

Andrew Albers

AAE 539

Project 1

Part i): Develop a code that calculates the gas phase conditions (pressure, temperature, density) as a function of axial distance along the nozzle. Create a new nozzle contour that reflects the contour shown in the sketch and use 100-200 points along the nozzle to provide significant resolution of heat flux gradients.

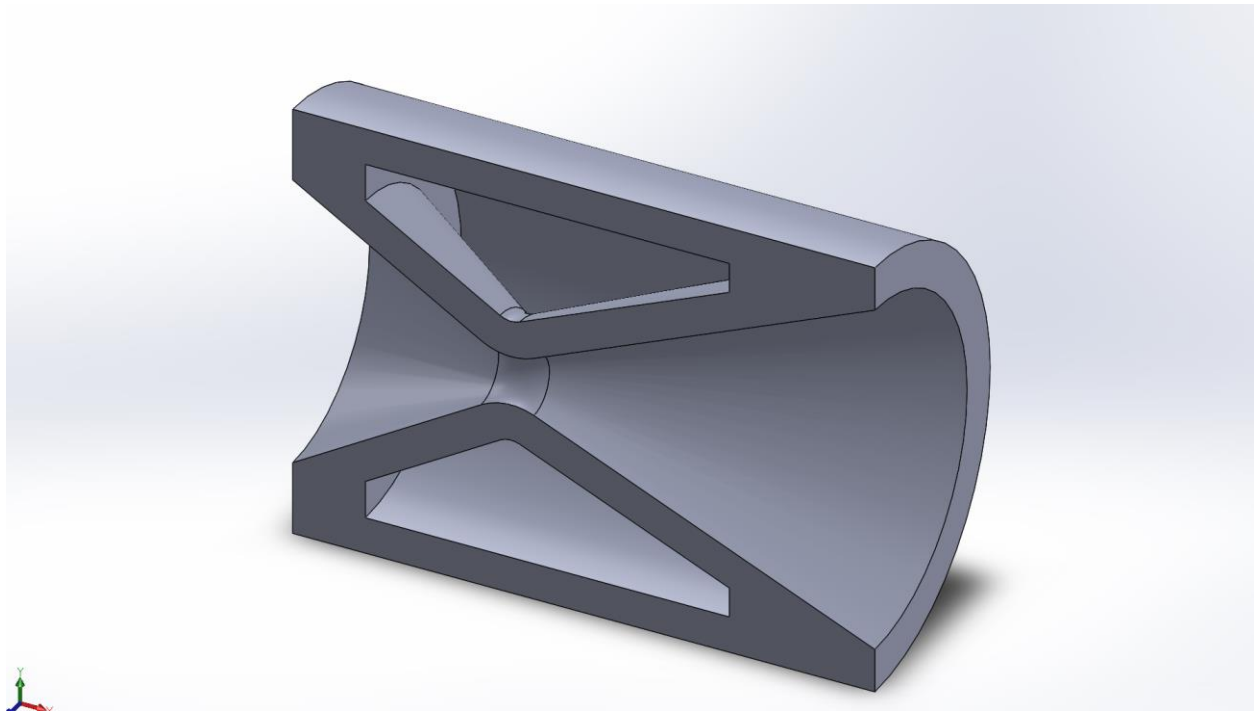
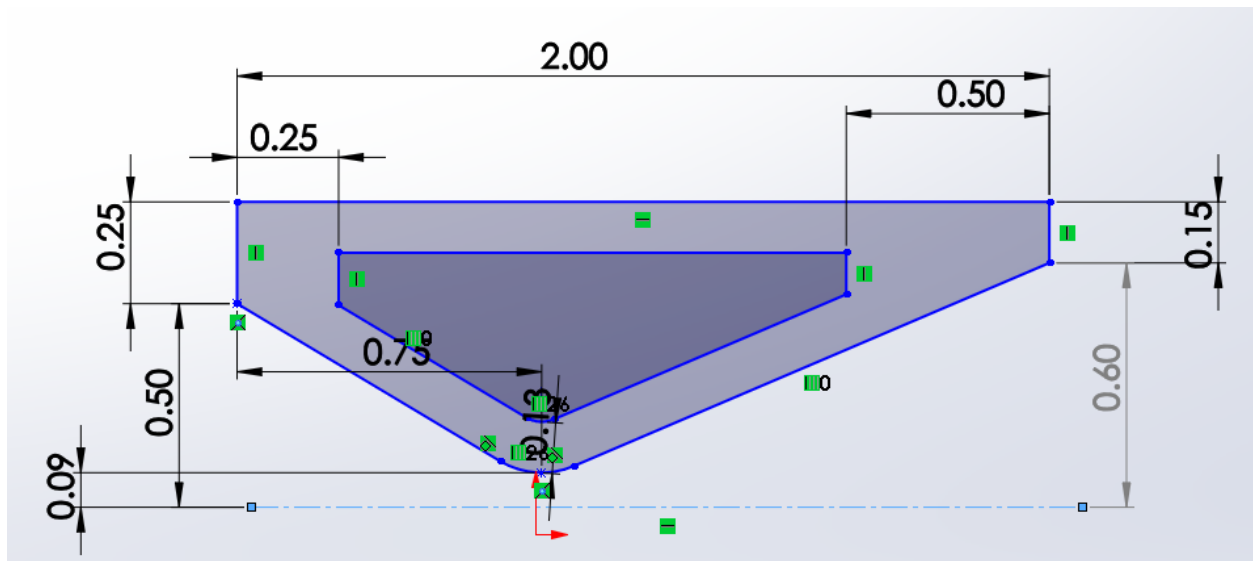
Procedure:

For this task, I developed a script that ran CEA for 5 evenly spaced points in both the subsonic and supersonic regions separately. I ended up with 13 points of CEA calculated values to be used in solving the variables in part 2 (chamber, 5 evenly spaced points of varying area ratios, throat, 5 evenly spaced points of varying area ratios, nozzle exit).

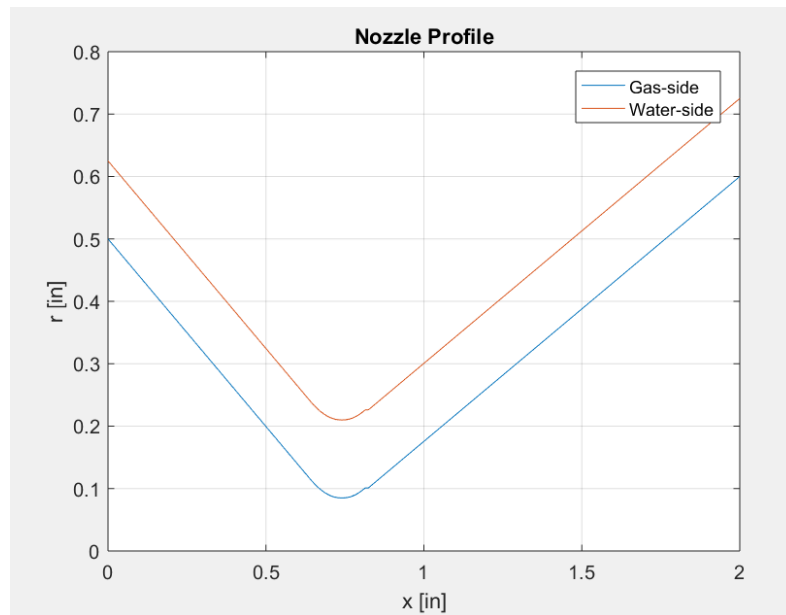
Next, to create a high resolution nozzle contour, I created a cubic spline the create 200 interpolated points. The following graphs show the developed nozzle contour as well as pressure, temperature and density of the flow from the chamber to the exit:

Next task was to create the contour. The most challenging portion of this section was accounting for the bend in the throat of the nozzle. In order to obtain an accurate contour of this section, I used Solidworks and drafted up the part according to the drawing provided. Then by placing points corresponding to the x-distance, (200 evenly spaced x points) determined the $d\theta$ between each x-distance point. This gave me a pretty accurate r value for every x value near the throat. See the matlab script attached for the specific calculation.

CAD:



Contour:



For part 1), this section of the script was used:

```
%% part 1)
%Develop a code that calculates gas phase conditions (pressure,
% temperature, and density) as a function of axial distance along the
% nozzle. Create a nozzle contour that creates 100-200 points along the
% nozzle to provide significant resolution of heat flux gradients.

% Procedure: run CEA for 5 ratios evenly distributed among both
% Sub/Supersonic regions. Then, polyfit for 200 points along axial distance
% to determine pressure, temperature, and density.

% Change this variable to true to rerun CEA instead of using saved values
CEA_RUN = true;
CEA_SAVE_FILE = 'cea.mat';

% The CEA MATLAB code takes a MATLAB map (called a dictionary in Python or
% hash in C) as input. The dictionary uses MATLAB character arrays as the
% keys, and the value data type varies by which key is used. Details of
% each key are listed in cea_rocket_run.m
% For example: inp('key') = value.
inp = containers.Map;
inp('type') = 'eq fr'; % Sets the type of CEA calculation
inp('p') = 200; % Chamber pressure
inp('p_unit') = 'psi'; % Chamber pressure units
inp('o/f') = 3.5; % Mixture ratio
inp('sup') = [1.412 2.824 4.236 5.6480 6.354 7.06]; % Supersonic area
ratios
% inp('sub') = [5.292 4.704 3.528 2.352 1.176];
inp('fuel') = 'Paraffin_Wax'; % Fuel name from thermo.inp
inp('fuel_t') = 298; % Fuel inlet temperature
inp('ox') = 'N2O4(L)'; % Ox name from thermo.inpj
inp('ox_t') = 298; % Ox inlet temperature
inp('file_name') = 'project1sup.inp'; % Input/output file name
if CEA_RUN
    data = cea_rocket_run(inp); % Call the CEA MATLAB code
    save(CEA_SAVE_FILE, 'data');
else
    load(CEA_SAVE_FILE);
end

% The output data structure, called 'data' in this case, is also a MATLAB
% map. 'data' contains a single entry for each of the CEA calculation types
% listed ('eq' and 'fr'). For instance, if only 'fr' is listed, then 'data'
% will only contain a single entry under data('fr').
data_eq = data('eq');
data_fr = data('fr');

% Use keys(data_eq) or keys(data_fr) to see the contents of each map
% respectively. Every output of CEA is contained in these keys, including
% molar concentrations. Most keys contain a 3D array with columns
% corresponding to the pressure, O/F, and area/pressure ratio inputs
% respectively. If only a single value is given for one of these inputs,
% the output will still be a 3D array. The squeeze() MATLAB function must
% be used to reduce the number of dimensions appropriately. Read the notes
% at the top of cea_rocket_read.m for more details.
temperature = squeeze(data_eq('t'));
pressure = squeeze(data_eq('p'));

% From CEA:
xfit = [0 .075 .15 .3 .45 .6 .75 1 1.25 1.5 1.75 1.875 2.0];
```

```

Pfit = [13.789 13.688 13.661 13.559 13.255 11.083 7.9207 3.0997 1.0227 0.5669 0.37755
.32038 0.27688]*14.5038;
Tfit = [3154.57 3152.11 3151.45 3148.94 3141.39 3081.78 2969.85 2647.07 2236.09
2018.36 1875.41 1819.81 1771.51];
rhofit = [1.2571 1.249 1.2469 1.2387 1.2144 1.0387 0.775 0.3448 0.1356 0.0834 0.0598
.05229 0.0464];

% generate cubic spline to interpolate for 200 points:
x = linspace(0,2,200);
dx = x(2)-x(1);
% generate radii from Solidworks sketch:
dtheta = 3.29868987;
theta = [];
rsubbend = [];
for i = 1:9
    theta(i) = 30.84893986-dtheta*(i);
    rsubbend(i) = .085+(.2-.2*cosd(theta(i)));
end
rsupbend = [];
theta = [];
for i = 1:7
    theta(i) = dtheta*(i);
    rsupbend(i) = .085+(.2-.2*cosd(theta(i)));
end
r = [linspace(.5,.1132954854,65) rsubbend .085 rsupbend linspace(.10099397,.6,118)];
D = r*2;
% r = [linspace(0.5,.17/2,75) linspace(.17/2+.0041,.6,124)];
P = spline(xfit,Pfit,x);
T = spline(xfit,Tfit,x);
rho = spline(xfit,rhofit,x);

```

Part 2): Using information from CEA, modify your program to solve the form of the Bartz equation provided on page 3 to predict the heat transfer coefficients, recovery temperature, and heat flux into the part for an assumed wall temperature of 1000 K.

Procedure:

Now that we have 200 points corresponding to our nozzle, we can calculate and plot all the variables asked for. We interpolated the values for pressure, temperature, density and Mach for every point x in part 1. We use Bartz equation to solve for hg at every point x, calculate for Tr using equation 3.13 in the notes, and solve for our heat flux using $hg \cdot (Tr - Twg)$ at every point x, with Twg assumed to be 1000K.

For Bartz equation, I was careful to use the appropriate variable for the formula:

```
sigma = 1./((.5.*Twg/Tc.*(1+(gammao-1)./2.*M.^2)+.5).^68.*(1+(gammao-1)./2.*M.^2).^12);  
hg = (0.026./(Dt.^2)).*(muo.^2*Cpo./Pr_o.^6)*(Pc*g./cstar).^8.*(At./A).^9.*sigma; % W/m^2/K  
hgmax = max(hg);
```

Where:

Pc = stagnation Pressure (chamber)

Gammao = gamma in chamber

At = throat area

A = area at current location x

Pr_o = Prandtl number using stagnation values of Cp, K, and mu (chamber)

Twg = 1000K, assumed wall temp

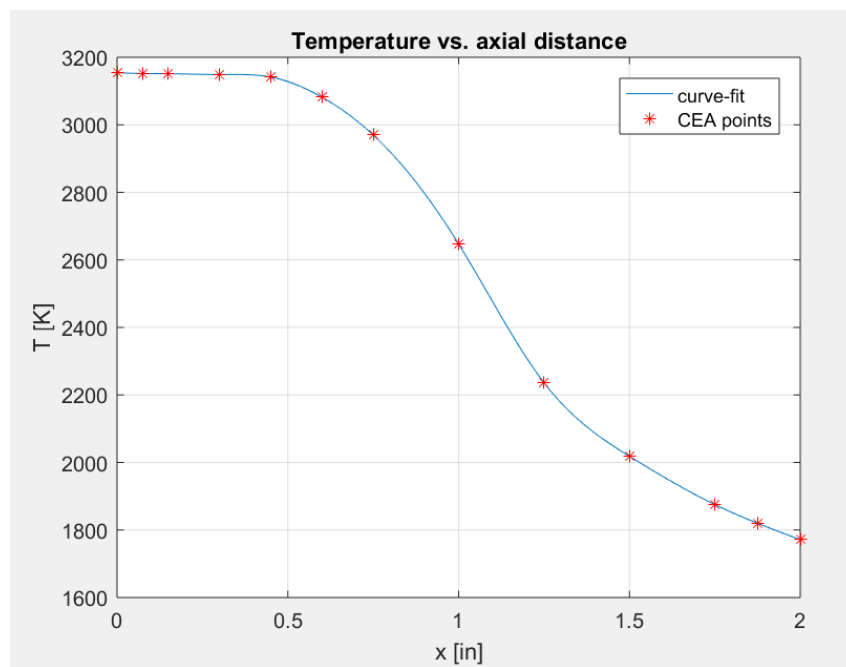
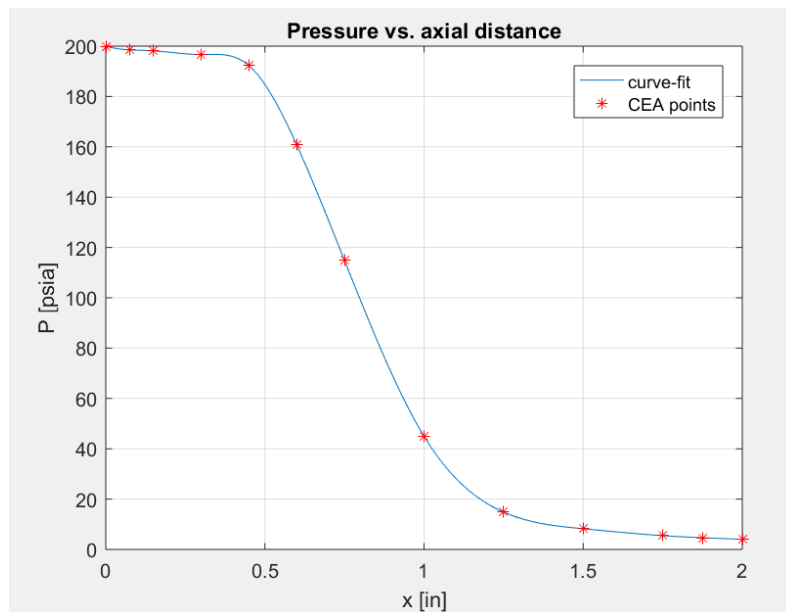
Tc = chamber temperature

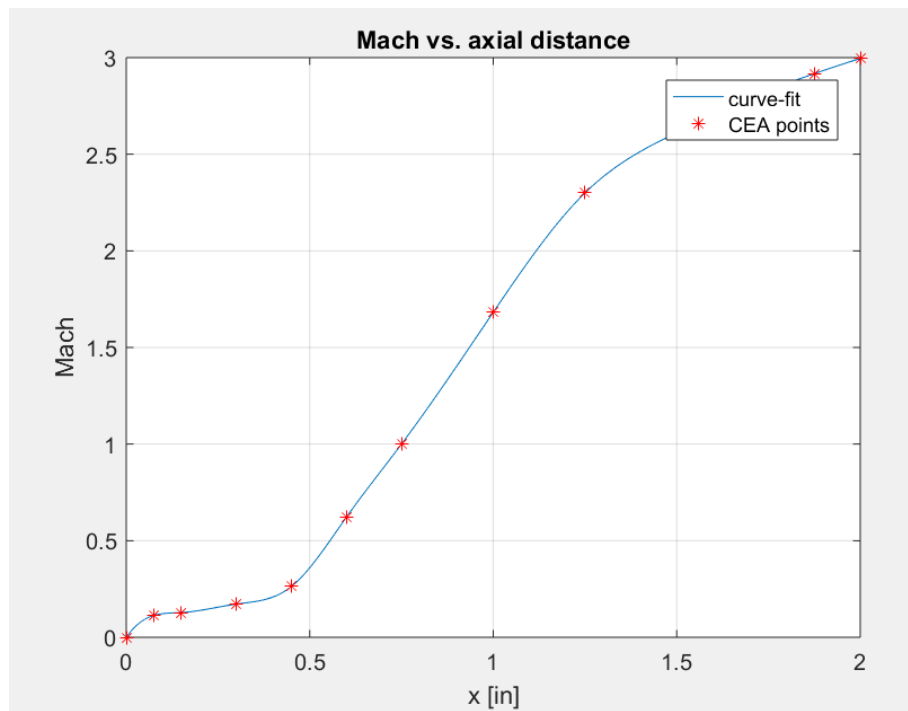
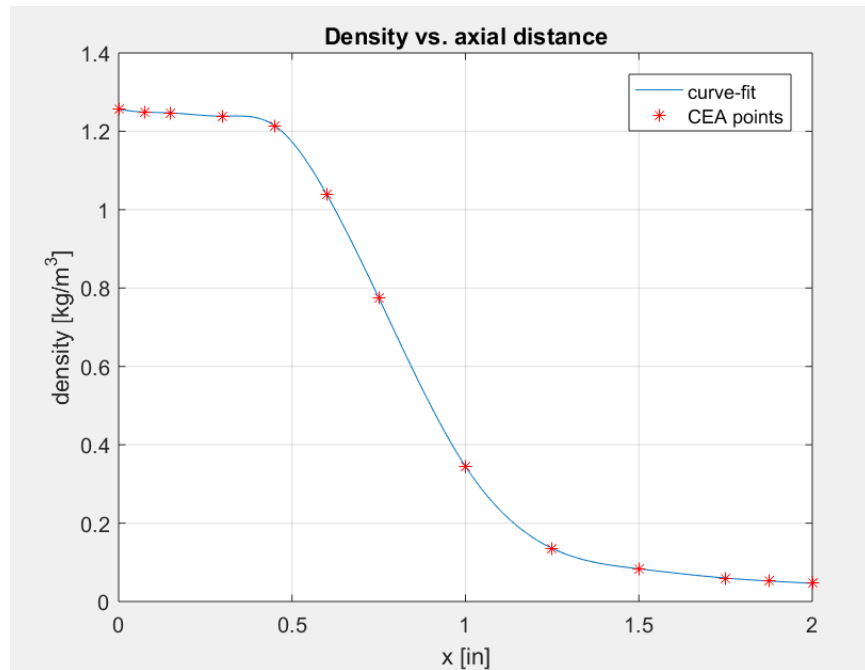
M = Mach at point x

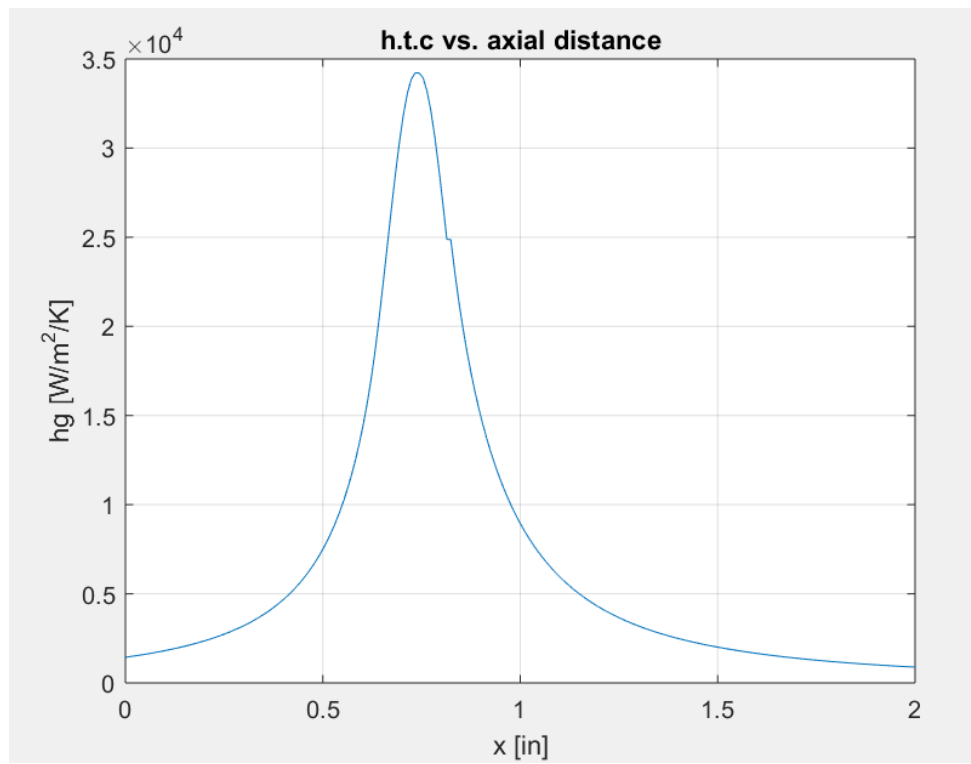
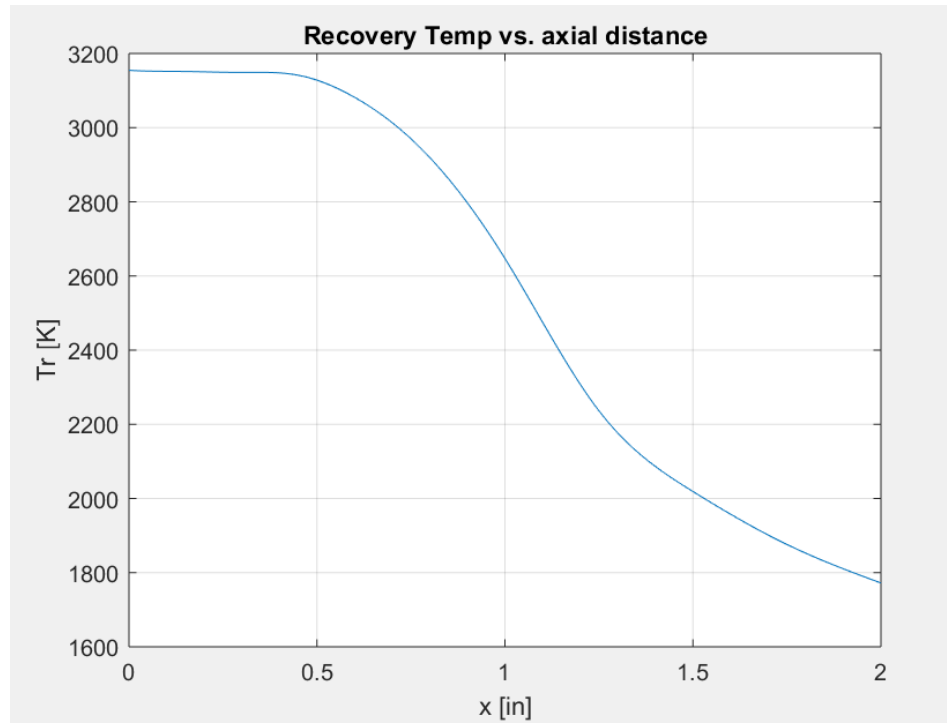
Dt = throat diameter

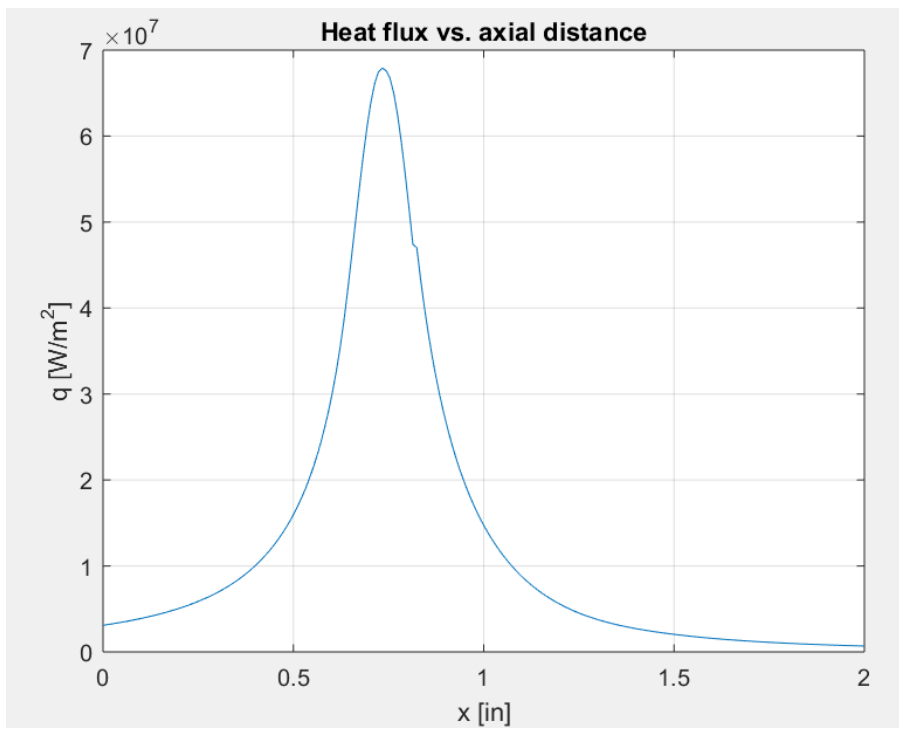
The max hg value found was: 34,218 [W/m^2/K]

The following plots were obtained for all variables asked for:



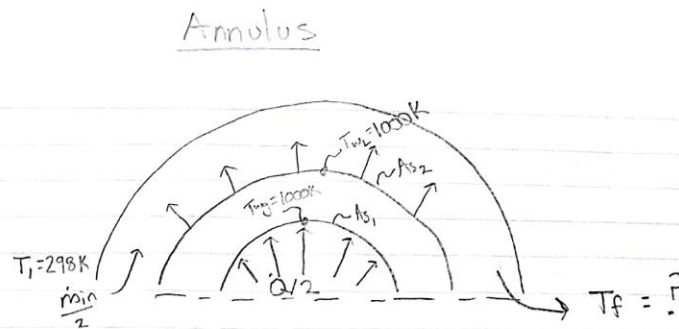






Part 3): Assume the water is injected into the annulus at ambient temperature (298 K) and estimate an average heat transfer coefficient for the annulus. Assume a wall temperature of 1000 K in your analysis. State clearly how you averaged the heat transfer coefficients to obtain your result and the assumptions regarding the flow pattern in the annulus. You may assume that the water is injected at a single point and exits at a single point on the opposite side of the annulus.

The following analysis was performed:



Assume $\dot{Q}_{in} = \dot{Q}_{out}$ (no losses)

$$\frac{\dot{Q}}{2} = h_g \frac{A_{s1}}{2} (T_r - T_{wg}) = h_L \frac{A_{s2}}{2} (T_{wL} - T_L)$$

We want average h_L , so find T_{avg} :

- Discretize half-annulus, find $T_i = T_{i-1} + \frac{\dot{Q}}{\dot{m} C_{p,H_2O}}$
- take average of $T_1 \dots T_{100}$

We don't know \dot{m} , from trial and error, determine what \dot{m} allows for $T_f < 373$ K to avoid bad heat transfer

found that $\dot{m} = 5 - 10$ is sufficient, and h_L is essentially constant in this range

$$h_L = \frac{\dot{Q}_{in}}{A_{s2} (T_{wL} - T_{avg})}$$

A few things were needed to be calculated prior to this analysis. We calculated Q_{in} by summing up all heat fluxes q for every x location, and dividing by 2 to account for only half the annulus (it was assumed that the flow was split evenly therefore, we can look at just one half of the system for simplicity). We similarly summed up all areas that were previously found using an integration and divided by 2 as well, for both the inner and outer surfaces. To discretize the system into 100 even sized slices, we simply divided each area again into 100 equal parts, and assumed $1/100^{th}$ of the heat would be added to the system in each discretized section.

By iterating, we solved for T_i for every discretized section by starting at $T_1 = 298K$ and adding the small heats as the water steps along the system. Then simply taking an average by adding all temperatures and dividing by 100.

T average for an \dot{m} of 5-10 produced a T_{final} of about 305K

We find the average h_i to be: $14100 \text{ W/m}^2/\text{K}$

Other intermediate values found:

$Q_{in} = 18,919 \text{ Watts}$

$A_{s1} = 0.0028 \text{ m}^2$

$A_{s2} = 0.0019 \text{ m}^2$

Part 3 code:

```
%% part iii)
% estimate an average heat transfer coefficient for the annulus
% perform an equilibrium balance with the heat going in vs the heat going
% out. then with everything known at the inlet of the annulus, solve for
% hl:
% step 1: obtain total heat out of system into annulus region - calculate
% and sum up all Qout for every point pertaining to annulus region.
% step 2: solve for exit temperature of the water region using  $Q = mcpdT$ ,
% where  $T_1$ 
% = 298K
% step 3: using  $T_{avg} (T_1+T_2/2)$  and  $Q_{total}$ , solve for average hl

% Try again:  $Q_{nozzle} = Q_{out}$ 
% stored energy in nozzle:  $Q_{stored} = mCpT$ 
%  $m_{nozzle} = 0.017133$ ; % kg
%  $Cp_{nozzle} = 390$ ; % J/kg/K

%  $Q_{stored} = m_{nozzle} * Cp_{nozzle} * T_{wg}$ ; % J

rho_water = 1000; % kg/m^3
% mdot = [5 10 35]*0.000063090*rho_water; % kg/s
% Cp = 4185.5; % J/kg/K for  $T_{avg} = 327, 312, 302$  K. (little change w.r.t.
temp)
Twl = 1000; % [K] assumed wall temp - liquid side
% annulus region can be found from x-points 26-150. find circumference for
% every point, multiply by distance of segment to next segment.

for i = 26:149 % all but last segment to be discretized (forward integration)
    As2(i-25) = pi*(D(i)+.25)*.0254*(sqrt(abs(x(i+1)*.0254-
x(i)*.0254)^2+(abs(r(i)*.0254-r(i+1)*.0254))^2)); % m^2
    % As(i-25) = Asannulus(i-25) + pi*(1.25*.0254)*(x(i+1)*.0254-x(i)*.0254);
end
% Surface area of nozzle area:
for i = 1:199 % all but last segment (forward step integration)
    As1(i) = pi*D(i)*.0254*(sqrt(abs(x(i+1)*.0254-
x(i)*.0254)^2+(abs(r(i)*.0254-r(i+1)*.0254))^2)); % m^2
end
% As(1) = As(1) + (pi*((1.25*.0254)^2)/4-pi*((1.15*.0254)^2)/4); % for first
discretization, add bottom region area
% As(end) = As(end) + (pi*((1.25*.0254)^2)/4 - pi*((1.18*.0254)^2)/4); % for
last discretization, add top region area
As2total = sum(As2); % + pi*(1.25*.0254)*1.25*.0254 + (pi*(1.25^2)/4 -
pi*(1.18^2)/4) + (pi*(1.25^2)/4-pi*(1.15^2)/4); % nozzle wall + top/bottom
wall + outer wall area of water region
As1total = sum(As1);

Qin = q(1:end-1).*As1; % Qin to nozzle
Qtotal = sum(Qin); % total Watts
Qwaterin = linspace(Qtotal/200,Qtotal/200,100); % looking at half of system;
As/2, Q/2...
% discretize water channel into 100 evenly sized chunks:
Aswaterin = linspace(As2total/200, As2total/200,100); % m^2
Twater = zeros(100,100);
Cp = zeros(100,100);
Twater(1,:) = 298; % K
% From NIST: (Shomate Equation)
```

```

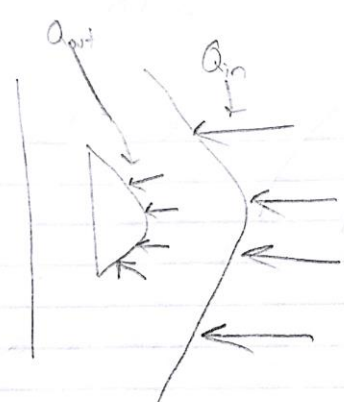
Cp(1,:) = (-203.606 + 1523.29*Twater(1)/1000 + -3196.413*(Twater(1)/1000)^2 +
2474.455*(Twater(1)/1000)^3 + 3.855326/(Twater(1)/1000)^2)/.01801488; %
J/kg/K
% assume mdot = 5 for now:
mdot = linspace(0,35/2,100)*0.000063090*rho_water; % kg/s
for j = 1:100
    for i = 2:100
        Twater(i,j) = Twater(i-1,j) + Qwaterin(1,i-1)/mdot(1,j)/Cp(i-1,j);
        Cp(i,j) = (-203.606 + 1523.29*Twater(i,j)/1000 + -
3196.413*(Twater(i,j)/1000)^2 + 2474.455*(Twater(i,j)/1000)^3 +
3.855326/(Twater(i,j)/1000)^2)/.01801488; % J/kg/K
    end
    Tavg(j) = mean(Twater(:,j));
end

% % solve for havg:
havg = Qtotal/As2total./(Tw1-Tavg); % W/m^2/K
% upon looking at these varying mdot values, we find that our hl values for
% htcs of reasonable flow rates (Tfinal < 373K) are 14200-14000.
% therefore, we will say that our hl value is: 14100.
hl = 14100;
% to validate this value, also calculate using Seider-Tate Correlation:
% define avg as location in which 50% of surface area is to the left and
% % right of the nozzle w.r.t. the axial direction.
% mu_water = [5.04e-4 6.53e-4 8.145e-4]; % Pa-s, viscosity at Tavg
% mu_s = .038e-3; % Pa-s, viscosity of vapor on surface of nozzle at 1000K
% k_water = [.6406 .6286 .6145]; % W/m/K for Tavg of 327,312.302 K
% Pr_water = Cp.*mu_water/k_water; % prandtl number for various flow rates

```

Part 4): Write out an energy balance for the bulk temperature of the nozzle (Energy In – Energy Out = Energy Stored) assuming heat input is only from the combustion gases and negligible heat loss on the outer diameter. Integrate the heat flux into the part using your results from Part (ii) and plot the bulk temperature history (from $t = 0$ to $t = 10$ s) assuming water flows of 0, 5, 10, and 35 gallons per minute. For the zero water flow case, you can assume that the entire part is made of copper (including the annular region normally occupied by the water). You may assume that the heat transfer coefficients remain constant at the values you computed in Parts (ii) and (iii) in your study.

The following analysis was performed:



$Q_{in} = Q_{total} = 18919 \text{ W}$
 $= 18.919 \text{ kW}$

$Q_{out} = h_L A_{s2} (T_w(t) - T_{avg})$

$Q_{stored} = Q_{in} - Q_{out}$

$m C_p (T_2 - T_1) = 18919 - h_L A_{s2} (T_w(t) - T_{avg})$

$T_i = T_{i-1} + \frac{18919 - h_L A_{s2} (T_{w,i-1} - T_{avg})}{m C_p}$

$T_{water}(i+1, j) =$

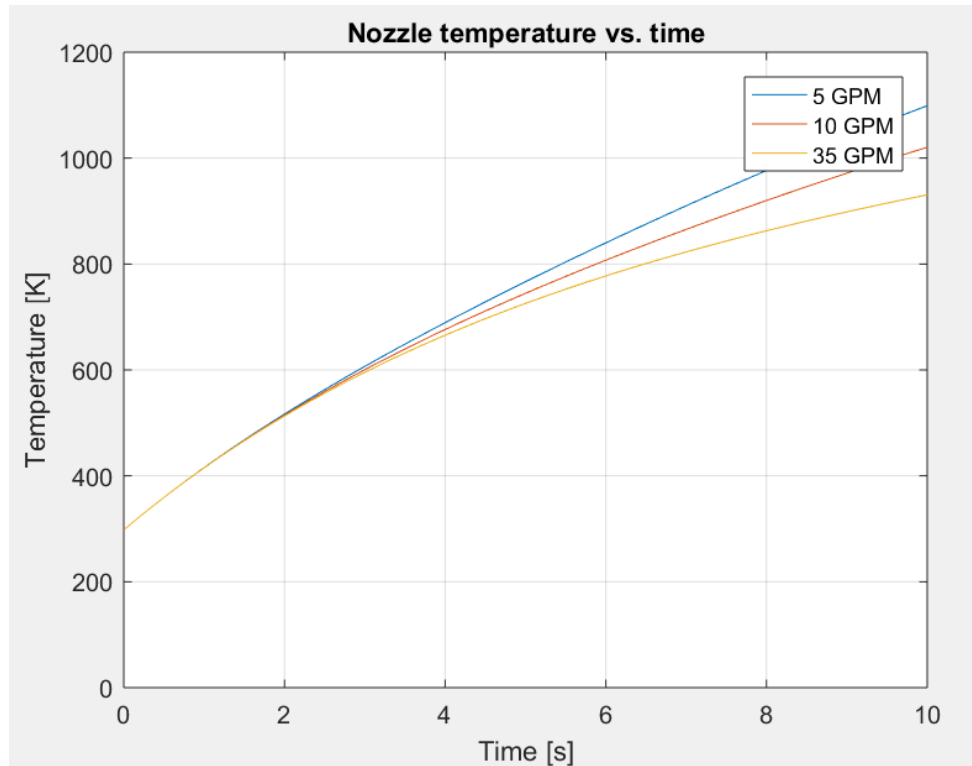
$\dot{m} C_p (T_i - T_{i-1}) = h_L A_s (T_{wL} - T_L)$

$\dot{Q} = \dot{m} C_p (T_{i-1} - T_{i-1})$

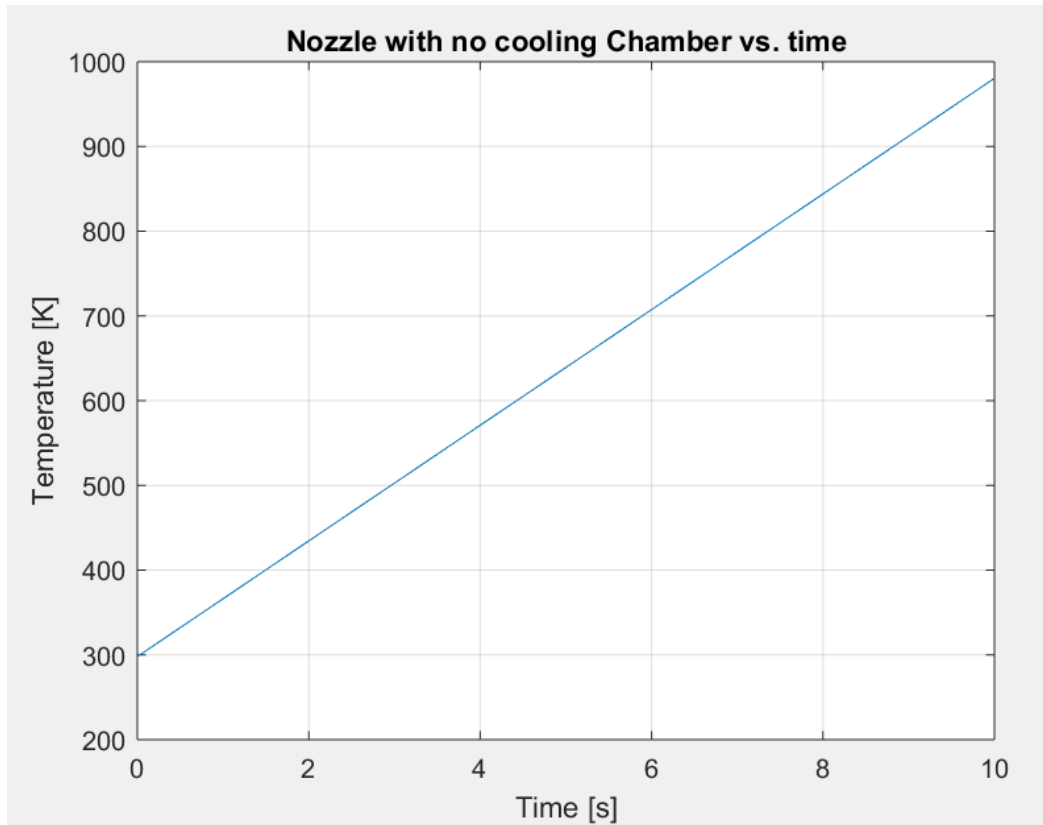
$T_i = T_{i-1} + \frac{Q_{out}}{\dot{m} C_p}$

Notes:
 - varies w/ flow rate (pointing to $T_w(t)$)
 - varies w/ flow rate (pointing to $T_{w,i-1}$)
 - bulk temp (pointing to T_{avg})
 - solid, so mostly constant (pointing to $m C_p$)

Using the values already previously calculated, and assuming a bulk temperature for the nozzle, we were able to calculate the final temperatures of the nozzle for various flow rates. The temperatures did not vary too greatly since our HI value was not solved as a function of flow rate (yet). The following Temperature vs. time plots were found:



For the 0 flow case, we assumed that the annulus region was simply filled with the copper of the nozzle. The volume and mass was found (using the CAD geometry again) and a new iteration was performed:



According to this analysis, the nozzle is not reaching its melting temperature yet, which is good. However, the bulk temperature assumption may not be the best assumption, and part 5 will go into this deeper.

Part 4 script:

```
% part iv): solve for bult temperature of the nozzle as a function of time:
% step 1: solve for Qstored: Qin-Qout = Qstored
% step 2: Solve for dT/dt: dT = (hgAs1(Tr-Twg) - hlAs2(Tw1-Tl))/m/Cp
% step 3: dT(0) = 0, dT(10) = ?
Tavg_final = [Tavg(15) Tavg(30) Tavg(100)];
mdot = [5 10 35]*1000*.000063090;% grabbed from data corresponding to 5 10 35 GPM
flows
% solve for Qin:
Vnozzle = 4.269e-5; % m^3, from CAD
mnozzle = Vnozzle*8960; % V*rho

% calculate nozzle with no flow, get volume of annulus:
Vannulus = 3.665e-5; % m^3
mnozzlenoflow = mnozzle + Vannulus*8960; % kg
Cpnozzle = 390; % J/kg/K, assume constant
```

```

% Initialize integrated temperatures/heats:
Twall = zeros(101,3);
Twallnoflow = zeros(101,1);
Twallnoflow = 298; % K
Twall(1,:) = 298; % initial temperature
Twater = zeros(101,3); % reinitialize Twater
Twater(1,:) = 298; % Assume bulk water temp
Qout = zeros(101,3);
Qin = Qtotal; % no longer assuming equilibrium, however all heat generated is still
this same value
% initialize Cp of water vs. temp again:
Cp(1,:) = (-203.606 + 1523.29*Twater(1)/1000 + -3196.413*(Twater(1)/1000)^2 +
2474.455*(Twater(1)/1000)^3 + 3.855326/(Twater(1)/1000)^2)/.01801488; % J/kg/K
for i = 1:100 % from 0-10s: 100 points for better resolution
    for j = 1:3
        Qout(i,j) = hl*As2total*(Twall(i,j)-Twater(i,j));
        Twater(i+1,j) = Twater(i,j) + (1/10)*Qout(i,j)/(mdot(j)*(1/10))/Cp(i,j);
        Twall(i+1,j) = Twall(i,j) + (1/10)*(Qin-Qout(i,j))/mnozzle/Cpnozzle;
        Cp(i+1,j) = (-203.606 + 1523.29*Twater(i+1,j)/1000 + -3196.413*(Twater(i+1,j)/1000)^2
+ 2474.455*(Twater(i+1,j)/1000)^3 + 3.855326/(Twater(i+1,j)/1000)^2)/.01801488; %
J/kg/K
        Twallnoflow(i+1,1) = Twallnoflow(i,1) + (1/10)*Qin/mnozzlenoflow/Cpnozzle;
    end
end
figure
plot(0:.1:10,Twall(:,1),0:.1:10,Twall(:,2),0:.1:10,Twall(:,3))
xlim([0 10]);
ylim([0 1200]);
title('Nozzle temperature vs. time')
xlabel('Time [s]')
ylabel('Temperature [K]')
legend('5 GPM','10 GPM','35 GPM')
grid on
figure
plot(0:.1:10,Twallnoflow(:,1));
title('Nozzle with no cooling Chamber vs. time')
xlabel('Time [s]');
ylabel('Temperature [K]');
grid on

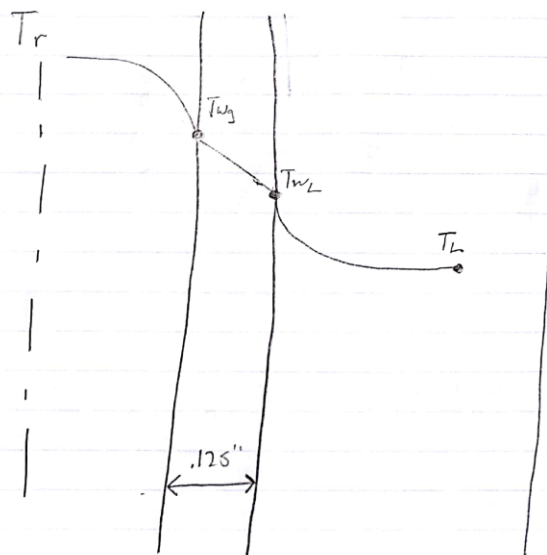
```

Part 5): Using 35 gpm water flow, perform a local analysis at the throat to determine the temperatures T_{wg} and T_{wl} . Here, you may assume that the part has reached a steadystate temperature distribution and use the techniques one would employ for a regenerative cooling jacket analysis. In this case, you will need to iterate on the heat transfer coefficients as a function of wall temperature using correlations for the liquid (Sieder-Tate, see page 3) and gas side (Bartz). What T_{wg} value do you obtain? The melting temperature of copper is about 1350 K, do we have a problem? What is the minimum required flowrate to avoid melting at the throat?

The following analysis was performed:

Part v)

Throat



$$\dot{q} = h_g (T_r - T_{wg}) = \frac{K(T_{wg} - T_{wl})}{t_w} = h_L (T_{wl} - T_L)$$

1) Assume T_{wg} :

2) solve for h_g using Bartz

3) solve for T_{wl}

4) solve for h_L using Sieder-Tate

5) check: $q_{gas} = q_{liq}$

if no, change T_{wg} and repeat

For this analysis, a lot of troubles came up. It was challenging for my script to reach a close equilibrium, and after many iterations of trying I was able to get results for a flow rate of 35 GPM as well as a higher flow rate seeking a wall temp underneath the 1350K melting temp.

The biggest issue I ran into was the heat transfer into the nozzle was so large that it generated a “negative” value for the Twl point at first. This negative Twl value caused issues with calculating \dot{q} and trying to equate the two. Other issues derived from various intermediate parameters as well.

This analysis was performed at the throat, the perceived location for melting to first occur.

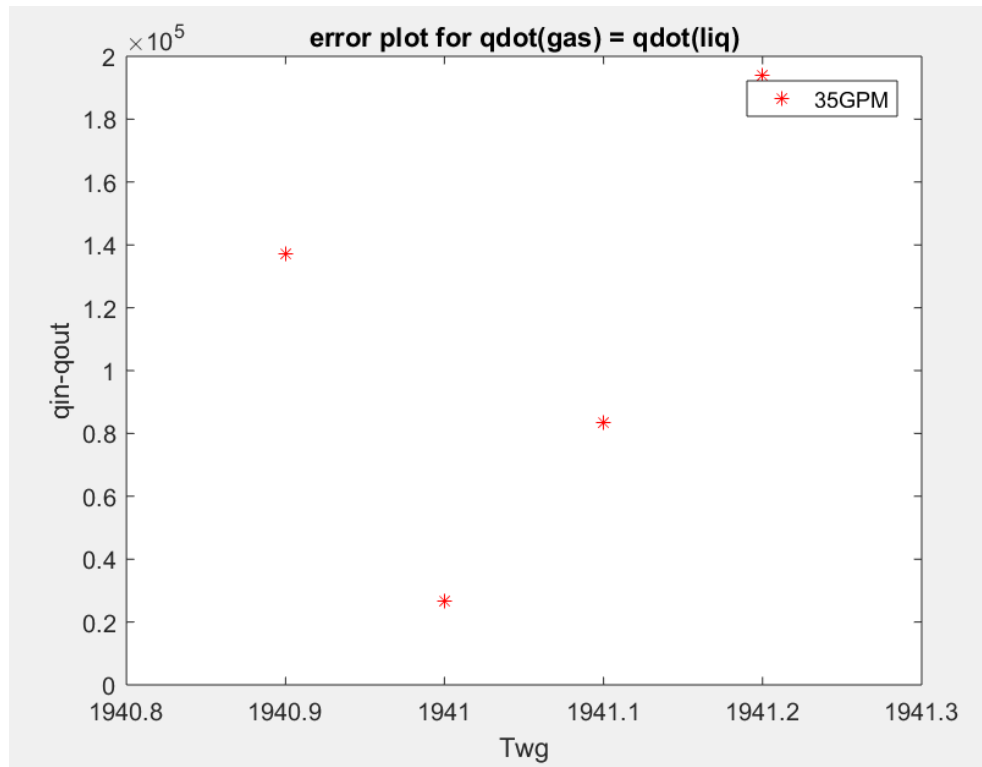
My final Twg temperature at 35GPM: 1941 K

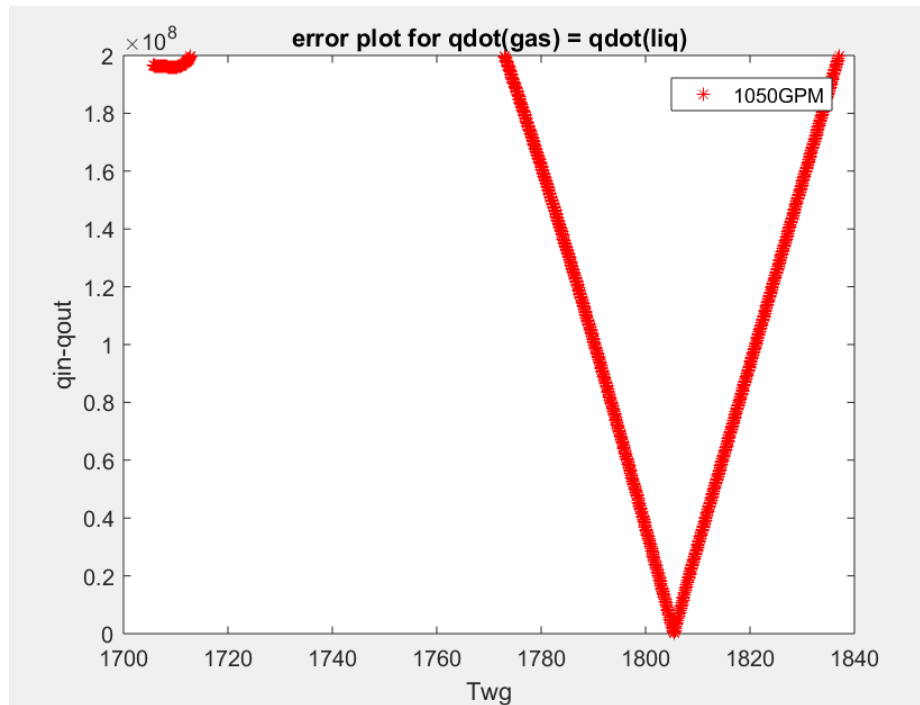
My final flow rate where 1350 K is achieved for Twg: NA

- **No reasonable flow rate was obtained, but the analysis approached 1800K at best around 1050+GPM flow rate. Clearly something went wrong which will likely be prevalent upon grading.**

In conclusion, we do in fact, have a problem. The nozzle will likely melt very quickly given the combustion event happening combined with the size and material selection of the material. I would advise that if this part were to be made, it be made with something containing a higher melting point such as steel, which contains a melting point of approx. 1700K. If my analysis were more accurate, it would likely prove that stainless steel would suffice along with reasonable water flow rate cooling.

Plots used to determine “equilibrium”:





These graphs are showing the calculation of $\text{abs}(q_{in} - q_{out})$. Where the graph approaches 0 is where we can assume is the equilibrium temperature for T_{wg} in which the heat flow into the nozzle equals the heat flow carried out by the water.

Part 5 script:

```
% part v): Perform a local analysis at the throat to determine the
% temperatures  $T_{wg}$  and  $T_{wl}$ / Here that the part a steady-state temperature
% distribution and use the techniques you would employ for a regenerative
% cooling jacket analysis. You will need to iterate on the heat transfer
% coefficients as a function of wall temperature using correlations for the
% liquid and gas side. what  $T_{wg}$  value do you obtain? The melting
% temperature of copper is 1350K. Do we have a problem? What is the
% minimum required flow rate to avoid melting at the throat?
clear all; clc; close all;
Dt = 0.004318000000000;
rho_water = 1000;
load A.mat At Asltotal
% Throat values from CEA:
Mt = 1; % mach #
Pt = 7.9207;
rhot = 0.775;
A = pi*(.17/2).^2; % m^2
At = .17*.0254; % m^2
mdot = linspace(35,1050,30)*1000*.000063090; % kg/s
kcopper = 398; % W/m/K , Assume constant (changes very little w.r.t. temp)
kc = polyfit([linspace(0,100,11)], [.55575 .57864 .59803 .61450 .62856 .64060
.65091 .65969 .66702 .67288 .67703], 3);
thw = .125*.0254; % thickness, m
% initialize temperature:
g = 9.81;
muo = 9.851100000000001e-05;
```

```

Cpo = 4.4727e3;
Ko = .94197; % W/m/K
Pr_o = (Cpo*muo/Ko);
gammao = 1.1443000000000000;
Pc = 200*6894.76; % Pa
Tc = 3.1545700000000000e+03;
cstar = 1641.7; % m/s
Tt = zeros(10000,10);
Tt(1,:) = 298;
Tl = zeros(10000,length(mdot));
Tl(1,:) = 2.990137362127620e+02; % Assume it has reached steady state temp,
299 K for 35GPM
Twl = zeros(10000,10);
Twl(1,:) = 298;
Twg = zeros(10000,10);
Twg(1,:) = 1000;
muL = [];
mu_S = [];
hg = [];
sigma = [];
qdotg = [];
kwater = [];
Pr_water = []; Nud = []; Re_water = []; hl_seider = []; qdotl = []; checkmin
= []; minEl = []; Twgfinal = [];
v_water = mdot./rho_water./(pi*Dt.^2/4); % m/s % constant, rho of water
constant
Trg = 3.149399787489976e+03; % Tr at the throat:
% q = hg(Tr-Twg) = k(Twg-Twl) = hl(Twl-Tl)
tol = 10000;
check = ones(10000,10)*10^8;
% check(:,1) = tol+1; % some arbitrary value larger than tolerance;
i = 1;
j = 1;
for j = 1:10
while check(i,j) >= tol & i <= 9999
% Guess Twg:
Twg(i,j) = 1000 + .1*(i-1); % from trial and error, learned that 2000-3000K
is close, begin there:
% Compute hg:
sigma(i,j) = 1./((.5.*Twg(i,j)/Tc.*(1+(gammao-
1)./2.*Mt.^2)+.5).^68.*(1+(gammao-1)./2*Mt.^2).^12);
hg = (0.026./(Dt.^2).*(muo.^2*Cpo./Pr_o.^6)*(Pc*g./cstar).^8).*sigma; %
W/m^2/K
% solve for qdot:
qdotg(i,j) = hg(i,j)*(Trg-Twg(i,j));
% Compute Twl:
Twl(i,j) = Twg(i,j)-qdotg(i,j)*thw/kcopper;
% Compute hl with Tl(i-1): assume Tl(1) is 299K, what we found our Tavg to
% be for 35GPM:
% viscosity:
muL = 1.74214e-3;
if Twl(i,j) <= 373
mu_S = exp(-3.7188 + 578.919/(-137.546+Twl(i,j)))/1000; % Vogel Equation
else
mu_S = 0.283636/1000;
end
% Cp of water:

```



```

Cp = (-203.606 + 1523.29*Tl(1)/1000 + -3196.413*(Tl(1)/1000)^2 +
2474.455*(Tl(1)/1000)^3 + 3.855326/(Tl(1)/1000)^2)/.01801488; % J/kg/K
% k of water:
kwater(i,j) = kc(1)*Tl(1)^3+kc(2)*Tl(1)^2 + kc(3)*Tl(1) + kc(4);
% Pr of water:
Pr_water(i,j) = (Cp.*muL./kwater(i,j));
% Re of water:
Re_water(i,j) = rho_water.*v_water(j).*Dt./muL; % not constant, need function
of viscosity vs temp.
% Seider-Tate:
Nud(i,j) =
0.027.*Re_water(i,j).^(4/5).*Pr_water(i,j).^(1/3).*(muL./mu_S).^.14;
hl_seider(i,j) = Nud(i,j).*kwater(i,j)./Dt;
% Compute qdotl:
qdotl(i,j) = hl_seider(i,j)*(Twl(i,j)-Tl(1));
% check to see if qdots are equal:
check(i,j) = abs(qdotg(i,j)-qdotl(i,j));
i = i+1;
end
i = 1;
[checkmin(j) minEl(j)] = min(check(1:end-1,j));
Twgfinal(j) = Twg(minEl(j),end);
end

% continue:
% figure
plot((1:1:9999)./10+1000,check(1:end-1,1),'r*') % error plot to check
progress
title('error plot for qdot(gas) = qdot(liq)')
ylim([0 2e5])
xlabel('Twg')
ylabel('qin-qout')
legend('35GPM')
figure
plot((1:1:9999)./10+1000,check(1:end-1,j),'r*') % error plot to check
progress
title('error plot for qdot(gas) = qdot(liq)')
ylim([0 2e8])
xlabel('Twg')
ylabel('qin-qout')
legend('1050GPM')

% mdot required for Twg <= 1350K:
Twgfinal1 = 1000+minEl(1)/10; % Kelvin at flow rate of 35GPM
Twgfinal2 = 1000+minEl(end)/10; % Kelvin at flow rate 1050 GPM, this was the
closest I was able to get to 1350K.

```

Part 6): Organize your results into a suitable form and attach a copy of your code(s). Discuss the potential drawbacks from the analysis and make conclusions as to the suitability of the cooling scheme.

Potential Drawbacks:

From this analysis, several drawbacks could be found in the methodology and assumptions used.

Firstly, for part 1), using a spline to generate the various properties asked for may not have been the most accurate calculation, but it was certainly more efficient time-wise, than running CEA 200 times. The fact of the matter is using a Spline to interpolate values for each point is good enough for a by-hand analysis of this system. If we find that these values are very important later on, an automated script could be written outside of matlab to execute and extract necessary parameters. But as said before, could take some time.

Part 2), the largest potential drawback in calculating our heat transfer properties was certainly the assumed 1000K wall temperature. The wall temperature directly impacts the heat transfer rate. If we are just beginning the ignition at $t = 0$, then we would see a heat flux per point much higher than what we calculated. Other than that, from class we discussed how Bartz is generally not very accurate, but frankly, is the best predictor we have for gas heat transfer coefficients.

Part 3), we again assume a wall temperature of 1000K, which may cause some inaccuracies in the final calculated heat transfer coefficient. This assumption is even more impactful in the analysis we performed, because both the heat transfer coefficient is dependent on wall temperature as well as the heat transfer into the water. This would make this value *very* sensitive to our heat transfer.

Part 4), Drawback here would be the bulk temperature assumption. For copper, this drawback would not be quite as devastating as a material with a lower conductivity, like Inconel for example. A higher thermal conductivity would allow for more heat to diffuse within the solid copper more quickly, creating a more uniform temperature gradient more quickly. We also are ignoring any atmospheric losses in this system which would likely create a slower time to reach an equilibrium temperature than we are seeing in our 10 second analysis. This would actually provide some benefit to our desired solution; more losses would mean lower bulk temperature.

Part 5) Any drawbacks in this part of the analysis would come from error in the Bartz and Seider-Tate equations. As stated previously, we have no real way of validating our HTC calculations, but frankly, these are the best we have.

Conclusions:

In conclusion, based on my results, we find that this system would likely be infeasible due to the very high flow rates needed to successfully cool the copper nozzle below its melting temperature. As stated previously, there was likely an error in part 5 on my end regarding the calculations of the equilibrium temperatures with respect to changes in flow rates. If fixed, I still believe this conclusion would hold true, however.

In class, we discussed the use of Inconel for the real part construction. Inconel has a melting temperature of approximately 1700K. From my results, we find that even this material would result in some melting, but if time allowed for me to debug longer, I would likely find that a reasonable flow rate would effectively cool the system.

If I were to go through this project again, I would probably go about the process a bit differently. I would try to balance my time for the later parts and the earlier parts more evenly. I found that my biggest issue was that I wanted to perfect my scripts for parts 1-3 so much, that by the time I got to 4 and 5, I was getting pretty burned out with coding. The analyses later on could have gone a lot smoother if I wasn't already running on fumes. Other than that, I feel a sense of accomplishment after finishing. This project felt very real and not just some textbook style problem.

If there are any questions regarding the scripts provided above, please feel free to let me know, I realize reading through them may be pretty unclear as to what exact calculations were done. I tried being clear but there may be some lines uncommented that should have been.