AE 5326 - Homework 4

James Grisham

3/7/2013

Problem 1

Problem Statement

A combustion chamber consists of a straight wall diffuser followed by constant area burner.

(**a**). Estimate the total pressure ratio $P_{t3.5}/P_{t3}$ and exit Mach number $M_{3.5}$ of the straight wall diffuser with an area ratio $A_{3.5}/A_3 = 4$ and $L/H = 10$ for air with $\gamma_3 = 1.33$ at $M_3 = 0.4$.

(**b**). Estimate the total pressure ratio $P_{t4}/P_{t3.5}$ and exit Mach number M_4 of the constant area burner, assuming $T_{t4}/T_{t3.5} = 2$, $M_{3.5} = 0.09$, $C_D = 1.5$, $\gamma_{3.5} = 1.33$, and $\gamma_4 = 1.3$.

(**c**). What is the overall π_B?

Part A

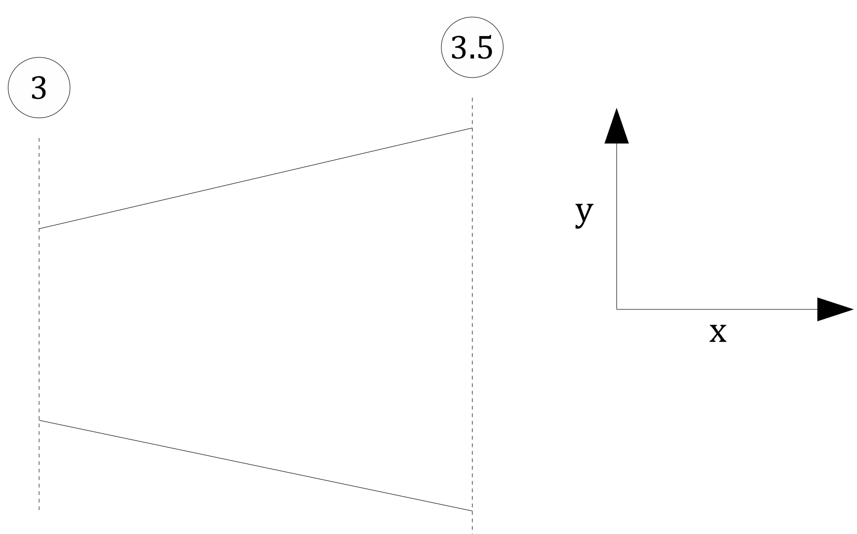

Figure 1. Control volume.

Provided with the ratios L/H and A_e/A_i, the diffuser performance, η, can be determined using Fig. 10.73 from Mattingly.

For $A_{3.5}/A_3 = 4$ and $L/H = 10$,

$$\eta = 79\%$$

Now, using Eq. 10.42b from Mattingly to determine the total pressure ratio of the diffuser,

$$\frac{P_{te}}{P_{ti}} = 1 - \frac{(\gamma/2)M_i^2(1-\eta)[1-(A_i/A_e)^2]}{(1+[(\gamma-1/2)]M_i^2)^{\gamma/(\gamma-1)}}$$

For this problem, Eq. 10.42b becomes

$$\frac{P_{t3.5}}{P_{t3}} = 1 - \frac{(\gamma_3/2)M_3^2(1-\eta)[1-(A_3/A_{3.5})^2]}{(1+[(\gamma_3-1/2)]M_3^2)^{\gamma_3/(\gamma_3-1)}}$$

$$\boxed{\frac{P_{t3.5}}{P_{t3}} = 0.98}$$

Because both the mass flow and total temperature are constant,

$$P_{t3}A_3^* = P_{t3.5}A_{3.5}^*$$

$$\frac{P_{t3.5}}{P_{t3}} = \frac{A_3^*}{A_{3.5}^*} = 0.98$$

$$\frac{A_{3.5}}{A_{3.5}^*} = \frac{A_3^*}{A_{3.5}^*}\frac{A_3}{A_3^*}\frac{A_{3.5}}{A_3}$$

The first and third terms on the right hand side of the equal sign are already known from the previous expression and the problem statement. The middle term A_3/A_3^* can be determined using the area-Mach number relation.

$$\left(\frac{A_3}{A_3^*}\right)^2 = \frac{1}{M_3^2}\left[\frac{2}{\gamma_3+1}\left(1+\frac{\gamma_3-1}{2}M_3^2\right)\right]^{(\gamma_3+1)/(\gamma_3-1)} \tag{1}$$

$$\frac{A_3}{A_3^*} = 3.75$$

Therefore,

$$\frac{A_{3.5}}{A_{3.5}^*} = (0.98)\,(3.75)\,(4) = 14.7$$

Applying (1) at station 3.5,

$$\left(\frac{A_{3.5}}{A_{3.5}^*}\right)^2 = \frac{1}{M_{3.5}^2}\left[\frac{2}{\gamma_3+1}\left(1+\frac{\gamma_3-1}{2}M_{3.5}^2\right)\right]^{(\gamma_3+1)/(\gamma_3-1)} \tag{2}$$

Solving (2) for $M_{3.5}$ and choosing the subsonic solution,

$$\boxed{M_{3.5} = 0.04}$$

Part B

Using Equations 10.35-10.38 from Mattingly,

$$\Phi = \frac{\gamma_{3.5}}{\gamma_4} \frac{M_{3.5}^2(1+[(\gamma_{3.5}-1)/2]M_{3.5}^2)}{[1+\gamma_{3.5}M_{3.5}^2(1-C_D/2)]^2} \frac{T_{t4}}{T_{t3.5}} = 0.0165$$

$$\boxed{M_4 = \sqrt{\frac{2\Phi}{1-2\gamma_4\Phi+\sqrt{1-2(\gamma_4+1)\Phi}}} = 0.131}$$

$$\frac{P_4}{P_{3.5}} = \frac{1+\gamma_{3.5}M_{3.5}^2(1-C_D/2)}{1+\gamma_4 M_4^2} = 0.981$$

$$\boxed{\frac{P_{t4}}{P_{t3.5}} = \frac{P_4}{P_{3.5}} \frac{(1+[(\gamma_4-1)/2]M_4^2)^{\gamma_4/(\gamma_4-1)}}{(1+[(\gamma_{3.5}-1)/2]M_{3.5}^2)^{\gamma_{3.5}/(\gamma_{3.5}-1)}} = 0.986}$$

MATLAB code for calculations:

```
%-------------------------------------------------------------------------------
%% Clearing workspace
%-------------------------------------------------------------------------------
clc,clear,close all

%-------------------------------------------------------------------------------
%% Inputs
%-------------------------------------------------------------------------------
T_t40qT_t35 = 2;
M_35 = 0.09;
C_D = 1.5;
gamma_35 = 1.33;
gamma_40 = 1.3;

%-------------------------------------------------------------------------------
%% Calculations
%-------------------------------------------------------------------------------

Phi = gamma_35/gamma_40*...
    M_35^2*(1+(gamma_35-1)/2*M_35^2)/...
    (1 + gamma_35*M_35^2*(1 - C_D/2))^2*...
    T_t40qT_t35;
fprintf('Phi = %1.4f\n\n',Phi)

M_40 = sqrt(2*Phi/(1 - 2*gamma_40*Phi + sqrt(1 - 2*(gamma_40 + 1)*Phi)));
fprintf('M_4 = %1.3f\n\n',M_40)

P_40qP_35 = (1 + gamma_35*M_35^2*(1 - C_D/2))/(1 + gamma_40*M_40^2);
fprintf('P_4/P_3.5 = %1.3f\n\n',P_40qP_35)

P_t40qP_t35 = P_40qP_35*(1 + (gamma_40-1)/2*M_40^2)^...
    (gamma_40/(gamma_40-1))/...
    (1 + (gamma_35-1)/2*M_35^2)^(gamma_35/(gamma_35-1));
fprintf('P_t4/P_t3.5 = %1.3f\n\n',P_t40qP_t35)
```

Part C

$$\boxed{\pi_B = \frac{P_{t4}}{P_{t3.5}}\frac{P_{t3.5}}{P_{t3}} = (0.986)(0.980) = 0.966}$$

Problem 2

Problem Statement

Estimate the burner efficiency of a main burner with $T_{t3} = 1000°\text{R}$, $\dot{m} = 100$ lbm/s, $\phi = 0.6$, $A_{\text{ref}} = 1.5\ \text{ft}^2$, $H = 2$ in, for $50 \leq P_{t3} \leq 200$ psia.

Solution

Using Equations 10.45 and 10.46 from Mattingly, the reaction rate parameter (θ) and b are determined.

$$b = 382\left(\sqrt{2} \pm \ln\left[\frac{\phi}{1.03}\right]\right)$$

$$\theta = \frac{P_{t3}^{1.75} A_{\text{ref}} H \exp(T_{t3}/b)}{\dot{m}} \times 10^{-5}$$

Using b and θ, η_B can be determined using Figure 10.80 from Mattingly.

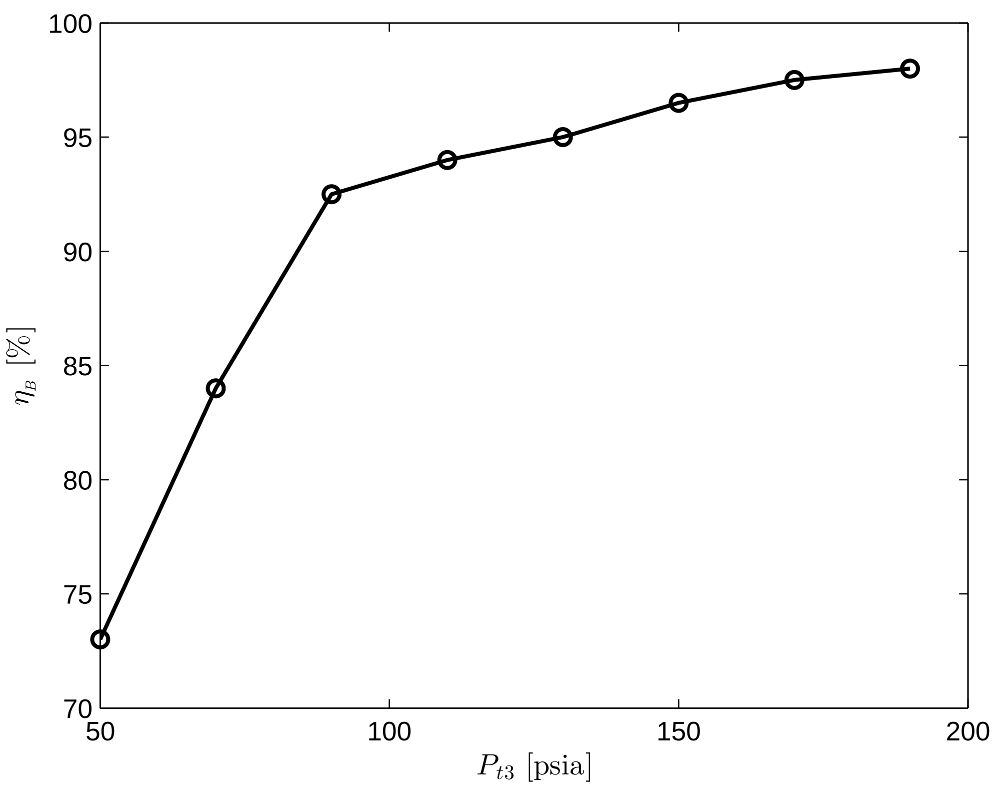

Figure 2. Plot of η_B vs P_{t3} determined from Figure 10.80.

MATLAB code:

```
%------------------------------------------------------------------------------
%% Clearing workspace
%------------------------------------------------------------------------------
clc,clear,close all

%------------------------------------------------------------------------------
%% Inputs
%------------------------------------------------------------------------------
T_t3 = 1000;
mdot = 100;
phi = 0.6;
A_ref = 1.5*144;
H = 2;
P_t3 = 50:20:200;

%------------------------------------------------------------------------------
%% Calculations
%------------------------------------------------------------------------------

b = 382*(sqrt(2) + log(phi/1.03));
fprintf('For phi = %1.1f, b = %3.1f\n\n',phi,b)
eta_b = zeros(1,numel(P_t3));

eta_b = [0.73 0.84 0.925 0.94 0.95 0.965 0.975 0.98];

% for n = 1:numel(P_t3)
%     theta = P_t3(n)^(1.75)*A_ref*H*exp(T_t3/b)/mdot*10^(-5);
%     fprintf('For P_t3 = %3.0f, theta = %2.2f\n\n',P_t3(n),theta)
%     eta_b(n) = input('Enter eta_b: ');
%     fprintf('\n')
% end
%
% dec = menu('Do you want to overwrite the old plot?','Yes','No');
% if dec == 1

    figure('Position',[1000 60 560 420])
    plot(P_t3,eta_b*100,'-ok','LineWidth',1.5)
    xlabel('$P_{t3}$ [psia]','Interpreter','LaTeX','FontSize',11)
    ylabel('$\eta_{_B}$ [\%]','Interpreter','LaTeX','FontSize',11)

    % Saving plot
    ImgPath = ['C:\Users\James\Desktop\School\Courses\UTA',...
    '\AE 5326 - Air-Breathing Propulsion\Images\'];
    set(gcf,'PaperPositionMode','auto')
    print(gcf, '-depsc', [ImgPath,'Hw5Prob2.eps'])
% end
```

Problem 3

Problem Statement

Assuming a burner with entrance temperature $T_{t3} = 800$K, and air mass flow rate of 100 kg/s, calculate the fuel air ratio as a function of T_{t4} for the range 1000K $\leq T_{t4} \leq$ 2500K 500K increments.

(**a**). Using the 2γ model, with $R = 287$ J/(kg K), γ_3=1.4, $\gamma_4 = 1.3$

(**b**). AFPROP Software

(**c**). NPSS `Burner_solver_example.run`

Solution

Part A

Energy balance on the burner yields

$$\dot{m}_0 c_{pc} T_{t3} + \eta_B \dot{m}_f h_{PR} = (\dot{m}_0 + \dot{m}_f) c_{ph} T_{t4}$$

Noting that the fuel/air ratio is given by $f = \dot{m}_f / \dot{m}_0$ and manipulating,

$$f = \frac{c_{ph} T_{t4} - c_{pc} T_{t3}}{\eta_B h_{PR} - c_{ph} T_{t4}}$$

Part B

Solving for the fuel/air ratio using the variable c_p model is an iterative process. Using $T_{t3} = 800$ K and $f = 0$ as inputs to AFProp to find h_{t3},

$$h_{t3} = 821.7 \text{ kJ/kg}$$

Using h_{t3}, η_B and h_{PR} as inputs to the `Burner.m` function, the fuel/air ratio can be determined. Values of $\eta_B = 0.99$ and $h_{PR} = 42800$ kJ/kg were assumed.

Part C

To determine the fuel/air ratio using NPSS, the `Burner_solver_example.run` was modified.

Results

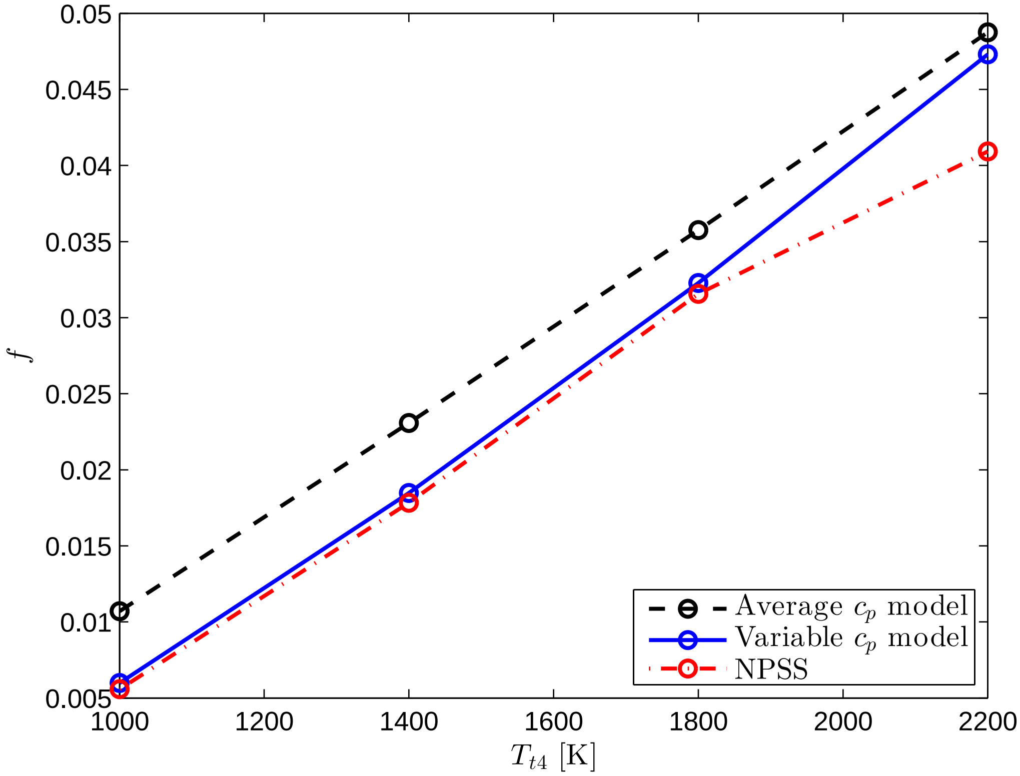

Figure 3. Fuel/air ratio vs turbine inlet temperature.

	Fuel/air ratio (f)		
T_{t4} [K]	Average c_p model	Variable c_p model	NPSS
1000	0.0107	0.0060	0.0056
1400	0.0231	0.0185	0.0178
1800	0.0358	0.0323	0.0316
2200	0.0488	0.0473	0.0409

Table 1. Fuel/air ratios.

The variable c_p model and NPSS compare well with the exception of the last data point at $T_{t4} =$ 2200 K. The average c_p model consistently predicts a higher fuel/air ratio compared to the variable c_p model and NPSS.

NPSS code

```

//   File Name: Burner_solver_example.run
//   Date: November 2010
//   Author: P. Johnson
//
//   Description: Model to demonstrate the burner element with a solver
//                balance that varies fuel flow to achieve a target ...
    temperature.
//
//------------------------------------------------------------------------------

// Set the thermo package
setThermoPackage("GasTbl");  // developed by Pratt and Whitney

#include <InterpIncludes.ncp>

//------------------------------------------------------------------------------
//                             Model Definition
//------------------------------------------------------------------------------

// Start the flow of air

Element FlowStart FsAir {
    Pt = 50.; // psi
    Tt = 1440.; // R
    W = 220.4622; // lbm/s
    WAR = 0; // Water to Air ratio
}

// Start the flow of fuel

Element FuelStart Fus {
    Wfuel = 0.3; // lbm/s, fuel flow rate (Used ONLY when Burner ...
        switchBurn = WFUEL)
    //LHV = 18000; // BTU/lbm, user input fuel lower heating value ...
        (LHV).  Default is 18400 BTU/lbm
}

// Burn the fuel and air

Element Burner Brn {
    dPqP_dmd         = 0.05; // user input friction relative pressure ...
        drop (Pin - Pout)/Pin
    effBase          = 0.99; // user input burner adiabatic efficiency

    switchBurn = "WFUEL"; // If WFUEL, then use fuel start element flow ...
        rate to set Wfuel (Fu_I.Wfuel inherited from fuel start Fu_O.Wfuel)

    // Declare a variable for requested combuster outlet temperature
    real Tout_req; // Rankine
```

```

    // Set requested outlet temperature
    Tout_req = 3690.; // Rankine

}

// End the flow of air

Element FlowEnd FeAir;

//---------------------------------------------------------------------------
//                            Component Linkages
//---------------------------------------------------------------------------
linkPorts( "FsAir.Fl_O"     , "Brn.Fl_I"      , "F030" );
linkPorts( "Brn.Fl_O"       , "FeAir.Fl_I"    , "F040" );

linkPorts( "Fus.Fu_O"       , "Brn.Fu_I"      , "Fu010" );

//---------------------------------------------------------------------------
//                            Solver Sequence
//---------------------------------------------------------------------------

// If solver execution sequence is not set by the user (below), the ...
    default sequence will be the order of the element instantiation above
/*
solver.executionSequence = {
        "FsAir",
        "AirDuct",
        "FeAir"
}
*/

//---------------------------------------------------------------------------
//                            Solver Variables
//---------------------------------------------------------------------------



// Declare solver independent variable to vary FuelStart fuel flow ...
    rate, Wfuel
Independent ind_Wfuel{
        varName = "Fus.Wfuel";
}
// Declare solver dependent that matches burner outlet temperature to a ...
    requested temperature
    Dependent dep_T4{
        eq_lhs = "Brn.Fl_O.Tt"; // actual outlet temperature
        eq_rhs = "Brn.Tout_req"; // target outlet temperature (user ...
            specified)
}


/*
// Declare a solver dependent variable to be used as a constraint
```

```
Dependent Xmax {
        eq_lhs = "X"; // variable to be constrained
        eq_rhs = "A"; // set maximum allowable value
}

// Apply the constraint to the desired solver dependent variable
dep_Y.addConstraint( "Xmax", "MAX");
// Invert the constraint if the constrained variable and corresponding ...
    dependant have opposite slopes
// dep_Y.invertConstraint("Xmax");
*/

//--------------------------------------------------------------------------
//      Print Desired Values to the Command Prompt - Custom Function
//--------------------------------------------------------------------------

// Create a custom function to print the model results values.
// Results will be printed to the command window whenever the
// function "printResults()" gets called.

void printResults(){
    // Print out a list of the active solver independents and ...
        dependents to the command prompt
    cout << endl;
    cout << "Solver Indep and Dep variables" << endl;
    cout << endl;
    cout << solver.list( "Independent" );
    cout << endl;
    cout << solver.list( "Dependent" );
    cout << endl;
    // Print some statepoint values
    cout << "F030.Tt = " << F030.Tt << " " << F030.Tt.units << endl;
    cout << "F030.Pt = " << F030.Pt << " " << F030.Pt.units << endl;
    cout << "F030.W = " << F030.W << " " << F030.W.units << endl;
    cout << endl;
    cout << "F040.Tt = " << F040.Tt << " " << F040.Tt.units << endl;
    cout << "F040.Pt = " << F040.Pt << " " << F040.Pt.units << endl;
    cout << "F040.W = " << F040.W << " " << F030.W.units << endl;
    cout << endl;
    cout << "Wair = " << FsAir.Fl_O.W << endl;
    cout << "Wfuel = " << Brn.Wfuel << endl;
    cout << "FAR = " << Brn.Fl_O.FAR << endl;

    cout << endl;
    cout << "Solver converged? = " << solver.converged << endl;
    //Constraints
    cout << "Constraints Active? = " << solver.constraintsActive << endl;
    cout << "Constraints Hit = " << solver.constraintsHit() << endl;
}


//------------------------------------------------------
// Add or remove solver independents and dependents
//------------------------------------------------------
```

```
solver.addIndependent( "ind_Wfuel" );
solver.addDependent( "dep_T4" );
//solver.removeIndependent( "ind_X" );
//solver.removeDependent( "dep_Y" );

// Turn solver constraints on or off
//dep_Y.useConstraints = FALSE; // TRUE or FALSE


//-----------------------------------------------------
//                    Run the model
//-----------------------------------------------------
cout << endl;
cout << "======================================= \n";
cout << "=====      RUNNING THE MODEL      ===== \n";
cout << "======================================= \n";
cout << endl;
// Run the model
run();

// Print model results to the command window
printResults();
```

Problem 3 MATLAB code

```
%--------------------------------------------------------------------------
%% Clearing workspace
%--------------------------------------------------------------------------
clc,clear,close all

%--------------------------------------------------------------------------
%% Inputs
%--------------------------------------------------------------------------
T_t3 = 800;                                         % K
mdot = 100;                                         % kg/s
T_t4 = linspace(1000,2200,4);                       % K
R = 287;                                            % J/kg*K
gamma_3 = 1.4;
gamma_4 = 1.3;

%--------------------------------------------------------------------------
%% Calculations
%--------------------------------------------------------------------------

% Average c_p model
eta_B = 0.99;
h_PR = 42800e3;
c_pc = gamma_3*R/(gamma_3-1);
c_ph = gamma_4*R/(gamma_4-1);
f_avg = (c_ph*T_t4 - c_pc*T_t3)./(eta_B*h_PR - c_ph*T_t4);
```

```
% Variable c_p model

% Using AFProp with T = T_t3 = 800 K and f = 0 to determine h_t3

h_t3 = 821.7085e3;                                     % J/kg

% Function call to Burner function that performs calculations for iterative
% process

f_var = zeros(1,numel(f_avg));
mystr = ['To determine the fuel/air ratio in the burner, would you ',...
    'like to use previous data or perform iterative process again?'];
dec = menu(mystr,...
    'Use previous data','Perform iterative process again');
if dec == 1
    f_var = [0.005984076693615, 0.018468186697670,...
        0.032273246331669, 0.047311798344142];
elseif dec == 2
    mystr = 'Are you sure you want to go through the iterations again?';
    doublecheck = menu(mystr,'Yes','No');
    if doublecheck == 1
        for n = 1:numel(T_t4)
            fprintf('For T_t4 = %4.0f K\n\n',T_t4(n))
            f_var(n) = Burner(eta_B,h_PR,h_t3);
        end
    end
end

% NPSS Model

% Unit conversions
fprintf('mdot = %3.0f kg/s = %3.0f lbm/s\n\n',mdot,mdot*2.20462248)
fprintf('T_t3 = %4.0f K = %4.0f R\n\n',T_t3,KtoR(T_t3))
for n = 1:numel(T_t4)
    fprintf('T_t4 = %4.0f K = %4.0f R\n\n',T_t4(n),KtoR(T_t4(n)))
end

% Output from NPSS
f_npss = [0.00557477 0.0178372 0.0315621 0.0409202];

%--------------------------------------------------------------------------
%% Plotting and outputting results
%--------------------------------------------------------------------------

plot(T_t4,f_avg,'--ok',T_t4,f_var,'-ob',T_t4,f_npss,'-.or',...
    'LineWidth',1.5)
lh = legend('Average $c_p$ model','Variable $c_p$ model','NPSS');
set(lh,'Location','SouthEast')
set(lh,'Interpreter','LaTeX')
set(lh,'FontSize',11)
xlabel('$T_{t4}$ [K]','Interpreter','LaTeX','FontSize',11)
ylabel('$f$','Interpreter','LaTeX','FontSize',11)

```

```
% Saving plot
ImgPath = ['C:\Users\James\Desktop\School\Courses\UTA',...
    '\AE 5326 - Air-Breathing Propulsion\Images\'];
set(gcf,'PaperPositionMode','auto')
print(gcf, '-depsc', [ImgPath,'Hw5Prob3.eps'])

% Outputting results
disp('T_t4')
disp(T_t4)
disp('Average c_p model')
disp(f_avg)
disp('Variable c_p model')
disp(f_var)
disp('NPSS')
disp(f_npss)
```

Burner function

```
function f_final = Burner(eta_B,h_R,h_t3)
f(1) = input('Enter initial guess for f: ');
h_t4 = 1e3*input('Enter h_t4 in kJ/kg for initial guess of f: ');

if f(1) > 0.0676
    error('Max value of f for hydrocarbon fuels is 0.0676.')
end

n = 2;
e = 1;

while e > .001

    h_t4 = 1e3*input('Enter h_t4,i: ');
    f(n) = (h_t4 - h_t3)/(eta_B*h_R - h_t4);
    e = abs((f(n) - f(n-1))/f(n-1));
    fprintf('\nf = %f \n\n',f(n))
    fprintf('e = %f\n\n',e)
    n = n + 1;

end

f_final = f(end);
disp('----------------------------------------')
disp('Calculation complete')
disp('----------------------------------------')
fprintf('f_final = %0.4f\n\n',f_final)
end
```