Exported Code - Team #15 Submission 2

Note: The MATLAB functions throw an error when being published to a PDF because the function is not being called within each MATLAB file. So, there is an error in each of these files that should be ignored. The error is not thrown when properly called.

Contents


```
function [num_engines_stage1, num_engines_stage2, stage1_only_total_mass, stage2_only_total_mass, total_mass, total_height, stage1_T_to_W, stage2_T_to_W] = get_MER_
```

```
    addpath("..\vehicle_level_analysis_tool\")

    % Set thrust to weight constants
    T_to_W_first = 1.3;
    T_to_W_second = 0.76;

    % Set delta, g0, payload mass, total deltaV, and tolerance constants
    delta1 = 0.08;
    delta2 = 0.08;
    g0 = 9.81; % m/s^2
    deltaV = 12300; % m/s
    M_l = 26000; % kg
    tol = 0.01;

    % Logic to set first stage Isp and thrust
    if first_stage == "LCH4"
        stage1_Isp = 327; % s
        stage1_thrust = 2.26e6; % N
    elseif first_stage == "LH2"
        stage1_Isp = 366; % s
        stage1_thrust = 1.86e6; % N
    elseif first_stage == "RP1"
        stage1_Isp = 311; % s
        stage1_thrust = 1.92e6; % N
    elseif first_stage == "solid"
        stage1_Isp = 269; % s
        stage1_thrust = 4.5e6; % N
    elseif first_stage == "storables"
        stage1_Isp = 285; % s
        stage1_thrust = 1.75e6; % N
    end

    % Logic to set second stage Isp and thrust
    if second_stage == "LCH4"
        stage2_Isp = 327; % s
        stage2_thrust = 0.745e6; % N
    elseif second_stage == "LH2"
        stage2_Isp = 366;
        stage2_thrust = 0.099e6; % N
    elseif second_stage == "RP1"
        stage2_Isp = 311; % s
        stage2_thrust = 0.061e6; % N
    elseif second_stage == "solid"
        stage2_Isp = 269; % s
        stage2_thrust = 2.94e6; % N
    elseif second_stage == "storables"
        stage2_Isp = 285; % s
        stage2_thrust = 0.067e6; % N
    end
```

```
Not enough input arguments.

Error in get_MER_total_mass (line 17)
    if first_stage == "LCH4"
       ^^^^^^^^^^
```

SECOND STAGE COMPUTATIONS %%

Get initial guess from mass_function

```
    deltaV2_frac = deltaV*(1-X);
    r = exp(-deltaV2_frac/(g0*stage2_Isp));
    [m_in1, m_in2, m_pr1, m_pr2, m0] = mass_function(stage1_Isp, stage2_Isp, X, delta1, delta2);

    % Set propellant and total mass (with mass margin)
    M_p = m_pr2;
    M_0 = 1.3*m_in2 + M_l + M_p;

    % Call get_stage2_mass for initial guess using guess from mass_function
    stage2_total_mass = get_stage2_mass(second_stage, M_p, M_0, 1, true);

    % Compute initial guess for number of engines
    num_engines_stage2 = ceil(stage2_total_mass*g0*T_to_W_second/stage2_thrust);
```

```
% Set single engine thrust and residual that will be continuously
% checked in convergence loop
stage2_thrust_single = stage2_thrust;
residual = realmax;

if second_stage ~= "solid"
    while residual > tol

        % Set M_p and M_0 using mass margin
        M_p = stage2_total_mass*(1-r);
        M_0 = (stage2_total_mass - M_p- M_l)*1.3 + M_l + M_p;

        % Call get_stage2_mass, compute number of engines considering
        % mass margin, compute residual
        [stage2_total_mass, stage2_height] = get_stage2_mass(second_stage, M_p, M_0, num_engines_stage2, false);
        margin_stage2_total_mass = (stage2_total_mass - M_p - M_l)*1.3 + M_l + M_p;
        num_engines_stage2 = ceil(margin_stage2_total_mass*g0*T_to_W_second/stage2_thrust_single);
        residual = abs(margin_stage2_total_mass - M_0);
    end
else
    % Slightly different for solids, not an convergence loop
    % Compute mass with margin and required engines. Run
    % get_stage2_mass. Check to see if new required number of engines
    % is the same as before (most likely will be). If not, make it the
    % new number of engines that satisfy thrust to weight ratio
    margin_stage2_total_mass = (stage2_total_mass - M_l- M_p)*1.3 + M_l + M_p;
    num_engines_required = ceil(margin_stage2_total_mass*g0*T_to_W_second/stage2_thrust_single);
    [stage2_total_mass, stage2_height] = get_stage2_mass(second_stage, M_p, M_0, num_engines_required, false);
    margin_stage2_total_mass = (stage2_total_mass - M_l - M_p)*1.3 + M_l + M_p;
    num_engines_required_recompued = ceil(margin_stage2_total_mass*g0*T_to_W_second/stage2_thrust_single);
    if ceil(num_engines_required) ~= ceil(num_engines_required_recompued)
        num_engines_stage2 = ceil(num_engines_required_recompued);
    else
        num_engines_stage2 = ceil(num_engines_required);
    end
end
```

FIRST STAGE COMPUTATIONS %%

Get initial guess from mass_function

```
deltaV1_frac = deltaV*X;
r = exp(-deltaV1_frac/(g0*stage1_Isp));
[m_in1, m_in2, m_pr1, m_pr2, m0] = mass_function(stage1_Isp, stage2_Isp, X, delta1, delta2);

% Set propellant and total mass (with mass margin)
M_p = m_pr1;
M_0 = (m0 - M_l - m_pr1)*1.3 + M_l + m_pr1;

% Call get_stage1_mass for initial guess using guess from mass_function
stage1_total_mass = get_stage1_mass(first_stage, M_p, M_0, stage2_total_mass, 1, true);

% Compute initial guess for number of engines
num_engines_stage1 = stage1_total_mass*g0*T_to_W_first/stage1_thrust;

% Set single engine thrust and residual that will be continuously
stage1_thrust_single = stage1_thrust;
residual = realmax;

if first_stage ~= "solid"
    while residual > tol

        % Set M_p and M_0 using mass margin
        M_p = stage1_total_mass*(1-r);
        M_0 = (stage1_total_mass - M_p - margin_stage2_total_mass)*1.3 + margin_stage2_total_mass + M_p;

        % Call get_stage1_mass, compute number of engines considering
        % mass margin and stage2 mass, compute residual
        [stage1_total_mass, stage1_height] = get_stage1_mass(first_stage, M_p, M_0, margin_stage2_total_mass, num_engines_stage1, false);
        margin_stage1_total_mass = (stage1_total_mass - M_p - margin_stage2_total_mass)*1.3 + margin_stage2_total_mass + M_p;
        num_engines_stage1 = ceil(margin_stage1_total_mass*g0*T_to_W_first/stage1_thrust_single);
        residual = abs(margin_stage1_total_mass - M_0);
    end
else
    % Slightly different for solids, not an convergence loop
    % Compute mass with margin and required engines. Run
    % get_stage2_mass. Check to see if new required number of engines
    % is the same as before (most likely will be). If not, make it the
    % new number of engines that satisfy thrust to weight ratio
    margin_stage1_total_mass = (stage1_total_mass - M_p - margin_stage2_total_mass)*1.3 + margin_stage2_total_mass + M_p;
    num_engines_required = ceil(margin_stage1_total_mass*g0*T_to_W_first/stage1_thrust_single);
    [stage1_total_mass, stage1_height] = get_stage1_mass(first_stage, M_p, M_0, stage2_total_mass, num_engines_required, false);
    margin_stage1_total_mass = (stage1_total_mass - M_p - margin_stage2_total_mass)*1.3 + margin_stage2_total_mass + M_p;
    num_engines_required_recompued = ceil(margin_stage1_total_mass*g0*T_to_W_first/stage1_thrust_single);
    if ceil(num_engines_required) ~= ceil(num_engines_required_recompued)
        num_engines_stage1 = ceil(num_engines_required_recompued);
```

```
        else
            num_engines_stage1 = ceil(num_engines_required);
        end
    end

    % Compute total height
    total_height = stage1_height + stage2_height;

    % Compute avionics mass and set as workspace variable
    mass_avionics = 10*stage1_total_mass^(0.361);
    assignin('base', 'mass_avionics', mass_avionics);

    % Compute stage only masses
    stage1_only_total_mass = margin_stage1_total_mass - margin_stage2_total_mass - M_l + mass_avionics*1.3;
    stage2_only_total_mass = margin_stage2_total_mass - M_l + mass_avionics*1.3;

    % Compute total masses
    stage1_total_mass = margin_stage1_total_mass + mass_avionics*1.3;
    stage2_total_mass = margin_stage2_total_mass + mass_avionics*1.3;
    total_mass = stage1_total_mass;

    % Get number of engines per stage
    num_engines_stage1 = ceil(num_engines_stage1);
    num_engines_stage2 = ceil(num_engines_stage2);

    % Output thrust to weight to ensure it meets requirement
    stage2_T_to_W = num_engines_stage2*stage2_thrust_single/g0/stage2_total_mass;
    stage1_T_to_W = num_engines_stage1*stage1_thrust_single/g0/stage1_total_mass;
```

```
end
```

Published with MATLAB® R2024b

```
function [stage1_total_mass, stage1_height] = get_stage1_mass(first_stage, M_p, M_0, stage2_total_mass, num_engines, init)

    % Set density constants
    rho_LH2 = 71;
    rho_LOX = 1140;
    rho_RP1 = 820;
    rho_LCH4 = 423;
    rho_solid = 1680;
    rho_N2O4 = 1442;
    rho_UDMH = 791;

    % Set tank constants
    radius = 6.4; % m
    cap_height = 1; % m
    payload_cone_height = 10; % m
    payload_cyl_height = 10; % m
    engine_space = 3; % m

    % Constants that depend on propellant choice
    if first_stage == "LCH4"
        stage1_ratio = 3.6;
        stage1_oxidizer_rho = rho_LOX;
        stage1_fuel_rho = rho_LCH4;
        stage1_thrust_single = 2.26e6; % N
        stage1_nozzle_exp = 34.34;
        chamber_pressure_1 = 35.16e6; % Pa
    elseif first_stage == "LH2"
        stage1_ratio = 6.03;
        stage1_oxidizer_rho = rho_LOX;
        stage1_fuel_rho = rho_LH2;
        stage1_thrust_single = 1.86e6; % N
        stage1_nozzle_exp = 78;
        chamber_pressure_1 = 20.64e6; % Pa
    elseif first_stage == "RP1"
        stage1_ratio = 2.72;
        stage1_oxidizer_rho = rho_LOX;
        stage1_fuel_rho = rho_RP1;
        stage1_thrust_single = 1.92e6; % N
        stage1_nozzle_exp = 37;
        chamber_pressure_1 = 25.8e6; % Pa
    elseif first_stage == "solid"
        stage1_thrust_single = 4.5e6; % N
        stage1_nozzle_exp = 16;
        chamber_pressure_1 = 10.5e6; % Pa
    elseif first_stage == "storables"
        stage1_ratio = 2.67;
        stage1_oxidizer_rho = rho_N2O4;
        stage1_fuel_rho = rho_UDMH;
        stage1_thrust_single = 1.75e6; % N
        stage1_nozzle_exp = 26.2;
        chamber_pressure_1 = 15.7e6; % Pa
    end

    % Set thrust according to number of engines
    if init
        stage1_thrust = stage1_thrust_single;
    else
        stage1_thrust = stage1_thrust_single*num_engines;
    end

    % Compute stage1 tank mass
    if first_stage == "solid"
        solid_volume = M_p/rho_solid;
        stage1_tank_mass = 12.16*solid_volume;
    else
        mass_split = M_p/(stage1_ratio+1);
        mass_oxidizer = stage1_ratio*mass_split;
        mass_fuel = mass_split;
        volume_oxidizer = mass_oxidizer/stage1_oxidizer_rho;
        volume_fuel = mass_fuel/stage1_fuel_rho;
        if first_stage == "LH2"
            stage1_tank_mass = 12.16*volume_oxidizer + 9.09*volume_fuel;
        else
            stage1_tank_mass = 12.16*(volume_oxidizer + volume_fuel);
        end
    end

    % Compute stage1 tank volume assuming cylinder and two sphere caps 1m
    % tall each
    if first_stage ~= "solid"
        ox_cap_vol = 2*(pi*cap_height)*(3*radius^2 + cap_height^2)/6;
        ox_cyl_vol = volume_oxidizer - ox_cap_vol;
        ox_cyl_height = ox_cyl_vol/(pi*radius^2);
        ox_cap_surf_area = 2*(pi*(radius^2 + cap_height^2));
        ox_cyl_surf_area = 2*pi*radius*ox_cyl_height;
```

```
        fuel_cap_vol = 2*(pi*cap_height)*(3*radius^2 + cap_height^2)/6;
        fuel_cyl_vol = volume_fuel - fuel_cap_vol;
        fuel_cyl_height = fuel_cyl_vol/(pi*radius^2);
        fuel_cap_surf_area = 2*(pi*(radius^2 + cap_height^2));
        fuel_cyl_surf_area = 2*pi*radius*fuel_cyl_height;
    end

    % Compute insulation from tank volume, edge cases for storables and
    % solids
    if first_stage ~= "solid" && first_stage ~= "storables"
        LOX_stage1_insulation_mass = 1.123*(ox_cap_surf_area + ox_cyl_surf_area);
        if first_stage == "LH2"
            LH2_stage1_insulation_mass = 2.88*(fuel_cap_surf_area + fuel_cyl_surf_area);
            stage1_insulation_mass = LOX_stage1_insulation_mass + LH2_stage1_insulation_mass;
        elseif first_stage == "LCH4"
            LCH4_stage1_insulation_mass = 1.123*(fuel_cap_surf_area + fuel_cyl_surf_area);
            stage1_insulation_mass = LOX_stage1_insulation_mass + LCH4_stage1_insulation_mass;
        else
            stage1_insulation_mass = LOX_stage1_insulation_mass;
        end
    elseif first_stage == "solid"
        stage1_insulation_mass = 0;
        solid_cap_vol = 2*(pi*cap_height)*(3*radius^2 + cap_height^2)/6;
        solid_cyl_vol = solid_volume - solid_cap_vol;
        solid_cyl_height = solid_cyl_vol/(pi*radius^2);
    else
        stage1_insulation_mass = 0;
    end

    % Set engine and casing mass, dependent on propellant
    if first_stage ~= "solid"
        stage1_engine_mass = 7.81e-4*stage1_thrust + 3.37e-5*stage1_thrust*sqrt(stage1_nozzle_exp) + 59;
        stage1_casing_mass = 0;
    else
        stage1_engine_mass = 0;
        stage1_casing_mass = 0.135*M_p;
    end

    % Compute payload and aft fairing areas
    interstage_fairing_area = 2*pi*radius*(engine_space + cap_height);
    aft2_fairing_area = 2*pi*radius*(engine_space + cap_height);

    interstage_fairing_mass = 4.95*interstage_fairing_area^(1.15);
    stage1_aft_fairing_mass = 4.95*aft2_fairing_area^(1.15);

    % Compute intertank fairing mass and overall height dependent on propellant
    if first_stage ~= "solid"
        intertank2_fairing_area = 2*pi*radius*(2*cap_height);
        stage1_intertank_fairing_mass = 4.95*intertank2_fairing_area^(1.15);
        stage1_height = payload_cone_height + payload_cyl_height + 4*cap_height + ox_cyl_height + fuel_cyl_height + engine_space;
    else
        stage1_intertank_fairing_mass = 0;
        stage1_height = payload_cone_height + payload_cyl_height + 2*cap_height + solid_cyl_height + engine_space;
    end

    % Compute wiring, thrust structure, and gimbals masses
    stage1_mass_wiring = 1.058*sqrt(M_0)*stage1_height^(0.25);

    stage1_mass_thrust_struct = 2.25e-4*stage1_thrust;

    stage1_mass_gimbals = 237.8*(stage1_thrust/chamber_pressure_1)^(0.9375);

    % Compute total mass
    stage1_total_mass = M_p + stage1_mass_wiring + stage1_tank_mass + stage1_insulation_mass + stage1_engine_mass + stage1_mass_thrust_struct + stage1_casing_mass +

    % Assign workspace variables
    assignin('base', 'stage1_propellant_mass', M_p);
    assignin('base', 'stage1_tank_mass', stage1_tank_mass);
    assignin('base', 'stage1_mass_wiring', stage1_mass_wiring);
    assignin('base', 'stage1_insulation_mass', stage1_insulation_mass);
    assignin('base', 'stage1_engine_mass', stage1_engine_mass);
    assignin('base', 'stage1_mass_thrust_struct', stage1_mass_thrust_struct);
    assignin('base', 'stage1_casing_mass', stage1_casing_mass);
    assignin('base', 'stage1_mass_gimbals', stage1_mass_gimbals);
    assignin('base', 'interstage_fairing_mass', interstage_fairing_mass);
    assignin('base', 'stage1_intertank_fairing_mass', stage1_intertank_fairing_mass);
    assignin('base', 'stage1_aft_fairing_mass', stage1_aft_fairing_mass);

end
```

```
Not enough input arguments.

Error in get_stage1_mass (line 20)
```

```
    if first_stage == "LCH4"
       ^^^^^^^^^^^
```

Published with MATLAB® R2024b

```
function [stage2_total_mass, stage2_height] = get_stage2_mass(second_stage, M_p, M_0, num_engines, init)

    % Set density constants
    rho_LH2 = 71;
    rho_LOX = 1140;
    rho_RP1 = 820;
    rho_LCH4 = 423;
    rho_solid = 1680;
    rho_N2O4 = 1442;
    rho_UDMH = 791;

    % Set tank constants
    M_l = 26000; % kg
    radius = 6.4; % m
    cap_height = 1; % m
    payload_cone_height = 10; % m
    payload_cyl_height = 10; % m
    engine_space = 3; % m

    % Constants that depend on propellant choice
    if second_stage == "LCH4"
        stage2_ratio = 3.6;
        stage2_oxidizer_rho = rho_LOX;
        stage2_fuel_rho = rho_LCH4;
        stage2_thrust_single = 0.745e6; % N
        stage2_nozzle_exp = 45;
        chamber_pressure_2 = 10.1e6; % Pa
    elseif second_stage == "LH2"
        stage2_ratio = 6.03;
        stage2_oxidizer_rho = rho_LOX;
        stage2_fuel_rho = rho_LH2;
        stage2_thrust_single = 0.099e6; % N
        stage2_nozzle_exp = 84;
        chamber_pressure_2 = 4.2e6; % Pa
    elseif second_stage == "RP1"
        stage2_ratio = 2.72;
        stage2_oxidizer_rho = rho_LOX;
        stage2_fuel_rho = rho_RP1;
        stage2_thrust_single = 0.061e6; % N
        stage2_nozzle_exp = 14.5;
        chamber_pressure_2 = 6.77e6; % Pa
    elseif second_stage == "solid"
        stage2_thrust_single = 2.94e6; % N
        stage2_nozzle_exp = 56;
        chamber_pressure_2 = 5e6; % Pa
    elseif second_stage == "storables"
        stage2_ratio = 2.67;
        stage2_oxidizer_rho = rho_N2O4;
        stage2_fuel_rho = rho_UDMH;
        stage2_thrust_single = 0.067e6; % N
        stage2_nozzle_exp = 81.3;
        chamber_pressure_2 = 14.7e6; % Pa
    end

    % Set thrust according to number of engines
    if init
        stage2_thrust = stage2_thrust_single;
    else
        stage2_thrust = stage2_thrust_single*num_engines;
    end

    % Compute stage2 tank mass
    if second_stage == "solid"
        solid_volume = M_p/rho_solid;
        stage2_tank_mass = 12.16*solid_volume;
    else
        mass_split = M_p/(stage2_ratio+1);
        mass_oxidizer = stage2_ratio*mass_split;
        mass_fuel = mass_split;
        volume_oxidizer = mass_oxidizer/stage2_oxidizer_rho;
        volume_fuel = mass_fuel/stage2_fuel_rho;
        if second_stage == "LH2"
            stage2_tank_mass = 12.16*volume_oxidizer + 9.09*volume_fuel;
        else
            stage2_tank_mass = 12.16*(volume_oxidizer + volume_fuel);
        end
    end

    % Compute stage2 tank volume assuming cylinder and two sphere caps 1m
    % tall each
    if second_stage ~= "solid"
        ox_cap_vol = 2*(pi*cap_height)*(3*radius^2 + cap_height^2)/6;
        ox_cyl_vol = volume_oxidizer - ox_cap_vol;
        ox_cyl_height = ox_cyl_vol/(pi*radius^2);
        ox_cap_surf_area = 2*(pi*(radius^2 + cap_height^2));
```

```
        ox_cyl_surf_area = 2*pi*radius*ox_cyl_height;

        fuel_cap_vol = 2*(pi*cap_height)*(3*radius^2 + cap_height^2)/6;
        fuel_cyl_vol = volume_fuel - fuel_cap_vol;
        fuel_cyl_height = fuel_cyl_vol/(pi*radius^2);
        fuel_cap_surf_area = 2*(pi*(radius^2 + cap_height^2));
        fuel_cyl_surf_area = 2*pi*radius*fuel_cyl_height;
    end

    % Compute insulation from tank volume, edge cases for storables and
    % solids
    if second_stage ~= "solid" && second_stage ~= "storables"
        LOX_stage2_insulation_mass = 1.123*(ox_cap_surf_area + ox_cyl_surf_area);
        if second_stage == "LH2"
            LH2_stage2_insulation_mass = 2.88*(fuel_cap_surf_area + fuel_cyl_surf_area);
            stage2_insulation_mass = LOX_stage2_insulation_mass + LH2_stage2_insulation_mass;
        elseif second_stage == "LCH4"
            LCH4_stage2_insulation_mass = 1.123*(fuel_cap_surf_area + fuel_cyl_surf_area);
            stage2_insulation_mass = LOX_stage2_insulation_mass + LCH4_stage2_insulation_mass;
        else
            stage2_insulation_mass = LOX_stage2_insulation_mass;
        end
    elseif second_stage == "solid"
        stage2_insulation_mass = 0;
        solid_cap_vol = 2*(pi*cap_height)*(3*radius^2 + cap_height^2)/6;
        solid_cyl_vol = solid_volume - solid_cap_vol;
        solid_cyl_height = solid_cyl_vol/(pi*radius^2);
    else
        stage2_insulation_mass = 0;
    end

    % Set engine and casing mass, dependent on propellant
    if second_stage ~= "solid"
        stage2_engine_mass = 7.81e-4*stage2_thrust + 3.37e-5*stage2_thrust*sqrt(stage2_nozzle_exp) + 59;
        stage2_casing_mass = 0;
    else
        stage2_engine_mass = 0;
        stage2_casing_mass = 0.135*M_p;
    end

    % Compute payload and aft fairing areas
    payload_fairing_area = pi*radius*sqrt(radius^2 + payload_cone_height^2) + 2*pi*radius*payload_cyl_height;
    aft2_fairing_area = 2*pi*radius*(engine_space + cap_height);

    payload_fairing_mass = 4.95*payload_fairing_area^(1.15);
    stage2_aft_fairing_mass = 4.95*aft2_fairing_area^(1.15);

    % Compute intertank fairing mass and overall height dependent on propellant
    if second_stage ~= "solid"
        intertank2_fairing_area = 2*pi*radius*(2*cap_height);
        stage2_intertank2_fairing_mass = 4.95*intertank2_fairing_area^(1.15);
        stage2_height = payload_cone_height + payload_cyl_height + 4*cap_height + ox_cyl_height + fuel_cyl_height + engine_space;
    else
        stage2_intertank2_fairing_mass = 0;
        stage2_height = payload_cone_height + payload_cyl_height + 2*cap_height + solid_cyl_height + engine_space;
    end

    % Compute wiring, thrust structure, and gimbals masses
    stage2_mass_wiring = 1.058*sqrt(M_0)*stage2_height^(0.25);

    stage2_mass_thrust_struct = 2.25e-4*stage2_thrust;

    stage2_mass_gimbals = 237.8*(stage2_thrust/chamber_pressure_2)^(0.9375);

    % Compute total mass
    stage2_total_mass = M_p + stage2_mass_wiring + stage2_tank_mass + stage2_insulation_mass + stage2_engine_mass + stage2_mass_thrust_struct + stage2_casing_mass +

    % Assign workspace variables
    assignin('base', 'stage2_propellant_mass', M_p);
    assignin('base', 'stage2_tank_mass', stage2_tank_mass);
    assignin('base', 'stage2_mass_wiring', stage2_mass_wiring);
    assignin('base', 'stage2_insulation_mass', stage2_insulation_mass);
    assignin('base', 'stage2_engine_mass', stage2_engine_mass);
    assignin('base', 'stage2_mass_thrust_struct', stage2_mass_thrust_struct);
    assignin('base', 'stage2_casing_mass', stage2_casing_mass);
    assignin('base', 'stage2_mass_gimbals', stage2_mass_gimbals);
    assignin('base', 'payload_fairing_mass', payload_fairing_mass);
    assignin('base', 'stage2_intertank2_fairing_mass', stage2_intertank2_fairing_mass);
    assignin('base', 'stage2_aft_fairing_mass', stage2_aft_fairing_mass);

end
```

```
Not enough input arguments.

Error in get_stage2_mass (line 21)
```

```
    if second_stage == "LCH4"
       ^^^^^^^^^^^^
```

Published with MATLAB® R2024b