SpaceRyde

Rocket Design Optimization Challenge

- Sepehr Salimi

PROBLEM STATEMENT

"You are tasked with the conceptual design optimization of a rocket. Using the ideal rocket equation for delta-V as your objective function, write a computational model for the mass of rocket and use numerical optimization to maximize the delta-V of the rocket subject to the following constraints and parameters" [1].

DETAILED DESIGN

OBJECTIVE

To design a rocket with maximum final velocity performance. The height, diameter, initial mass, burn time, maximum height achieved, and the time to that point will be determined.

DATA GIVEN OR KNOWN

i. Mass of the payload is 10 kg.
ii. Mass of the engine is 15 kg.
iii. The thrust of the rocket is at a constant 10,000 N.
iv. Fuel is Kerosene, oxidizer is liquid oxygen.
v. Oxidizer to fuel ratio is 2.5.
vi. The fuel and oxidizer tank operate at 2.5 MPa pressure.
vii. Specific impulse of 250 s.
viii. Rocket structure thickness of 2 mm.
ix. On Earth so gravity is 9.80665 m/s^2.

ASSUMPTIONS/LIMITATIONS/CONSTRAINTS

i. To complete this challenge within 7 days and less than 10 hours.

ii. The initial thrust to weight ratio must be greater than 3.

iii. The body tube height to diameter ratio must be greater than 4.

iv. The volume of the propellants cannot exceed their container respective volumes.

v. All components must be able to fit inside the rocket's body and fairing envelope.

vi. The fuel is kerosene, the oxidizer is liquid oxygen.

vii. The rocket has a single, pressure fed, liquid fuel engine.

viii. The fairing is in a cone shape and the body of the rocket is cylindrical.

ix. Gravity remains constant, even though the rocket is increasing its distance from the Earth.

x. Rocket is made of Aerospace-grade (7075) Aluminum Alloy with a density value of 2.81 g/cm^3 [2].

xi. Density of liquid O2, and kerosene (C12H24) gas are 810 [3] and 1141 kg/m^3 [4], respectively.

xii. Starting temperature and pressure is at the standard 25 °C and 101.325 kPa.

xiii. The specific heat of the combustion products is taken at 1300 Kelvin, based on the adiabatic flame temperature of kerosene reacting with air at 2300 Kelvin [8].

xiv. Adiabatic combustion reaction with ideal gasses in a constant-volume chamber.

xv. The heat specific ratio of combustion products in the converging-diverging nozzle is that of CO2's, at 1.28 [10].

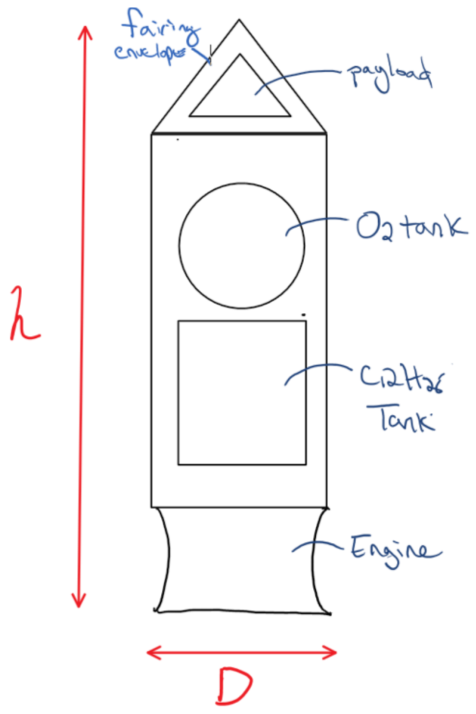
xvi. No pressure or skin friction drag is experienced by the rocket.

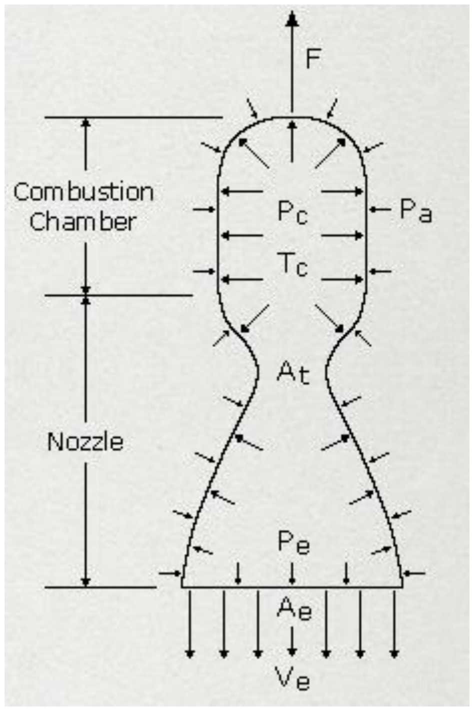
xvii. There is no rotation throughout the flight and the rocket flies straight up.

xviii. Isentropic, steady, adiabatic, 1-dimensional, compressible, flow in the converging-diverging nozzle with ideal gasses.

SKETCH

The first image is the structure of the rocket. The O2 and kerosene (C12H26) tanks are subject to change shape but in the image the O2 is spherical and the kerosene is cylindrical. The image on the right is the close up view of the converging-diverging engine [11].



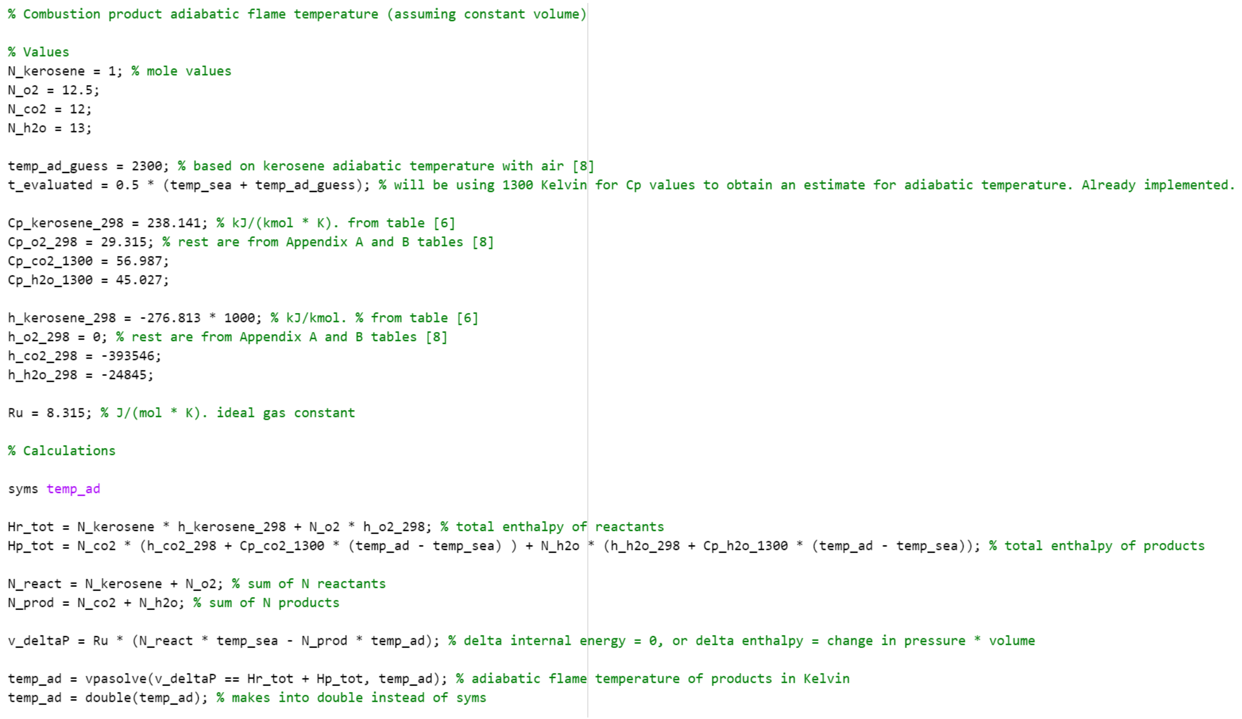


ANALYSIS

Because this challenge has unknown variables intertwined within many equations, a software must be used. Furthermore, to obtain the best results, the Genetic Algorithm within MATLAB was utilized. The code will be broken down and provided in their relevant sections. The complete code is provided in Appendix A and B. The final results are all tabulated in the Conclusion.

COMBUSTION TEMPERATURE & PRESSURE

The calculations are broken down into sections. First, the combustion product temperature, pressure, and density were calculated to obtain the exhaust temperature, pressure, and Mach #. Adiabatic and constant volume conditions were assumed to obtain the maximum temperature possible since no heat will be lost from combustion of the propellants. Below is the code to calculate the adiabatic flame temperature.



```
% Combustion product adiabatic flame temperature (assuming constant volume)

% Values
N_kerosene = 1; % mole values
N_o2 = 12.5;
N_co2 = 12;
N_h2o = 13;

temp_ad_guess = 2300; % based on kerosene adiabatic temperature with air [8]
t_evaluated = 0.5 * (temp_sea + temp_ad_guess); % will be using 1300 Kelvin for Cp values to obtain an estimate for adiabatic temperature. Already implemented.

Cp_kerosene_298 = 238.141; % kJ/(kmol * K). from table [6]
Cp_o2_298 = 29.315; % rest are from Appendix A and B tables [8]
Cp_co2_1300 = 56.987;
Cp_h2o_1300 = 45.027;

h_kerosene_298 = -276.813 * 1000; % kJ/kmol. % from table [6]
h_o2_298 = 0; % rest are from Appendix A and B tables [8]
h_co2_298 = -393546;
h_h2o_298 = -24845;

Ru = 8.315; % J/(mol * K). ideal gas constant

% Calculations

syms temp_ad

Hr_tot = N_kerosene * h_kerosene_298 + N_o2 * h_o2_298; % total enthalpy of reactants
Hp_tot = N_co2 * (h_co2_298 + Cp_co2_1300 * (temp_ad - temp_sea) ) + N_h2o * (h_h2o_298 + Cp_h2o_1300 * (temp_ad - temp_sea)); % total enthalpy of products

N_react = N_kerosene + N_o2; % sum of N reactants
N_prod = N_co2 + N_h2o; % sum of N products

v_deltaP = Ru * (N_react * temp_sea - N_prod * temp_ad); % delta internal energy = 0, or delta enthalpy = change in pressure * volume

temp_ad = vpasolve(v_deltaP == Hr_tot + Hp_tot, temp_ad); % adiabatic flame temperature of products in Kelvin
temp_ad = double(temp_ad); % makes into double instead of syms
```

The key equations that were used in this code are provided. The combustion reaction of kerosene and O2 (C12H24) [12]:

$$\mathrm{C12H24_g} + 12.5\ \mathrm{O2_l} \rightarrow 13\ \mathrm{H2O_g} + 12\ \mathrm{CO2_g} \tag{1}$$

And adiabatic, constant volume, and ideal gas equations (assuming no dissociation) [8]:

$$\sum(N_i h_i)_{react} - \sum\left(N_j h_j\right)_{prod} = V * \left(P_{prod} - P_{react}\right)$$

Where N is the number of moles, h is the specific enthalpy, V is the volume and P is the pressure.

To calculate the adiabatic flame temperature (combustion product temperature), ideal gas law was used to manipulate the equation into

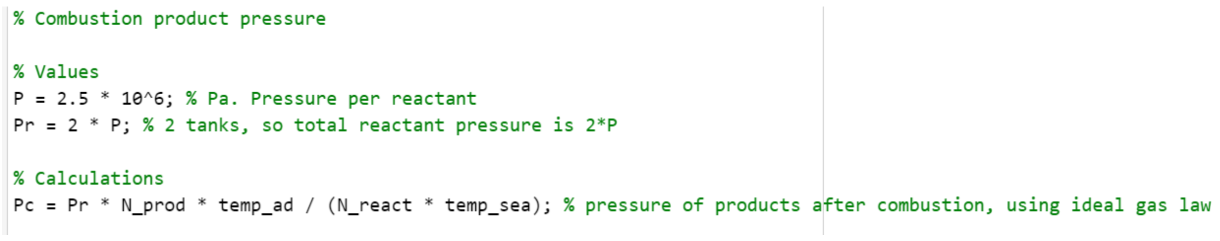
$$\sum(N_i h_i)_{react} - \sum\left(N_j\left(h_{298} + Cp_{Tad}(T_{ad} - 298)\right)\right)_{prod} = R_u\left(N_{react} * T_{initial} - N_{prod} * T_{prod}\right) \tag{3}$$

Where,

- Tad is the adiabatic combustion product temperature,
- C_p is the specific heat at this temperature,
- and R_u is the ideal gas constant.

However, due to the Cp's value being dependent on the adiabatic temperature, there are difficulties with the calculation. The 4 ways to approach this equation is to either use an existing software, through a coding software loop by storying all of the Cp values and iterating until the temperatures match, or by using an initial guess [8]. The method used for this challenge, due to time constraints, was the initial guess method by calculating the average of room temperature with the adiabatic flame temperature of kerosene reacting with air (2300 Kelvin). This value was chosen because it is tabulated in Appendix B of Stephen R. Turns' 'An introduction to combustion' [8]. The adiabatic flame temperature was calculated to be 3882 Kelvin.

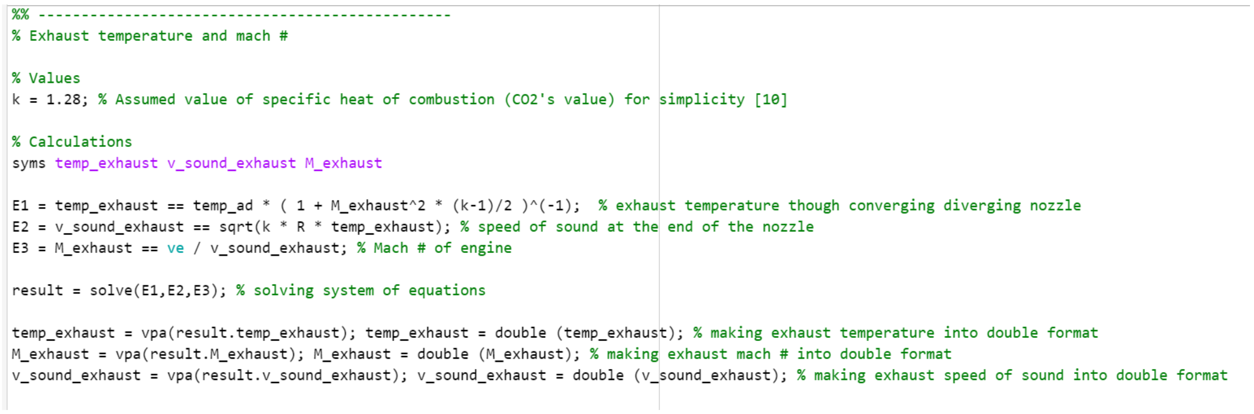
After using this and the assumed reactant pressure of 5 MPa in the ideal gas law, the combustion pressure was calculated to be 120563362.99 Pa.



```
% Combustion product pressure

% Values
P = 2.5 * 10^6; % Pa. Pressure per reactant
Pr = 2 * P; % 2 tanks, so total reactant pressure is 2*P

% Calculations
Pc = Pr * N_prod * temp_ad / (N_react * temp_sea); % pressure of products after combustion, using ideal gas law
```

EXHAUST TEMPERATURE AND MACH

With these values, the adiabatic, isentropic, steady, 1-dimensional, compressible flow in the converging-diverging nozzle was analyzed to obtain the exhaust temperature, which then was used to calculate the exhaust Mach #. Below is listed the code to accomplish this through numerical analysis in MATLAB, as well as the equations used.



```
%% -----------------------------------------------
% Exhaust temperature and mach #

% Values
k = 1.28; % Assumed value of specific heat of combustion (CO2's value) for simplicity [10]

% Calculations
syms temp_exhaust v_sound_exhaust M_exhaust

E1 = temp_exhaust == temp_ad * ( 1 + M_exhaust^2 * (k-1)/2 )^(-1);  % exhaust temperature though converging diverging nozzle
E2 = v_sound_exhaust == sqrt(k * R * temp_exhaust); % speed of sound at the end of the nozzle
E3 = M_exhaust == ve / v_sound_exhaust; % Mach # of engine

result = solve(E1,E2,E3); % solving system of equations

temp_exhaust = vpa(result.temp_exhaust); temp_exhaust = double (temp_exhaust); % making exhaust temperature into double format
M_exhaust = vpa(result.M_exhaust); M_exhaust = double (M_exhaust); % making exhaust mach # into double format
v_sound_exhaust = vpa(result.v_sound_exhaust); v_sound_exhaust = double (v_sound_exhaust); % making exhaust speed of sound into double format
```

The three equations in this code are [13]:

$$T_e = T_{ad}\left(1 + M_e^2\left(\frac{k-1}{2}\right)\right)^{-1} \tag{4}$$

$$v_{sound.exhaust} = \sqrt{k * R * T_e} \tag{5}$$

$$M_e = \frac{I_{sp} * g}{v_{sound.exhaust}} \tag{6}$$

Where,

- T_e is the exhaust temperature,
- T_{ad} is the adiabatic flame temperature,
- M_e is the exhaust Mach #
- k is the specific heat ratio (CO2's value assumed for simplicity),
- R is the specific gas constant,
- $V_{sound.exhaust}$ is the speed of sound at the exhaust temperature,
- I_{sp} is the specific impulse,
- And g is the value for gravity.

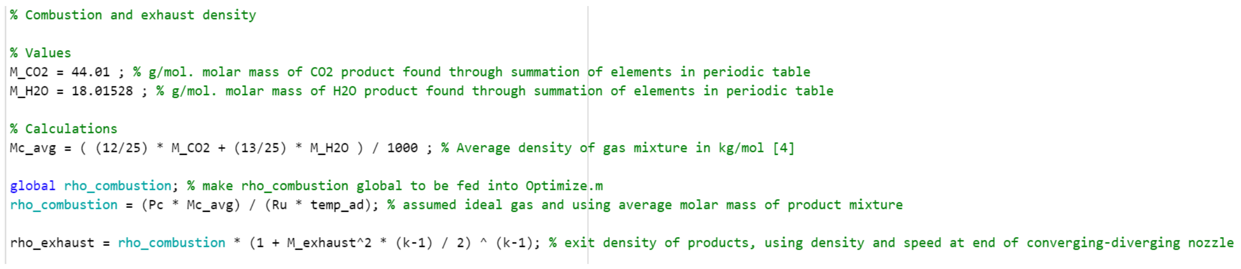
It is worth mentioning that the product of I_{sp} and g produces the exhaust velocity, v_e.

The exhaust temperature and Mach # were calculated to be 1591.52 Kelvin and 3.2063, respectively.

Exhaust Density & Pressure

Density

The combustion density was calculated using the ideal gas law, which in turn was used in the adiabatic, isentropic, converging – diverging equation to calculate the exhaust mass flow rate of the propellant. The code and equations are listed below [13].



```
% Combustion and exhaust density

% Values
M_CO2 = 44.01 ; % g/mol. molar mass of CO2 product found through summation of elements in periodic table
M_H2O = 18.01528 ; % g/mol. molar mass of H2O product found through summation of elements in periodic table

% Calculations
Mc_avg = ( (12/25) * M_CO2 + (13/25) * M_H2O ) / 1000 ; % Average density of gas mixture in kg/mol [4]

global rho_combustion; % make rho_combustion global to be fed into Optimize.m
rho_combustion = (Pc * Mc_avg) / (Ru * temp_ad); % assumed ideal gas and using average molar mass of product mixture

rho_exhaust = rho_combustion * (1 + M_exhaust^2 * (k-1) / 2) ^ (k-1); % exit density of products, using density and speed at end of converging-diverging nozzle
```

$$\rho_e = \rho_c + \left(1 + M_e^2 * \left(\frac{k-1}{2}\right)\right)^{k-1} \tag{7}$$

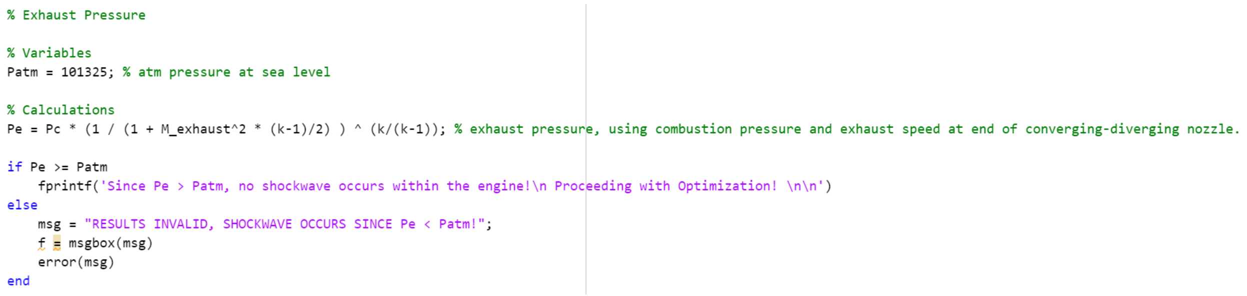
Where,

- ρ_c is the combustion density, after kerosene combusting,
- and ρ_e is the exhaust density.

The density was calculated to be 146.18 kg/m^3.

Pressure

The exhaust pressure was calculated with the isentropic flow equation listed below [13].



```
% Exhaust Pressure

% Variables
Patm = 101325; % atm pressure at sea level

% Calculations
Pe = Pc * (1 / (1 + M_exhaust^2 * (k-1)/2) ) ^ (k/(k-1)); % exhaust pressure, using combustion pressure and exhaust speed at end of converging-diverging nozzle.

if Pe >= Patm
    fprintf('Since Pe > Patm, no shockwave occurs within the engine!\n Proceeding with Optimization! \n\n')
else
    msg = "RESULTS INVALID, SHOCKWAVE OCCURS SINCE Pe < Patm!";
    f = msgbox(msg)
    error(msg)
end
```

$$P_e = P_c * \left(1 + M_e^2 * \left(\frac{k-1}{2}\right)\right)^{\frac{k}{k-1}} \tag{8}$$

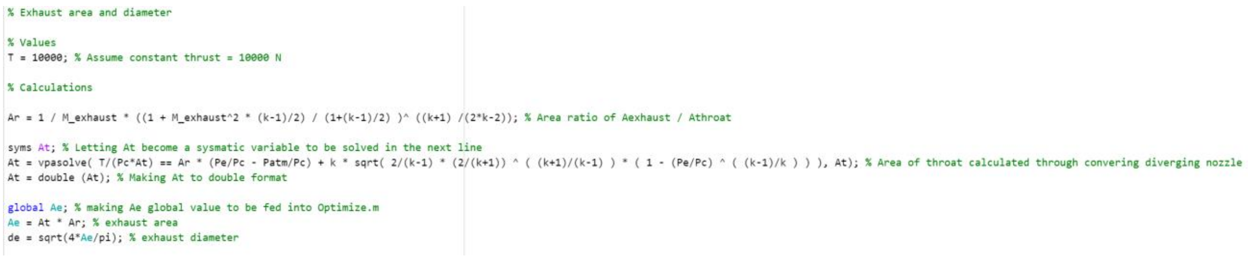
Where,

- P_e is the exhaust pressure
- And P_c is the combustion pressure calculated before.

If the exit pressure is less than the assumed atmospheric pressure, then a shockwave occurs, leading to exhaust Mach # less than 1. By previously calculating the exit velocity, with the specific impulse value provided, this is assumed not be the case. Nevertheless, in the code, there is an *if* function to stop the code should this happen. The exit pressure was later calculated to be 2045862.75 Pa, which confirms that it is larger than the atmospheric pressure.

EXHAUST AREA

To obtain the mass flow rate of the propellant, the exhaust area is required. To calculate it, two equations were used and solved numerically. The two equations and code are [13]:



```
% Exhaust area and diameter

% Values
T = 10000; % Assume constant thrust = 10000 N

% Calculations

Ar = 1 / M_exhaust * ((1 + M_exhaust^2 * (k-1)/2) / (1+(k-1)/2) )^ ((k+1) /(2*k-2)); % Area ratio of Aexhaust / Athroat

syms At; % Letting At become a sysmatic variable to be solved in the next line
At = vpasolve( T/(Pc*At) == Ar * (Pe/Pc - Patm/Pc) + k * sqrt( 2/(k-1) * (2/(k+1)) ^ ( (k+1)/(k-1) ) * ( 1 - (Pe/Pc) ^ ( (k-1)/k ) ) ), At); % Area of throat calculated through convering diverging nozzle
At = double (At); % Making At to double format

global Ae; % making Ae global value to be fed into Optimize.m
Ae = At * Ar; % exhaust area
de = sqrt(4*Ae/pi); % exhaust diameter
```

$$\frac{A_e}{A_T} = \frac{1}{M_e} * \left(\frac{1 + \frac{k-1}{2} * M_e^2}{1 + \frac{k-1}{2}} \right)^{\frac{k+1}{2(k-1)}} \tag{9}$$

$$\frac{T}{P_c * A_T} = \frac{A}{A_T} * \left(\frac{P_e}{P_c} - \frac{P_{atm}}{P_c} \right) + k * \sqrt{\frac{2}{k-1} * \left(\frac{2}{k+1} \right)^{\left(\frac{k+1}{k-1}\right)} * \left(1 - \frac{P_e}{P_c} \right)^{\frac{k-1}{k}}} \tag{10}$$

Where,

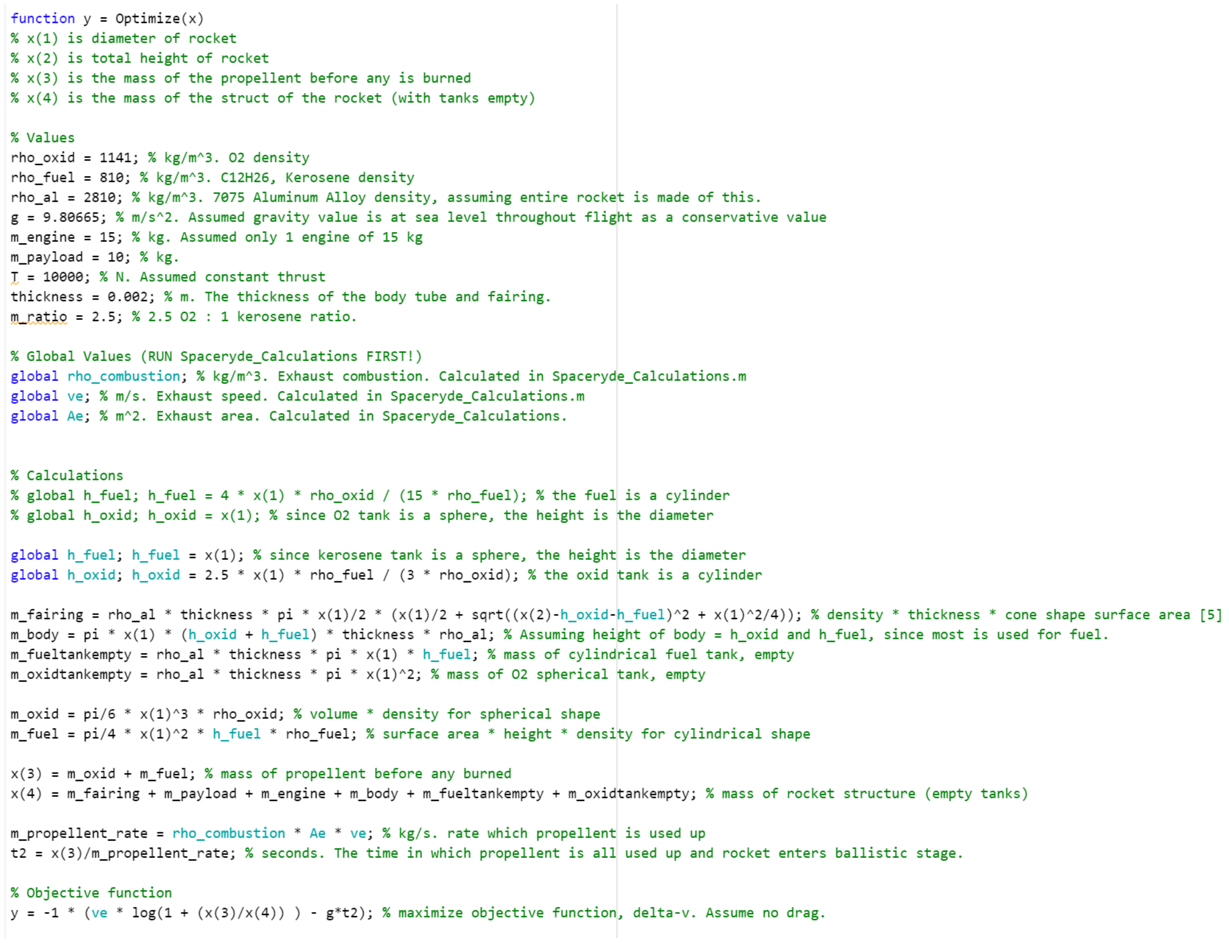
- A_e is the exhaust area
- A_T is the throat area
- A_r (in the code) is the ratio the exhaust to throat area,
- And P_{atm} is the atmospheric pressure at 101,325 kPa.

The exhaust area was calculated to be 0.000346 m^2. This seems unreasonable, giving a diameter of only 2.1 centimeter. However, for a rocket this small, it might be feasible. This can be tweaked however if hard to manufacture, by reducing the pressure of the tanks.

OPTIMIZATION

A separate function created in MATLAB was called within this script to maximize the velocity of the rocket. In this function, the masses of the rocket components were be solved with unknown variables: the diameter and height of the rocket, as well as the mass of the propellant and rocket structure. Furthermore, the mass propellant flow rate was fed into these equations using the exhaust density, velocity, and area calculated before, along with the unknown time elapsed. While running this Genetic Algorithm function, MATLAB randomly guessed and iterate these 4 variables to maximize the velocity.

On the next page is listed the code for the OpimitizeParameters.m function.



```
function y = Optimize(x)
% x(1) is diameter of rocket
% x(2) is total height of rocket
% x(3) is the mass of the propellent before any is burned
% x(4) is the mass of the struct of the rocket (with tanks empty)

% Values
rho_oxid = 1141; % kg/m^3. O2 density
rho_fuel = 810; % kg/m^3. C12H26, Kerosene density
rho_al = 2810; % kg/m^3. 7075 Aluminum Alloy density, assuming entire rocket is made of this.
g = 9.80665; % m/s^2. Assumed gravity value is at sea level throughout flight as a conservative value
m_engine = 15; % kg. Assumed only 1 engine of 15 kg
m_payload = 10; % kg.
T = 10000; % N. Assumed constant thrust
thickness = 0.002; % m. The thickness of the body tube and fairing.
m_ratio = 2.5; % 2.5 O2 : 1 kerosene ratio.

% Global Values (RUN Spaceryde_Calculations FIRST!)
global rho_combustion; % kg/m^3. Exhaust combustion. Calculated in Spaceryde_Calculations.m
global ve; % m/s. Exhaust speed. Calculated in Spaceryde_Calculations.m
global Ae; % m^2. Exhaust area. Calculated in Spaceryde_Calculations.


% Calculations
% global h_fuel; h_fuel = 4 * x(1) * rho_oxid / (15 * rho_fuel); % the fuel is a cylinder
% global h_oxid; h_oxid = x(1); % since O2 tank is a sphere, the height is the diameter

global h_fuel; h_fuel = x(1); % since kerosene tank is a sphere, the height is the diameter
global h_oxid; h_oxid = 2.5 * x(1) * rho_fuel / (3 * rho_oxid); % the oxid tank is a cylinder

m_fairing = rho_al * thickness * pi * x(1)/2 * (x(1)/2 + sqrt((x(2)-h_oxid-h_fuel)^2 + x(1)^2/4)); % density * thickness * cone shape surface area [5]
m_body = pi * x(1) * (h_oxid + h_fuel) * thickness * rho_al; % Assuming height of body = h_oxid and h_fuel, since most is used for fuel.
m_fueltankempty = rho_al * thickness * pi * x(1) * h_fuel; % mass of cylindrical fuel tank, empty
m_oxidtankempty = rho_al * thickness * pi * x(1)^2; % mass of O2 spherical tank, empty

m_oxid = pi/6 * x(1)^3 * rho_oxid; % volume * density for spherical shape
m_fuel = pi/4 * x(1)^2 * h_fuel * rho_fuel; % surface area * height * density for cylindrical shape

x(3) = m_oxid + m_fuel; % mass of propellent before any burned
x(4) = m_fairing + m_payload + m_engine + m_body + m_fueltankempty + m_oxidtankempty; % mass of rocket structure (empty tanks)

m_propellent_rate = rho_combustion * Ae * ve; % kg/s. rate which propellent is used up
t2 = x(3)/m_propellent_rate; % seconds. The time in which propellent is all used up and rocket enters ballistic stage.

% Objective function
y = -1 * (ve * log(1 + (x(3)/x(4)) ) - g*t2); % maximize objective function, delta-v. Assume no drag.
```

The equations used in this are:

$$mass_{structure} = A_{surface} * thickness * height * \rho_{structure} \tag{10}$$

$$m_{rocket} = m_{fairing} + m_{payload} + m_{engine} + m_{body} + m_{fueltankempty} + m_{oxidtankempy} \tag{11}$$

$$\frac{m_{o2}}{m_{kerosene}} = \frac{\pi * D^2 * h_{o2} * \rho_{o2}}{\frac{4}{3}\pi\left(\frac{D}{2}\right)^3 * \rho_{kerosene}} \tag{12}$$

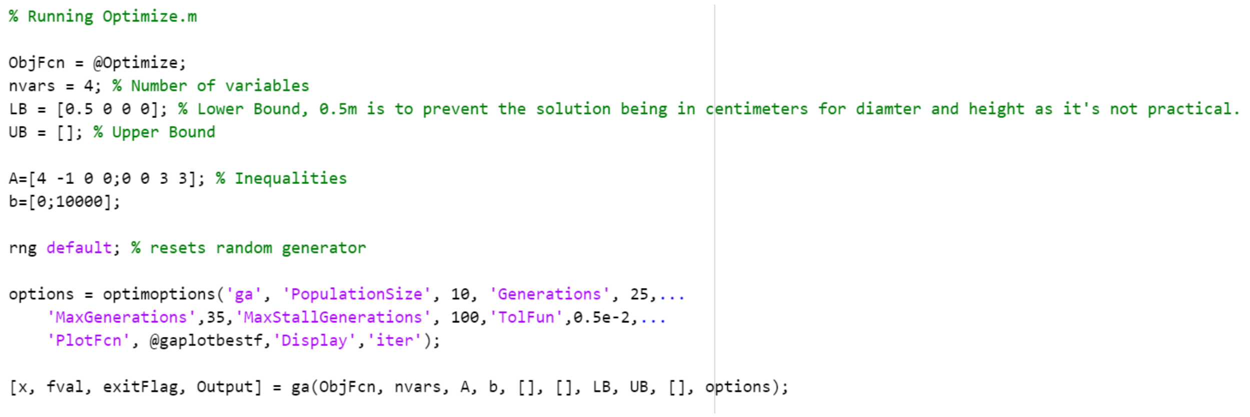
$$t_2 = \frac{mass_{oxid} + mass_{kerosene}}{\rho_{combust} * A_e * v_e} \tag{13}$$

$$-1 * \ v_2 = \ v_0 + v_e * \ln\left(1 + \frac{m_{propellant}}{m_{rocket}}\right) - g * t_2 \tag{14}$$

Where,

- Equation 10 is a general formula for calculating any part of the structure of the rocket. The sum of all of these equals the mass of the rocket with empty propellant tanks as shown in Equation 11. The fairing was assumed to be of a cone shape, the kerosene tank spherical, and the rest cylindrical.
- The height of the rocket was equal to the summation of the height of the fairing, oxygen, and kerosene tanks. The engine was ignored, assuming it would be negligible, and the height of the body was assumed to be equal to the sum of the height of the two tanks, minimizing empty space within the rocket.
- Equation 12 was used to calculate the height of the tanks. A spherical shape was chosen for the kerosene tank to alleviate the pressure caused by a gas. And since the O2 is liquid, a cylindrical shape was chosen as it is easier to manufacture and more space efficient. Furthermore, the diameter of the two tanks was set to be equal to that of the rocket, utilizing the most space possible.
- Equation 13 calculates the time it takes for the propellant to run out. At this time, the rocket enters into a ballistic with gravity as its downward acceleration.
- Equation 14 is the function that MATLAB minimized. Genetic Algorithm minimizes a function so the negative of the velocity was used to maximize the velocity. This equation is the ideal rocket equation, assuming no pressure or skin friction drag.
- The variables are self explanatory and also mentioned in the script.
- The results are tabulated in the Conclusion.

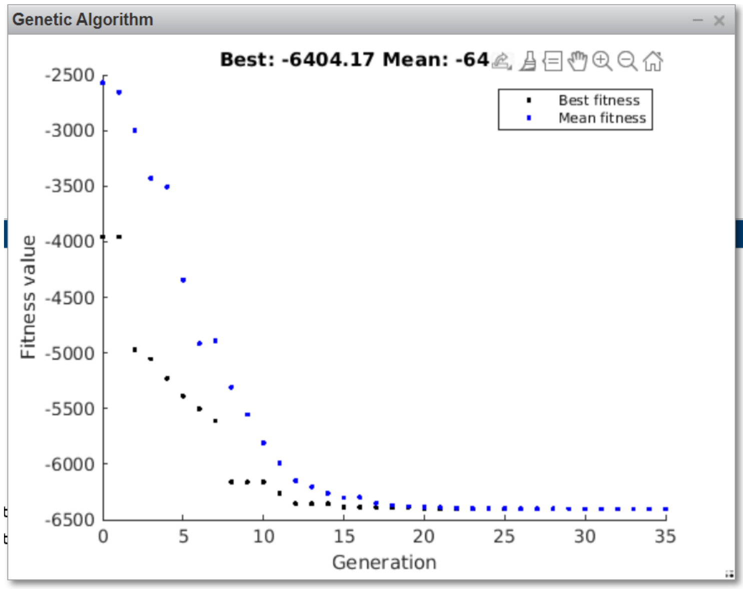
This function was called by the main script as shown:



```
% Running Optimize.m

ObjFcn = @Optimize;
nvars = 4; % Number of variables
LB = [0.5 0 0 0]; % Lower Bound, 0.5m is to prevent the solution being in centimeters for diamter and height as it's not practical.
UB = []; % Upper Bound

A=[4 -1 0 0;0 0 3 3]; % Inequalities
b=[0;10000];

rng default; % resets random generator

options = optimoptions('ga', 'PopulationSize', 10, 'Generations', 25,...
    'MaxGenerations',35,'MaxStallGenerations', 100,'TolFun',0.5e-2,...
    'PlotFcn', @gaplotbestf,'Display','iter');

[x, fval, exitFlag, Output] = ga(ObjFcn, nvars, A, b, [], [], LB, UB, [], options);
```

The constraint of the length to body ratio being greater than 4, and the thrust to initial weight ratio being greater than 3, is presented in a linear combination presented as **A** and **b** of the formula **Ax = b**. The volume constraints provided by the challenge were not implemented since there is no possible way for the volume to increase.

Lastly, below is an image to show the convergence.



Maximum Height and Its Time

After the Genetic Algorithm function, the main script proceeds to calculating the maximum height achieved by the rocket as well as how long it takes to achieve it. Below are the code and equations:

```
height2 = v_max * t2 / 2; % height at which propellent is all burned, using average velocity
height3 = height2 + v_max^2 / (2*g); % max height

syms t3 % solving for t3

t3 = vpasolve(height3 - height2 == v_max * t3 - 0.5 * g * t3^2, t3);
t3 = double(t3); % making syms into double
t3 = t3 + t2; % total time till max height
```

$$h_2 = v_{max} * \frac{t_2}{2} \tag{15}$$

$$h_3 = h_2 + \frac{v_{max}^2}{2g} \tag{16}$$

$$h_3 - h_2 = v_{max} * t_3 - g * \frac{t^3}{2} \tag{16}$$

The flight is split into two steps while assuming its path is strictly vertical. The first step which the rocket is burning propellant, and the second step where it enters ballistic mode with gravity acting as its only acceleration downward. In the first step, the height, h_2, was calculated with Equation 16. Then, the maximum height, h_3, where the maximum velocity is 0, is calculated with equation 16. These were derived from the kinematic equations, substituting acceleration out.

Lastly, equation 17 was solved numerically, to obtain the time elapsed during the ballistic period. This was added to the time beforehand to obtain the total time elapsed to reach the maximum height.

CONCLUSION

For SpaceRyde's Design Challenge Optimization challenge, a rocket's parameters were optimized using MATLAB. The maximum velocity was achieved this way, while abiding to certain restrictions. The challenge asked for a few deliverables. These are:

- Gross take-off weight: 13035 kg
- Burn time: 128.848 s
- Maximum height and its flight time: 42686.535 m, 219.49 s.

Tabulated on the next page is a summary of the design parameters and results:

Performance Results

Description	Symbol	Value	Units
Exhaust Velocity	v_e	2451.7	m/s
Exhaust Mach #	M_e	3.2063	-
Maximum Velocity	v_{max}	888.895	m/s
Height at which all propellant is burned	h_2	2400.911	m
Time till all propellant is burned	t_2	128.848	s
Maximum height	h_{max}	42686.535	m
Time at which max height is achieved	t_3	219.49	s

Mass Values

Description	Value (kg)
Mass of body	131.3252
Mass of fairing	250.2163
Mass of engine	15
Mass of kerosene gas	6427.2
Mass of kerosene tank	82.5121
Mass of O2 liquid	6035.8
Mass of O2 tank	82.5121
Mass of payload	10

Dimensions

Description	Value (m)
Diameter of rocket	2.1628
Height of rocket	15.4224
Diameter of kerosene tank	2.5334
Diameter of O2 tank	2.5334
Height of Oxygen tank	1.2789

NEXT STEPS

If given more time, the following would be implemented for improvements.

- Implement gravity value changing as change in height, though difference is minimal.
- Implement gravity and pressure/skin friction drag into ideal rocket equation.
- Implement height of engine into equations.
- Get composite rather than 7075 aluminum alloy.
- Not use $(T_{ad} + T_{sea})/2$ to get C_p values, iterate it or use an existing software.
- Better report, with digital calculations, excel to show iteration, and graphs.
- Solid model or sketch with calculated dimensions.

References

[1] 2021. Design Optimization Challenge Question. [PDF] SpaceRyde. Available at: https://www.spaceryde.com/> [Accessed 12 February 2021].

[2] Asm.matweb.com. 2021. ASM Material Data Sheet. [online] Available at: <http://asm.matweb.com/search/SpecificMaterial.asp?bassnum=MA7075T6> [Accessed 12 February 2021].

[3] En.wikipedia.org. 2021. Kerosene. [online] Available at: <https://en.wikipedia.org/wiki/Kerosene#:~:text=Kerosene%20is%20a%20low%20viscosity,10%20and%2016%20carbon%20atoms> [Accessed 12 February 2021].

[4] En.wikipedia.org. 2021. Liquid oxygen. [online] Available at: <https://en.wikipedia.org/wiki/Liquid_oxygen#:~:text=Liquid%20oxygen%20has%20a%20density,101.325%20kPa%20(760%20mmHg).> [Accessed 12 February 2021].

[5] Surface Area of a Cone - Web Formulas (web-formulas.com)

[6] Xu, R., Wang, H., Colket, M. and Edwards, T., 2015. Thermochemical Properties of Jet Fuels.

[8] Turns, S. and Haworth, D., n.d. An introduction to combustion.

[10] Engineeringtoolbox.com. 2021. Engineering ToolBox. [online] Available at: <https://www.engineeringtoolbox.com/> [Accessed 12 February 2021].

[11] Braeunig.us. 2021. Basics of Space Flight: Rocket Propulsion. [online] Available at: <http://www.braeunig.us/space/propuls.htm#Isp> [Accessed 12 February 2021].

[12] Asm.matweb.com. 2021. ASM Material Data Sheet. [online] Available at: <http://asm.matweb.com/search/SpecificMaterial.asp?bassnum=MA7075T6> [Accessed 12 February 2021].

[13] White, F. and Xue, H., n.d. Fluid mechanics.

APPENDIX A: SPACERYDE_CALCULATIONS.M

```
%%%%%%%%%%%%%%%%%%%%%%%
% SpaceRyde Design Optimization Challenge
%
% Name : Sepehr Salimi
%
% Acknowledgements:
% N/A
%
% Description:
% MATLAB code to use numerical optimization to maximize the delta-v of the
% rocket subject to various constraints and parameters
%
%%%%%%%%%%%%%%%%%%%%%%%%


clear all; clc; % reset
tic % timer


%% -------------------------------------------------
% Exhaust velocity


% Values
g = 9.80665; % m/s^2. Assumed gravity value is at sea level throughout flight as a conservative value.
Isp = 250; % Seconds. Specific impulse is time for which the thrust equals the weight of propellant
consumed, when thrust and flow rate are constant.
k_air = 1.3928; R = 287; % Values for air at sea level to calculate speed of sound. First is specific
heat, second is specific gas constant in J/kg/k.
temp_sea = 273.15 + 25; % assuming room temperature (25 deg celsuis)


% Calculations
global ve; % makes exhaust velocity into global, to be fed into Optimize.m
ve = Isp*g; % exhaust velocity in m/s
v_sound_sea = sqrt(k_air*R*temp_sea); % speed of sound at sea level


%% -------------------------------------------------
% Combustion product adiabatic flame temperature (assuming constant volume)


% Values
N_kerosene = 1; % mole values
N_o2 = 12.5;
N_co2 = 12;
N_h2o = 13;


temp_ad_guess = 2300; % based on kerosene adiabatic temperature with air [8]
t_evaluated = 0.5 * (temp_sea + temp_ad_guess); % will be using 1300 Kelvin for Cp values to obtain an
estimate for adiabatic temperature. Already implemented.


Cp_kerosene_298 = 238.141; % kJ/(kmol * K). from table [6]
Cp_o2_298 = 29.315; % rest are from Appendix A and B tables [8]
Cp_co2_1300 = 56.987;
Cp_h2o_1300 = 45.027;
```

```
h_kerosene_298 = -276.813 * 1000; % kJ/kmol. % from table [6]
h_o2_298 = 0; % rest are from Appendix A and B tables [8]
h_co2_298 = -393546;
h_h2o_298 = -24845;


Ru = 8.315; % J/(mol * K). ideal gas constant


% Calculations


syms temp_ad


Hr_tot = N_kerosene * h_kerosene_298 + N_o2 * h_o2_298; % total enthalpy of reactants
Hp_tot = N_co2 * (h_co2_298 + Cp_co2_1300 * (temp_ad - temp_sea) ) + N_h2o * (h_h2o_298 + Cp_h2o_1300 *
(temp_ad - temp_sea)); % total enthalpy of products


N_react = N_kerosene + N_o2; % sum of N reactants
N_prod = N_co2 + N_h2o; % sum of N products


v_deltaP = Ru * (N_react * temp_sea - N_prod * temp_ad); % delta internal energy = 0, or delta enthalpy =
change in pressure * volume


temp_ad = vpasolve(v_deltaP == Hr_tot + Hp_tot, temp_ad); % adiabatic flame temperature of products in
Kelvin
temp_ad = double(temp_ad); % makes into double instead of syms


%% ----------------------------------------
% Combustion product pressure


% Values
P = 2.5 * 10^6; % Pa. Pressure per reactant
Pr = 2 * P; % 2 tanks, so total reactant pressure is 2*P


% Calculations
Pc = Pr * N_prod * temp_ad / (N_react * temp_sea); % pressure of products after combustion, using ideal
gas law


%% ------------------------------------------------
% Exhaust temperature and mach #


% Values
k = 1.28; % Assumed value of specific heat of combustion (CO2's value) for simplicity [10]


% Calculations
syms temp_exhaust v_sound_exhaust M_exhaust


E1 = temp_exhaust == temp_ad * ( 1 + M_exhaust^2 * (k-1)/2 )^(-1); % exhaust temperature though converging
diverging nozzle
E2 = v_sound_exhaust == sqrt(k * R * temp_exhaust); % speed of sound at the end of the nozzle
E3 = M_exhaust == ve / v_sound_exhaust; % Mach # of engine
```

```
result = solve(E1,E2,E3); % solving system of equations


temp_exhaust = vpa(result.temp_exhaust); temp_exhaust = double (temp_exhaust); % making exhaust
temperature into double format
M_exhaust = vpa(result.M_exhaust); M_exhaust = double (M_exhaust); % making exhaust mach # into double
format
v_sound_exhaust = vpa(result.v_sound_exhaust); v_sound_exhaust = double (v_sound_exhaust); % making
exhaust speed of sound into double format




%% ------------------------------------------------
% Combustion and exhaust density


% Values
M_CO2 = 44.01 ; % g/mol. molar mass of CO2 product found through summation of elements in periodic table
M_H2O = 18.01528 ; % g/mol. molar mass of H2O product found through summation of elements in periodic
table


% Calculations
Mc_avg = ( (12/25) * M_CO2 + (13/25) * M_H2O ) / 1000 ; % Average density of gas mixture in kg/mol [4]


global rho_combustion; % make rho_combustion global to be fed into Optimize.m
rho_combustion = (Pc * Mc_avg) / (Ru * temp_ad); % assumed ideal gas and using average molar mass of
product mixture


rho_exhaust = rho_combustion * (1 + M_exhaust^2 * (k-1) / 2) ^ (k-1); % exit density of products, using
density and speed at end of converging-diverging nozzle


%% -------------------------------------------------
% Exhaust Pressure


% Variables
Patm = 101325; % atm pressure at sea level


% Calculations
Pe = Pc * (1 / (1 + M_exhaust^2 * (k-1)/2) ) ^ (k/(k-1)); % exhaust pressure, using combustion pressure
and exhaust speed at end of converging-diverging nozzle.


if Pe >= Patm
    fprintf('Since Pe > Patm, no shockwave occurs within the engine!\n Proceeding with Optimization!
\n\n')
else
    msg = "RESULTS INVALID, SHOCKWAVE OCCURS SINCE Pe < Patm!";
    f = msgbox(msg)
    error(msg)
end


%% ---------------------------------------------
% Exhaust area and diameter


% Values
T = 10000; % Assume constant thrust = 10000 N
```

```
% Calculations


Ar = 1 / M_exhaust * ((1 + M_exhaust^2 * (k-1)/2) / (1+(k-1)/2) )^ ((k+1) /(2*k-2)); % Area ratio of
Aexhaust / Athroat


syms At; % Letting At become a sysmatic variable to be solved in the next line
At = vpasolve( T/(Pc*At) == Ar * (Pe/Pc - Patm/Pc) + k * sqrt( 2/(k-1) * (2/(k+1)) ^ ( (k+1)/(k-1) ) * ( 1
- (Pe/Pc) ^ ( (k-1)/k ) ) ), At); % Area of throat calculated through convering diverging nozzle
At = double (At); % Making At to double format


global Ae; % making Ae global value to be fed into Optimize.m
Ae = At * Ar; % exhaust area
de = sqrt(4*Ae/pi); % exhaust diameter


%% -----------------------------------------------------------
% Running Optimize.m


ObjFcn = @OptimizeParameters;
nvars = 4; % Number of variables
LB = [0.5 0 0 0]; % Lower Bound, 0.5m is to prevent the solution being in centimeters for diamter and
height as it's not practical.
UB = []; % Upper Bound


A=[4 -1 0 0;0 0 3 3]; % Inequalities
b=[0;10000];


rng default; % resets random generator


options = optimoptions('ga', 'PopulationSize', 10, 'Generations', 25,...
    'MaxGenerations',35,'MaxStallGenerations', 100,'TolFun',0.5e-2,...
    'PlotFcn', @gaplotbestf,'Display','iter');


[x, fval, exitFlag, Output] = ga(ObjFcn, nvars, A, b, [], [], LB, UB, [], options);




% Printing


fprintf('The number of generations was : %d\n', Output.generations); % # of generations
fprintf('The number of function evaluations was : %d\n', Output.funccount); % # of functions evaluated


%% ----------------------------------------------------------
% Time to burn all propelent, maximum velocity, max height and its time


% First, the rocket burns propellent and reaches a height in which it no
% longer has any fuel left. This is its ballistic stage. The solution will
% be broken into two phases.
% Calculations


m_propellent_rate = rho_combustion * Ae * ve; % kg/s. rate which propellent is used up
t2 = x(3)/m_propellent_rate; % seconds. The time in which propellent is all used up and rocket enters
ballistic stage.
v_max = ve * log(1 + (x(3)/x(4)) ) - g*t2; % tells maximum velocity


m_propellent_rate = rho_combustion * Ae * ve; % kg/s. rate which propellent is used up


t2 = x(3)/m_propellent_rate; % seconds. The time in which propellent is all used up and rocket enters
ballistic stage.


height2 = v_max * t2 / 2; % height at which propellent is all burned, using average velocity
height3 = height2 + v_max^2 / (2*g); % max height


syms t3 % solving for t3


t3 = vpasolve(height3 - height2 == v_max * t3 - 0.5 * g * t3^2, t3);
t3 = double(t3); % making syms into double
t3 = t3 + t2; % total time till max height


% Print


fprintf('The maximum velocity is : %g m/s\n\n', v_max); % maximum velocity
fprintf('The diameter of the rocket was found to be %g meters, and the total height to be %g meters
\n',x(1),x(2));
fprintf('The mass of the propellents initially will weigh %g kg, and the mass of the rocket with empty
tanks weighs %g kg \n\n', x(3),x(4));
fprintf('The amount of time that the fuel is burned for is : %g seconds\n', t2); % maximum velocity
fprintf('The maximum height it reaches is : %g meters\n', height3); % maximum height
fprintf('And the time it takes for it to reach this is : %g seconds\n\n', t3); % maximum height's time


save Results


toc
```

APPENDIX B: OPTIMIZEPARAMTERS.M

```
function y = OptimizeParameters(x)
% x(1) is diameter of rocket
% x(2) is total height of rocket
% x(3) is the mass of the propellent before any is burned
% x(4) is the mass of the struct of the rocket (with tanks empty)


% Values
rho_oxid = 1141; % kg/m^3. O2 density
rho_fuel = 810; % kg/m^3. C12H26, Kerosene density
rho_al = 2810; % kg/m^3. 7075 Aluminum Alloy density, assuming entire rocket is made of this.
g = 9.80665; % m/s^2. Assumed gravity value is at sea level throughout flight as a conservative value
m_engine = 15; % kg. Assumed only 1 engine of 15 kg
m_payload = 10; % kg.
T = 10000; % N. Assumed constant thrust
thickness = 0.002; % m. The thickness of the body tube and fairing.
m_ratio = 2.5; % 2.5 O2 : 1 kerosene ratio.


% Global Values (RUN Spaceryde_Calculations FIRST!)
global rho_combustion; % kg/m^3. Exhaust combustion. Calculated in Spaceryde_Calculations.m
global ve; % m/s. Exhaust speed. Calculated in Spaceryde_Calculations.m
global Ae; % m^2. Exhaust area. Calculated in Spaceryde_Calculations.




% Calculations
% global h_fuel; h_fuel = 4 * x(1) * rho_oxid / (15 * rho_fuel); % the fuel is a cylinder
% global h_oxid; h_oxid = x(1); % since O2 tank is a sphere, the height is the diameter


h_fuel = x(1); % since kerosene tank is a sphere, the height is the diameter
h_oxid = 2.5 * x(1) * rho_fuel / (3 * rho_oxid); % the oxid tank is a cylinder


m_fairing = rho_al * thickness * pi * x(1)/2 * (x(1)/2 + sqrt((x(2)-h_oxid-h_fuel)^2 + x(1)^2/4)); %
density * thickness * cone shape surface area [5]
m_body = pi * x(1) * (h_oxid + h_fuel) * thickness * rho_al; % Assuming height of body = h_oxid and
h_fuel, since most is used for fuel.
m_fueltankempty = rho_al * thickness * pi * x(1) * h_fuel; % mass of cylindrical fuel tank, empty
m_oxidtankempty = rho_al * thickness * pi * x(1)^2; % mass of O2 spherical tank, empty


m_oxid = pi/6 * x(1)^3 * rho_oxid; % volume * density for spherical shape
m_fuel = pi/4 * x(1)^2 * h_fuel * rho_fuel; % surface area * height * density for cylindrical shape


x(3) = m_oxid + m_fuel; % mass of propellent before any burned
x(4) = m_fairing + m_payload + m_engine + m_body + m_fueltankempty + m_oxidtankempty; % mass of rocket
structure (empty tanks)


m_propellent_rate = rho_combustion * Ae * ve; % kg/s. rate which propellent is used up
t2 = x(3)/m_propellent_rate; % seconds. The time in which propellent is all used up and rocket enters
ballistic stage.


% Objective function
y = -1 * (ve * log(1 + (x(3)/x(4)) ) - g*t2); % maximize objective function, delta-v. Assume no drag.
```

```
save OptmizeResults
end
```