Iterative Sizing Algorithm

- Calculating the Takeoff Weights for Aircrafts Using

Function description

```
function [W_gross, T, Wempty_W0, W0_new] = takeoff_weight_sizing(W_payload, W_crew,...
    range, V_mach, engineType, unitType, category, LDratio, loiterTime1, loiterTime2)
```

INPUTS

1. W_payload: The weight of the payload [lb]
2. W_crew: The weight of the crew [lb]
3. range: The flight range [nautical miles]
4. V_mach: The mach number of the aircraft [Mach]
5. engineType: The type of engine (e.g. pure turbojet, turboprop, etc.)
6. unitType: Metric or English
7. category: The category of the aircraft (e.g. single engine, military cargo, etc.)
8. LDratio: The lift and drag ratio
9. loiterTime1 & loiterTime2: The time of loitering [hr]

OUTPUTS

1. W_gross: The estimated gross weight of the designed aircraft
2. T: The table with the guessed gross weight, Wempty and Wgross ratio, Wempty, and calculated Wgross
3. Wempty_W0: The vector with the outputs of the Wempty and Wgross ratio
4. W0_new: The vector of the outputs of the calculated Wgross

Preparation

Convert based on the unit input and also designate constants based on the engine type that is given.

The constant to determine is the $\text{Specific Fuel Consumption} \equiv C$ for cruise and loiter

```
%By default if the W_crew is undetermined or zero 800 will be the weight of crew
if W_crew == 0
    W_crew = 800; %[lb]
end

if unitType == "metric"
    %Converting the input values to metric
    W_payload = W_payload * 0.4536; %[lb] to [kg]
    W_crew = W_crew * 0.4536; %[lb] to [kg]
    range = range * 1852; %[nmi] to [m]
    %Converting inputs to proper units
    loiterTime1 = loiterTime1 * 3600; %[hr] to [s]
    loiterTime2 = loiterTime2 * 3600; %[hr] to [s]
    V = V_mach * 343; %[m/s] (approximate to sea level)
```

```
%Assigning proper specific fuel consumption values for each engine type
if engineType == "pure turbojet"
    C_cruise = 25.5; %Specific fuel consumption for cruise [mg/Ns]
    C_loiter = 22.7; %Specific fuel consumption for loiter [mg/Ns]
    %Convert to proper units
    C_cruise = C_cruise * 10^(-6); %[kg/Ns]
    C_loiter = C_loiter * 10^(-6); %[kg/Ns]
elseif engineType == "low-bypass turbofan"
    C_cruise = 22.7; %Specific fuel consumption for cruise [mg/Ns]
    C_loiter = 19.8; %Specific fuel consumption for loiter [mg/Ns]
    %Convert to proper units
    C_cruise = C_cruise * 10^(-6); %[kg/Ns]
    C_loiter = C_loiter * 10^(-6); %[kg/Ns]
elseif engineType == "high-bypass turbofan"
    C_cruise = 14.1; %Specific fuel consumption for cruise [mg/Ns]
    C_loiter = 11.3; %Specific fuel consumption for loiter [mg/Ns]
    %Convert to proper units
    C_cruise = C_cruise * 10^(-6); %[kg/Ns]
    C_loiter = C_loiter * 10^(-6); %[kg/Ns]
elseif engineType == "piston-prop fixed pitch"
    C_cruise = 0.068; %Specific fuel consumption for cruise [mg/W-s]
    C_loiter = 0.085; %Specific fuel consumption for loiter [mg/W-s]
    %Convert to proper units
    C_cruise = C_cruise * 10^(-6) * V / 0.7; %[kg/Ns]
    C_loiter = C_loiter * 10^(-6) * V / 0.7; %[kg/Ns]
elseif engineType == "piston-prop variable pitch"
    C_cruise = 0.068; %Specific fuel consumption for cruise [mg/W-s]
    C_loiter = 0.085; %Specific fuel consumption for loiter [mg/W-s]
    %Convert to proper units
    C_cruise = C_cruise * 10^(-6) * V / 0.8; %[kg/Ns]
    C_loiter = C_loiter * 10^(-6) * V / 0.8; %[kg/Ns]
else %Turboprop
    C_cruise = 0.085; %Specific fuel consumption for cruise [mg/W-s]
    C_loiter = 0.101; %Specific fuel consumption for loiter [mg/W-s]
    %Convert to proper units
    C_cruise = C_cruise * 10^(-6) * V / 0.8; %[kg/Ns]
    C_loiter = C_loiter * 10^(-6) * V / 0.8; %[kg/Ns]
end

%The identification of the category is essential to estimate the empty weight
%fraction
if category == "military cargo" || category == "military bomber"
    %Variable sweep constant Kvs
    Kvs = 1.00;
    %Other constants
    A = 0.88;
    c = -0.07;
elseif category == "general aviation single engine"
    %Variable sweep constant Kvs
    Kvs = 1.00;
    %Other constants
    A = 2.05;
    c = -0.18;
```

```
    elseif category == "general aviation twin engine"
        %Variable sweep constant Kvs
        Kvs = 1.00;
        %Other constants
        A = 1.4;
        c = -0.10;
    elseif category == "jet transport"
        %Variable sweep constant Kvs
        Kvs = 1.04;
        %Other constants
        A = 0.97;
        c = -0.06;
    end

else %English unit
    %Converting the input value to proper unit
    loiterTime1 = loiterTime1 * 3600; %[hr] to [s]
    loiterTime2 = loiterTime2 * 3600; %[hr] to [s]
    range = range * 6076.12; %[nmi] to [m]
    V = V_mach * 994.8; %[ft/s] (crude estimate of altitude 30000ft/9144m)

    %Assigning proper specific fuel consumption values for each engine type
    if engineType == "pure turbojet"
        C_cruise = 0.9; %Specific fuel consumption for cruise [1/hr]
        C_loiter = 0.8; %Specific fuel consumption for loiter [1/hr]
        %Convert to proper units
        C_cruise = C_cruise / 3600; %[1/s]
        C_loiter = C_loiter / 3600; %[1/s]
    elseif engineType == "low-bypass turbofan"
        C_cruise = 0.8; %Specific fuel consumption for cruise [1/hr]
        C_loiter = 0.7; %Specific fuel consumption for loiter [1/hr]
        %Convert to proper units
        C_cruise = C_cruise / 3600; %[1/s]
        C_loiter = C_loiter / 3600; %[1/s]
    elseif engineType == "high-bypass turbofan"
        C_cruise = 0.5; %Specific fuel consumption for cruise [1/hr]
        C_loiter = 0.4; %Specific fuel consumption for loiter [1/hr]
        %Convert to proper units
        C_cruise = C_cruise / 3600; %[1/s]
        C_loiter = C_loiter / 3600; %[1/s]
    elseif engineType == "piston-prop fixed pitch"
        C_cruise = 0.4; %Specific fuel consumption for cruise [lb/hr-bhp]
        C_loiter = 0.5; %Specific fuel consumption for loiter [lb/hr-bhp]
        %Convert to proper units
        C_cruise = C_cruise * V / 550 / 0.7 / 3600; %[1/s]
        C_loiter = C_loiter * V / 550 / 0.7 / 3600; %[1/s]
    elseif engineType == "piston-prop variable pitch"
        C_cruise = 0.4; %Specific fuel consumption for cruise [lb/hr-bhp]
        C_loiter = 0.5; %Specific fuel consumption for loiter [lb/hr-bhp]
        %Convert to proper units
        C_cruise = C_cruise * V / 550 / 0.8 / 3600; %[1/s]
        C_loiter = C_loiter * V / 550 / 0.8 / 3600; %[1/s]
    else %Turboprop
```

```
        C_cruise = 0.5; %Specific fuel consumption for cruise [lb/hr-bhp]
        C_loiter = 0.6; %Specific fuel consumption for loiter [lb/hr-bhp]
        %Convert to proper units
        C_cruise = C_cruise * V / 550 / 0.8 / 3600; %[1/s]
        C_loiter = C_loiter * V / 550 / 0.8 / 3600; %[1/s]
    end

    %The identification of the category is essential to estimate the empty weight
    %fraction
    if category == "military cargo" || category == "military bomber"
        %Variable sweep constant Kvs
        Kvs = 1.00;
        %Other constants
        A = 0.93;
        c = -0.07;
    elseif category == "general aviation single engine"
        %Variable sweep constant Kvs
        Kvs = 1.00;
        %Other constants
        A = 2.36;
        c = -0.18;
    elseif category == "general aviation twin engine"
        %Variable sweep constant Kvs
        Kvs = 1.00;
        %Other constants
        A = 1.52;
        c = -0.10;
    elseif category == "jet transport"
        %Variable sweep constant Kvs
        Kvs = 1.04;
        %Other constants
        A = 1.02;
        c = -0.06;
    end
end

%Depending on if the engine is jet engine or propeller the maximum value of L/D
%ratio differs; therefore, we must adjust it depending on the engineType
if engineType == "pure turbojet" || engineType == "low-bypass turbofan"...
        || engineType == "high-bypass turbofan"
    LDratio_cruise = 0.866 * LDratio;
    LDratio_loiter = LDratio;
else %propellers
    LDratio_cruise = LDratio;
    LDratio_loiter = 0.866 * LDratio;
end
```

Main

Now that the preparations are complete, the next step is the main part of this program.

In this part, there will be an algorithm that will initialize a loop with a guessed W-gross value and calculate a new W_gross_new value and compare this value with the guessed W_gross. The condition of the loop will be that until the W_gross and W_gross_new have an 0.0001 accuracy.

```
%Setting the temporary/initial W_gross, W_gross_new, and W_save to initiate the loop
if unitType == "metric"
    W_gross_temp = 22679.619; %[kg]
    W_save = 22679.619; %[kg]
    W_gross_new = 10000000; %[kg]
else %English
    W_gross_temp = 50000; %[lb]
    W_save = 50000; %[lb]
    W_gross_new = 10000000; %[lb]
end
```

These are the thoeries implemented in the loop

$$W_{\text{gross}} = \frac{W_{\text{crew}} + W_{\text{payload}}}{1 - \frac{W_{\text{fuel}}}{W_{\text{gross}}} - \frac{W_{\text{empty}}}{W_{\text{gross}}}} \quad \cdots (1)$$

$$\frac{W_{\text{empty}}}{W_{\text{gross}}} = AW^{C}_{\text{gross}}K_{\text{vs}} \quad \cdots (2)$$

$$\frac{W_{\text{mission}}}{W_{\text{gross}}} = \frac{W_{\text{end-to}}}{W_{\text{start-to}}} \cdot \frac{W_{\text{end-cl}}}{W_{\text{start-cl}}} \cdot \frac{W_{\text{end-cr}}}{W_{\text{start-cr}}} \cdot \frac{W_{\text{end-lr}}}{W_{\text{start-lr}}} \cdot \frac{W_{\text{end-ds}}}{W_{\text{start-ds}}} \cdot \frac{W_{\text{end-land}}}{W_{\text{start-land}}} \quad \cdots (3)$$

$$\frac{W_{\text{fuel}}}{W_{\text{gross}}} = 1.06\left(1 - \frac{W_{\text{mission}}}{W_{\text{gross}}}\right) \quad \cdots (4)$$

```
%Weight for the mission
%Warmup and takeoff
%From the table2 in chapter 3 Raymer
W1_W0 = 0.97; %[ratio unitless]

%Climb
%From the table2 in chapter 3 Raymer
W2_W1 = 0.985; %[ratio unitless]

%Cruise
W3_W2 = exp((-1) * range * C_cruise / (V * LDratio_cruise));

%Loiter
%Set conditions for where there is no loitering
if loiterTime1 ~= 0
    W4_W3 = exp((-1) * loiterTime1 * C_loiter / LDratio_loiter);
else
    W4_W3 = 1;
end

%Cruise (second)
```

```
W5_W4 = W3_W2;

%Loiter (second)
%Set conditions for where there is no loitering
if loiterTime2 ~= 0
    W6_W5 = exp((-1) * loiterTime2 * C_loiter / LDratio_loiter);
else
    W6_W5 = 1;
end

%Landing
%From the table2 of Raymer Chapter3
W7_W6 = 0.995; %[ratio unitless]

%Fuel fraction
%Calculate the W_mission and W_gross fraction
W7_W0 = W1_W0*W2_W1*W3_W2*W4_W3*W5_W4*W6_W5*W7_W6;
%Using the above value calculate the fuel fraction
Wf_W0 = 1.06 * (1 - W7_W0);

%Initialize the arrays
%Preallocate
W0_guess = zeros(100,1);
Wempty_W0 = zeros(100,1);
Wempty = zeros(100,1);
W0_new = zeros(100,1);

%Initialize counter
n = 1;

%Loop: The condition to exit this loop is going to be that the calculated W_gross_new must
%be within 0.0001 accuracy of the temporary guessed W_gross_temp value
while (W_gross_temp - W_gross_temp*0.0001) >= W_gross_new || ...
        W_gross_new >= (W_gross_temp + W_gross_temp*0.0001)

    %Updating the temporary W_gross
    W_gross_temp = W_save;

    %Empty weight fraction
    We_W0 = A * Kvs * W_gross_temp^(c);

    %Finally the new gross weight is computed
    W_gross_new = (W_crew + W_payload) / (1 - Wf_W0 - We_W0);

    %Save the value for W_gross_new
    W_save = W_gross_new;

    %Creating a table with the data (inserts data at end)
    W0_guess(n) = W_gross_temp;
    Wempty_W0(n) = We_W0;
    Wempty(n) = We_W0 * W_gross_temp;
    W0_new(n) = W_gross_new;
```

```
    %Increment counter
    n = n + 1;
end
```

After this loop ends we have a final answer which is the W_gross or the optimal takeoff weight for the designed aircraft

```
W_gross = W_gross_new; %Assigning the final output
```

Additionally we will create a table for that will give all of the data throughout the process of the iterative loop

```
%Creating table
%Rule out the trivial rows
W0_guess = W0_guess(W0_guess > 0);
Wempty_W0 = Wempty_W0(Wempty_W0 > 0);
Wempty = Wempty(Wempty > 0);
W0_new = W0_new(W0_new > 0);
%Table
T = table(W0_guess, Wempty_W0, Wempty, W0_new)
%This table will also be an output
```

Terminate function

```
end
```