cm}$$

- Exit area (A_e), we know that, area ratio is given as

$$\epsilon = \frac{A_e}{A_t} = \left(\frac{\Gamma^2}{2}\frac{h}{P_e/P_c}\right)^{\frac{1}{\gamma-1}} = 8.2886$$

  therefore, Exit area,

$$A_e = 0.07175\text{m}^2$$

  Exit diameter,

$$d_e = 0.3022\text{m} = 30.2249\text{cm}$$

- $\frac{A_c}{A_t}$ is taken as 4 or more, here I am taking it 4

$$\frac{A_c}{A_t} = 4$$

  therefore, chamber area,

$$A_c = 4A_t = 0.03462\text{m}^2$$

  hence, chamber diameter,

$$d_c = 0.2099\text{m} = 20.9951\text{cm}$$

- Now, we have almost all the parameters inside the thrust chamber we can go for the parabolic estimation of nozzle contour, now to summarize all the parameters

Parameter		Value
Thrust	T	100kN
Propellant	LOX – H_2	$\gamma = 1.26$ R = 831.4 J/KgK $C^* = 2432$ m/s $I_{sp} = 390$ s
Exit Velocity	V_e	3825.9 m/s
Mass flow rate	m	26.1376 kg/s
Coefficient of Thrust	C_f	1.5731
Chamber Pressure	$\overline{P_c}$	7.3MPa
Exit Area	A_e	0.07175 m^2
Throat Area	$\dot{A_t}$	0.008657 m^2
Area Ratios	A_e/A_t	8.2886
	A_c/A_t	4
Exit Radius	R_e	15.112 cm
Throat radius	R_t	5.25 cm
Chamber radius	R_c	10.4975 m

(b). Parabolic estimation of Nozzle contour

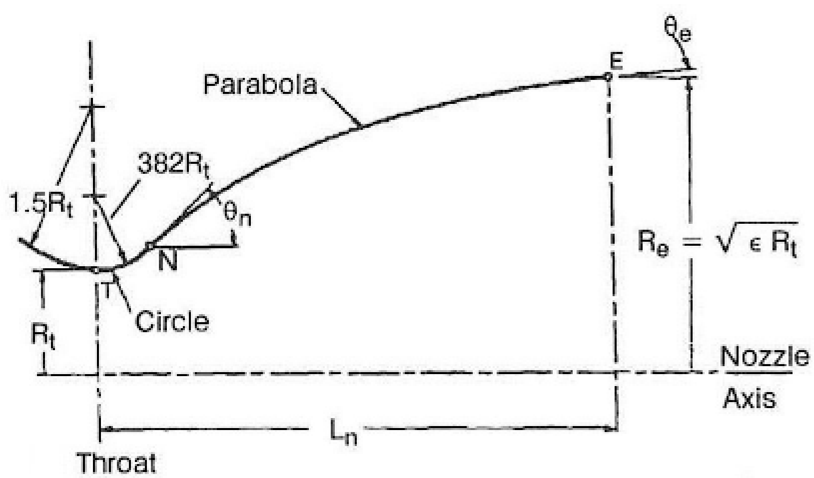


Figure 17: Bell Nozzle Contour

- First, we will find out the length of the convergent and divergent portions of the contour nozzle, using the formula,

$$L = \frac{R(\sqrt{\varepsilon} - 1) + R(\sec\alpha - 1)}{\tan\alpha}$$

This same formula is applicable for both convergent and divergent sections, we only have to use the corresponding divergent angle (α), and area ratio (ϵ). Here, I am taking.

$$\alpha_d = 15$$
$$\alpha_c = 20$$