Joseph Hauerstein, Anderson Romero,
Mahek Shah, Abdullah Alnamlah,
Emma Perez

Submission 1

ENAE 483
September 19, 2025

1 Flow Chart

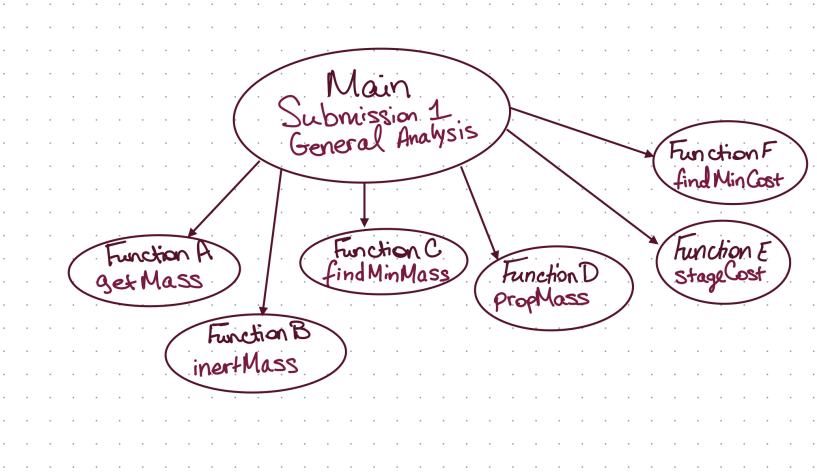

Figure 1: Flow Chart of Function Organization

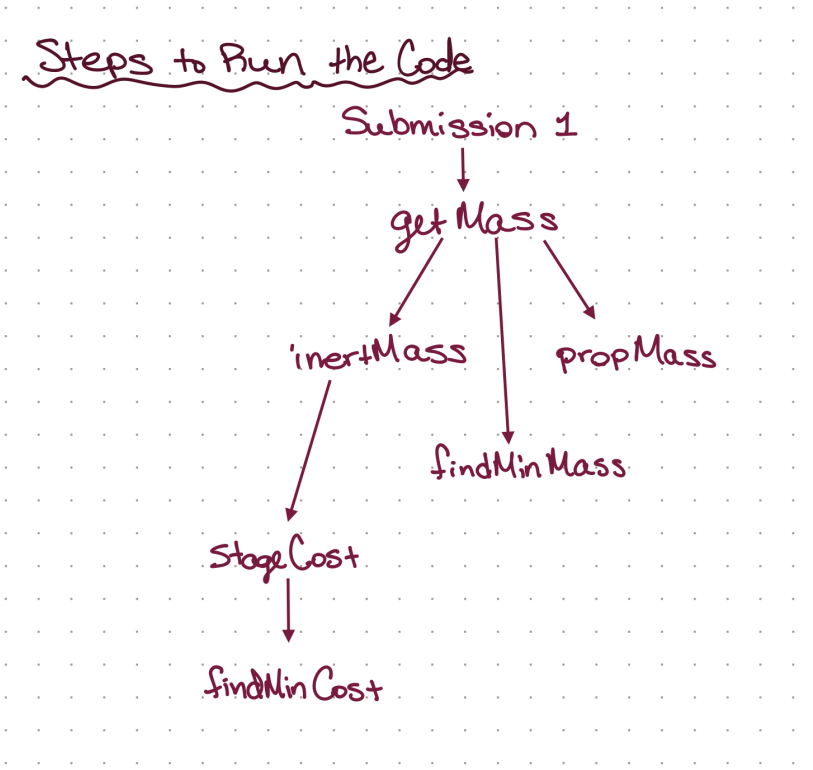

Figure 2: Flow Chart of Function Call Order

2 Code

```
% Section 1.2, Joseph Hauerstein

close all;
clear;
clc;
% Givens
delta_v = 12.3; % km/s
m_pl = 26000; % kg
delta = 0.08; % Inert mass fraction for both stages
chi = 0.2:0.01:0.8; %array
Isp1 = 327; % s, 1st stage, LOX/CH4
%       LOX/CH4 LOX/LH2 LOX/RP1 Solid   Storables
Isp2 = [327,    366,    311,    269,    285]; % s, 2nd stage
stage1Prop = "LOX/CH_4";
stage2Prop = ["LOX/CH_4", "LOX/LH_2", "LOX/RP1", "Solid", "Storables"];


for k=1:length(Isp2)
    [M01_array, M02_array, chi_array] = getMass(delta_v,m_pl,delta,chi,Isp1,
        Isp2(k));
    [m_pr1, m_pr2] = propMass(delta, M01_array, M02_array, m_pl);
    [m_in1, m_in2] = inertMass(delta, M01_array, M02_array);
    Mo = M01_array + M02_array;

    M01_array = M01_array./1000;
    M02_array = M02_array./1000;
    Mo = Mo./1000;

    [minMass, mIndex] = findMinMass(Mo);
    minMass
    costForMinMassS1 = stageCost(m_in1(mIndex));
    costForMinMassS2 = stageCost(m_in2(mIndex));
    costForMinMass = costForMinMassS1 + costForMinMassS2
    costS1 = stageCost(m_in1);
    costS2 = stageCost(m_in2);
    costTotal = costS1 + costS2;
    [minCost, cIndex] = findMinCost(costTotal);
    minCost
    massForMinCost = Mo(cIndex)

    l = round(length(chi_array)/4);
    figure(k)
    hold on;
    grid on;
    plot(chi_array(mIndex-l:mIndex+l), M02_array(mIndex-l:mIndex+l));
    plot(chi_array(mIndex-l:mIndex+l), M01_array(mIndex-l:mIndex+l));
    plot(chi_array(mIndex-l:mIndex+l), Mo(mIndex-l:mIndex+l));
    plot(chi_array(mIndex), minMass, "o");
    legend ('M_{02}', 'M_{01}', 'M_0', "M_{min}");
    titleMass = sprintf("Total Masses vs. Chi (S1: %s, S2: %s)", stage1Prop,
        stage2Prop(k));
    title(titleMass);
```

```
    ylabel('Mass (tonnes)');
    xlabel('Chi');

    figure(length(Isp2) + k)
    hold on;
    grid on;
    plot(chi_array(cIndex-1:cIndex+1), costS2(cIndex-1:cIndex+1));
    plot(chi_array(cIndex-1:cIndex+1), costS1(cIndex-1:cIndex+1));
    plot(chi_array(cIndex-1:cIndex+1), costTotal(cIndex-1:cIndex+1));
    plot(chi_array(cIndex), minCost, "o");
    legend ('Cost_2', 'Cost_1', 'Cost_0', "Cost_{min}");
    titleMass = sprintf("Cost vs. Chi (S1: %s, S2: %s)", stage1Prop,
        stage2Prop(k));
    title(titleMass);
    ylabel('Cost ($M)');
    xlabel('Chi');

end
```

```
function [M01_array, M02_array, chi_array] = getMass(delta_v,m_pl,delta,chi,
    Isp1,Isp2)
% GETMASS  Get total stage masses, M01 and M02.
%
%   [M01_array, M02_array, chi_array] = GETMASS(delta_v, m_pl, delta, chi,
    Isp1, Isp2)
%   Use the rocket equation to determine total stage masses given:
%   - delta_v: the required change in velocity
%   - m_pl: the required payload mass
%   - delta: the estimated inert mass fraction
%   - chi: the target delta V split between the first and second stage
%   - Isp1: the specific impulse of the first stage engines
%   - Isp2: the specific impulse of the second stage engines
%
%   See also PROPMASS, INERTMASS, FINDMINMASS.
    syms M02 M01
    %Initialize mass arrays
    M02_array = [];
    M01_array = [];
    chi_array = [];

    for k = 1:length(chi)
        g = 9.81/1000;%km/s^2
        %Rocket equation for stage two
        eqn2 = -Isp2*g*log((m_pl+delta*M02) / M02) - (1-chi(k))*delta_v == 0;
        soln2 = vpasolve(eqn2,M02); %Use vpa solver to find M02
        %The below loop excludes mass values that are negative and
        %populates the matrices for second stage
        if soln2 > 0 %
            try
                M02_array(end+1) = double(soln2); %Store mass values
                chi_array(end+1) = [chi(k)]; %Store corresponding chi values
            catch
            end
        end
```

```
    end

    %Repeat same steps but for first stage
    for k = 1:length(chi_array)
        g = 9.81/1000; %km/s^2
        %Rocket equation for stage one
        eqn1 = -Isp1*g*log((M02_array(k)+delta*M01) / (M02_array(k)+M01)) -
            chi_array(k)*delta_v == 0;
        soln1 = vpasolve(eqn1,M01);
        if soln1 > 0
            try
                M01_array(end+1) = double(soln1); %Store mass values
            catch
            end
        end
    end
    %The below loop scales the chi array to the mass array with lower length
    if length(M02_array) > length(M01_array)
        chi_array = chi_array(1:length(M01_array));%Scale chi to M01
        M02_array = M02_array(1:length(M01_array));%Scale M02 to M01
    elseif length(M01_array) > length(M02_array)
        chi_array = chi_array(1:length(M02_array));%Scale chi to M02
        M01_array = M01_array(1:length(M02_array));%Scale M01 to M02
    end

end
```

```
function [m_in1, m_in2] = inertMass(delta, M01_array, M02_array)
% INERTMASS  Find the inert mass given the inert mass fraction and the stage
    masses.
%
%   [m_in1, m_in2] = INERTMASS(delta, M01_array, M02_array) Finds the
%   inert mass of stage 1 (from M01_array) and the inert mass of stage 2
%   (from M02_array) using inert mass fraction delta.
%
%   See also GETMASS, PROPMASS, FINDMINMASS.
    m_in1 = delta.*M01_array;
    m_in2 = delta.*M02_array;
end
```

```
function [m_pr1, m_pr2] = propMass(delta,M01,M02, m_pl)
% PROPMASS  Find the propellant masses from the total stage masses.
%
%   [m_pr1, m_pr2] = PROPMASS(delta, M01, M02, m_pl) Find propellant masses
%   for stage 1 and 2 (m_pr1 and m_pr2), from total stage masses for stage 1
    and 2 (M01 and
%   M02), using inert mass fraction (delta) and payload mass (m_pl).
%
%   See also GETMASS, INERTMASS, FINDMINMASS.
    m_pr1 =[];
    m_pr2 = [];
    for k = 1:length(M01)
        m_in1 = delta*M01(k);
        m_pr1(k) = M01(k) - m_in1;
```

```
        m_in2 = delta*M02(k);
        m_pr2(k) = M02(k) - m_pl - m_in2;
    end
end
```

```
function [mass, index] = findMinMass(Mos)
% FINDMINMASS  Find the minimum mass given a list of total vehicle masses.
%
%   [mass, index] = FINDMINMASS(Mos) Finds the minimum mass in Mos.
%
%   See also FINDMINCOST.


    % call min function to find the min total mass, return the mass value
    % and index
    [mass,index]=min(Mos);
end
```

```
function cost = stageCost(stageInertMass)
% STAGECOST  Find the cost of a stage based on inert mass.
%
%   [cost] = STAGECOST(stageInertMass) Find the cost of a stage in millions
%   of 2025 dollars based on the stageInertMass
%
%   See also FINDMINCOST, INERTMASS.
    cost = 13.52.*stageInertMass.^0.55;
end
```

```
function [cost, index] = findMinCost(costs)
% FINDMINCOST  Find the minimum cost given a list of total vehicle costs.
%
%   [cost, index] = FINDMINCOST(costs) Finds the minimum cost in costs.
%
%   See also FINDMINMASS, STAGECOST.


    [cost,index]=min(costs);
end
```