```
% ME 140 Project #5
% FUEL CELL EVALUATION & HYRDOGEN PRODUCTION ANALYSIS
% Frankie Willcox, Jon Renslo, Kendall Fagan, Emily Bohl, Natasha Berk

%Jon's todo list
% double check power loss (inefficiencies)
% ask about starting from STP (extra methane used?)

% ASSUME:
% (i)  mol_H2 = 1
clear; close all; clc;
format compact;
entireTime = tic;

global PERMIN_TO_PERSEC PERHR_TO_PERSEC G_PER_KG LHV F N_TO_O
 SCF_TO_MOLS ...
    C_TO_K PSI_TO_PA MM_h MM_h2 MM_o MM_n MM_ch4 MM_h2o MM_air PATM
 HORSEPOWER_TO_W
defineGlobals();
mol_H2 = 1;
savePlots = 0;
                % 1,2,3,4,5,6,7,8,9,10,11

supressplots =   [1,      1,    1,  1];         % supresses plots by
 section

% Part A, Section 1
% Currents (load & stack)
i_load =  [0.00 15.06 27.25 36.48 45.1 52.1 56.3 57.6 56.4];       %
 [Amps]
i_stack = [4.82 21.40 35.65 47.20 59.8 69.7 77.0 79.0 80.0];


% Potentials (load & stack)
v_load =  [17.07 15.05 14.08 13.10 12.07 11.27 10.31 9.87  9.05 ]; %
 [Volts]
v_stack = [17.09 15.22 14.26 12.98 12.42 11.60 10.73 10.21 9.48];


% Temperatures from Thermocouple Readings [C]
% KEY: (Kendall please fill in with photo you took)
T1_C = [42.8 42.9 46.1 48.5 50.5 52.8 54.8 55.8 56.5];
T2_C =     [42.5 45.8 45.8 48.4 50.3 51.9 53.3 53.9 54.3];
Tstack_C = [40.7 41.3 42.5 42.9 44.6 45.6 46.9 46.9 47.6];

T1 = T1_C + C_TO_K;                                                %
 [K],  T1, air into stack
T2 = T2_C + C_TO_K;                                                %
 [K],  T2, air out of stack
Tstack = Tstack_C + C_TO_K;                                        %
 [K],  metal plates on the stack
```

```
% NOTE: T3-T5 are not needed for now
% T3_C =     [48.0 47.1 48.6 48.9 50.4 51.1 51.2 51.1 51.1];       %
 T3, water reservoir DON'T USE!
% T3 = T3_C + C_TO_K;
T4_C =     [48.0 47.2 48.2 48.9 50.4 51.1 51.2 51.1 51.1];
T5_C =     [40.7 41.3 42.5 42.9 44.6 45.6 46.9 46.9 47.6];
T4 = T4_C + C_TO_K;                                                %
 T4, water into stack
T5 = T5_C + C_TO_K;                                                %
 T5, water into heat exchanger


% Mass Flow Rates (TODO: check what units the mdots should be in)
mdot_total_scf = [0.75 1.10 1.45 1.81 2.55 3.10 3.30 3.25 3.40];   %
 [scf/min]
mdot_fuel_scf =  [2.50 6.20 10.5 14.3 18.2 22.0 24.6 25.0 26.1];   %
 [scf/hr] (standard cubic feet/hour)

mdot_total = mdot_total_scf * SCF_TO_MOLS * PERMIN_TO_PERSEC *
 ( MM_air / G_PER_KG);  % [kg/s]
mdot_fuel = mdot_fuel_scf * SCF_TO_MOLS * PERHR_TO_PERSEC * (MM_h2  /
 G_PER_KG);      % [kg/s]
mdot_h2o = 40 /G_PER_KG;
                % [kg/s]

% Pressures
Pfuel_psi =  [2.9 2.9 3.1 3.3 3.30 3.20 3.00 3.0 3.1];              %
 [psi] (gauge)
Ptotal_psi = [0.2 0.3 0.6 0.7 1.15 1.25 1.35 1.3 1.5];              %
 [psi] (gauge), pressure of combined air and H2O after humidifier
Pfuel =  Pfuel_psi  .* PSI_TO_PA + PATM;                            %
 [Pa]
Ptotal = Ptotal_psi .* PSI_TO_PA + PATM;                            %
 [Pa]

% CALCULATIONS
% ------------
% Power (USE: p = i*v)
% NOTE: "Accessories" include H2O pump, air pump, H2 vent, &
 controller
p_load =  i_load  .* v_load;                                        %
 [W] = [kg*m^2*s^-3], a.k.a. "load" (power delivered to resistor bank)
p_stack = i_stack .* v_stack;
p_access = p_stack - p_load;                                        %
 [W], Acessory Power, i.e. power used to run controls. Pstack-Pload
if(~supressplots(1))
    hold off;
    f1 = figure(1);
    plot(p_load,i_load,p_load,i_stack);
    title('Current as a Function of Load');
    xlabel('Load [Watts]'); ylabel('Current [Amps]');
    legend('I_{load}','I_{stack}','Location','best'); grid on;

    f2 = figure(2);
    plot(p_load,v_load,p_load,v_stack);
    title('Potential as a Function of Load');
    xlabel('Load [Watts]'); ylabel('Potential [Volts]');
    legend('V_{load}','V_{stack}','Location','best'); grid on;

    f3 = figure(3);
    plot(p_load,p_stack,p_load,p_access);
    title('Stack and Accessory Power as a Function of Load');
    xlabel('Load [Watts]'); ylabel('Power [Watts]');
    text(5,50,'Net Power = 0 @ 0 Load');
    legend('P_{stack}','P_{accessory}','Location','best'); grid on;

    f4 = figure(4);
    plot(p_load, mdot_fuel*100, p_load, mdot_total);
    title('Mass Flow Rate as a Function of Load');
    xlabel('Load [Watts]'); ylabel('Mass Flow Rate [kg/s]');
    legend('mdot_{H}*100','mdot_{air}','Location','best'); grid on;

end

% Part A, Section 2

% SOURCE: LEC 8, SLIDES 21 & 22

% 1st & 2nd Law Efficiencies (eta_I & eta_II) & Inefficiencies (Idot)
% Stack
[etaI_stack ,etaII_stack, Idot_stack,lambda_stack,dGstack] =
 findEtas(mdot_total, mdot_fuel, Ptotal, Pfuel, T2, p_stack);

% Entire System (Load)
[etaI_load ,etaII_load, Idot_load,lambda_load,dGload] =
 findEtas(mdot_total, mdot_fuel, Ptotal, Pfuel, T2, p_load);
if(~supressplots(2))
    f6 = figure(6);
    plot(p_load,lambda_load);
    title('Air Equivalent as a Function of Load');
    xlabel('Load [Watts]'); ylabel('Lambda');
    legend('\lambda','Location','best');  grid on;

    f5 = figure(5);
    plot(p_load,etaI_stack,'c',p_load,etaI_load,'bp--',...
        p_load,etaII_stack,'r',p_load,etaII_load,'gp--');
    title('Efficiency as a Function of Load');
    xlabel('Load [Watts]'); ylabel('Efficiency');
    legend('\eta_{I,stack}','\eta_{I,system}',...

  '\eta_{II,stack}','\eta_{II,system}', 'Location','Best');grid on;

    f7 = figure(7);
    plot(p_load,-dGstack-p_stack,'c',p_load,-dGload-p_load,'bp--');
    title('Power Loss/Inefficiences as a Function of Load');
    xlabel('Load [Watts]'); ylabel('Power Loss/Inefficiencies, Idot
 [Watts]');
    legend('Idot_{stack}','Idot_{system}','Location','best'); grid on;
```

```
end

% Part A, Section 3
% Comparing First Law Efficiencies of PEM Fuel Cell with Diesel &
 Hybrid Engines

% Typical modern Diesel engine (eta_disel = 42%) (chose diesel truck
 because it's better than a car and worse than a freight ship)
% Source Efficiency: Slide 3, http://www.sae.org/events/gim/
presentations/2011/RolandGravel.pdf
% Source Horsepower: https://cumminsengines.com/isx15-heavy-duty-
truck-2013#overview
eta_diesel = 0.42;
eta_diesel = linspace(eta_diesel, eta_diesel, length(p_load)); %Make
 it a line instead of points
Wdot_diesel = 400 * HORSEPOWER_TO_W;  % [W]

% Typical gasoline hybrid engine (eta_hybrid = max of 40%)
% Source Efficiency & Horsepower: Toyota Hybrid Vehicles, http://
www.toyota-global.com/innovation/environmental_technology/hybrid/
eta_hybrid = 0.40;
eta_hybrid = linspace(eta_hybrid, eta_hybrid, length(p_load)); % Make
 it a line
Wdot_hybrid = 121 * HORSEPOWER_TO_W;  % [W]

% Calcuate Heat Removal (Qdot) --> 40 g/s necessary only for
% intensive/extensive conversion
Qdot_fuelCell = zeros(length(T4));
for i = 1:length(T4)
    Qdot_fuelCell(i) = hEng(T4(i),'h2o') - hEng(T5(i),'h2o');
end
Qdot_fuelCell_max = max(Qdot_fuelCell);

% Theoretical Number of Fuel Cells Needed
% Finding total power of cell out = load power plus Qdot
powerOut = p_load + Qdot_fuelCell_max;

num_fuelCells_diesel = Wdot_diesel ./ powerOut;
num_fuelCells_hybrid = Wdot_hybrid ./ powerOut;

if(~supressplots(3))
    % Overall First Law Efficiency of the PEM Fuel Cell = Stack
 Efficiency
    f8 = figure(8);
    plot(p_load, etaI_stack, 'c', p_load, eta_diesel, 'b:', p_load,
 eta_hybrid, 'g');
    title('Comparing 1st Law Efficiency: PEM Fuel Cell, Diesel, and
 Gasoline Hybrid');
    xlabel('Load [Watts]'); ylabel('Efficiency, eta_{I}');

 legend('eta_{I,stack}','eta_{I,Diesel}', 'eta_{I,Hybrid}','Location','northwest')
  grid on;
end
```

```
% Comments: To scale this up, we would need somewhere between 280-540
 fuel
% cells to equal the diesel output, and 85-165 fuel cells to equal the
% hybrid output.

% Part B, Section 1
% Part B, Section 1 - Emily & Kendall
% Calculating Kp Values
% SOURCE Kp Formula: LECTURE 14, SLIDE 4

% SMR: CH4 + H2O --> CO + 3H2
% v values are stoichiometric coefficients
v_CO_SMR = 1;
v_H2_SMR = 3;
v_H2O_SMR = 1;
v_CH4_SMR = 1;

% Calculating Kp for SMR
% Nv_CO = mm
% SMRnumKp =

% WGS: H2O + CO --> H2 + CO2
v_H2_WGS = 1;
v_CO2_WGS = 1;
v_H2O_WGS = 1;
v_CO_WGS = 1;

T_B1 = linspace(25, 1200, 100); %Temperature for part B1 = T_B1
T_B1 = T_B1 + C_TO_K;
%NOTE: Standard pressure, is usually defined as 100,000, however in
 energyF
%we have standard presssure as 101300. Because the pressure needs to
%cancel out, I have changed this pressure to 101300, however, we
 should
%perhaps consider changing the reference pressure in energyF to
 100,000Pa.
P_ref = 101300; %This is the pressure defined for standard conditions.
% Standard conditions are what we need because that is what
% the little zero indicates in the equation for g.
R_u = 8.314; %Universal gas constant

%G_reaction = G_products - G_reactants
g_SMR = (gEng(T_B1, P_ref, 'co',v_CO_SMR) + gEng(T_B1,
 P_ref, 'h2',v_H2_SMR)) - ...
    (gEng(T_B1, P_ref, 'h2ovap',v_H2O_SMR) + gEng(T_B1,
 P_ref, 'ch4',v_CH4_SMR));

g_WGS = (gEng(T_B1, P_ref, 'h2',v_H2_WGS) + gEng(T_B1,
 P_ref, 'co2',v_CO2_WGS)) - ...
    (gEng(T_B1, P_ref, 'h2ovap',v_H2O_WGS) + gEng(T_B1,
 P_ref, 'co',v_CO_WGS));

%Lecture 13 - Slide 15
kp_SMR = exp(-g_SMR ./ (R_u .* T_B1)); %increases with temp
```

```
kp_WGS = exp(-g_WGS ./ (R_u .* T_B1)); %decrease with temp


%functions for convenience
f_kp_SMR = @(T_B1) exp(-((gEng(T_B1, P_ref, 'co',v_CO_SMR) ...
    + gEng(T_B1, P_ref, 'h2',v_H2_SMR))  ...
    - (gEng(T_B1, P_ref, 'h2ovap',v_H2O_SMR) ...
    + gEng(T_B1, P_ref, 'ch4',v_CH4_SMR)))...
    ./ (R_u.*T_B1));

f_kp_WGS = @(T_B1) exp(-((gEng(T_B1, P_ref, 'h2',v_H2_WGS) ...
    + gEng(T_B1, P_ref, 'co2',v_CO2_WGS)) ...
    -(gEng(T_B1, P_ref, 'h2ovap',v_H2O_WGS) ...
    + gEng(T_B1, P_ref, 'co',v_CO_WGS))) ...
    ./ (R_u.*T_B1));

%Prep for plot
%convert back to celcius
T_B1 = T_B1 - C_TO_K;
%find index of where kp=10^-3 and kp = 10^3, as the problem asks that
 we
%limit the graph to this range
[~,i_min_SMR] = min(abs(kp_SMR - 10^-3));
[~,i_max_SMR] = min(abs(kp_SMR - 10^3)); %yes, this is supposed to use
 min() to find the max ;P
[~,i_min_WGS] = min(abs(kp_WGS - 10^3));
[~,i_max_WGS] = min(abs(kp_WGS - 10^-3));

[zero_smr,izero_smr] = min(abs(log(kp_SMR)));
[zero_wgs,izero_wgs] = min(abs(log(kp_WGS)));

%Plot
if(~supressplots(3))
    f9 = figure(9);
    kpIsOne = ones(size(T_B1));
    semilogy(T_B1(i_min_SMR:i_max_SMR),
 kp_SMR(i_min_SMR:i_max_SMR), ...
        T_B1(i_min_WGS:i_max_WGS),
 kp_WGS(i_min_WGS:i_max_WGS),T_B1,kpIsOne,'k');
    xlabel('Temperature [C]')
    ylabel('Equilibrium Constant')
    text(T_B1(izero_smr) -300 , 1.5,...
        strcat('SMR equil @ ',num2str(round(T_B1(izero_smr))),'K'));
    text(T_B1(izero_wgs), 1.5,...
        strcat('WGS equil @ ',num2str(round(T_B1(izero_wgs))),'K'));
    legend('SMR', 'WGS')
    title('Part B.1: Equilibrium Constant vs. Temperature')
    ylim([0.001,1000]);
    text(50,5,{'H-Power','Operating Temp','25-100K'})
    grid on
    patch([25,100,100,25],
[10^-3,10^-3,10^3,10^3],'g','FaceAlpha',.5,'EdgeAlpha',0);
    set(gca,'children',flipud(get(gca,'children'))) %puts shading
 beneath lines
```

```
end
% Part B No. 2
% Find the Equilibrium Composition (Mol Fractions) of the Steam
 Methane
% Reformation(SMR) Reaction
% SOURCE Nernst Atom Balance: LECTURE 14, SLIDE 4 (equation in lower
 right corner)
npts = 20;
syms nco nch4 nh2 nh2o;
warning('off','symbolic:numeric:NumericalInstability');

temps = linspace(25,1200,npts);
temps = temps + C_TO_K;
pres = [1,10,100];
soln = zeros(length(temps),4,length(pres));
tic
for i =  1:length(temps)
    t = temps(i);
    parfor j = 1:length(pres)
        p = pres(j);

        warning('off','symbolic:numeric:NumericalInstability');
        eqs = [1  == nco   + nch4;...              carbon atom balance
            10 == nh2*2 + nch4*4 + nh2o*2; ... hydrogen atom balance
            3  == nco   + nh2o;...                oxygen atom balance
            nco.*nh2.^3./(nch4.*nh2o).* ...    Nernst atom balance
            (p ./ (nco + nch4 + nh2 + nh2o).^2) ...
            == f_kp_SMR(t)];
        % 4 eq, 4 unknown
        assume([nco,nch4,nh2,nh2o],'real');
        assumeAlso([nco,nch4,nh2,nh2o] > 0);
        assumeAlso([nco,nch4,nh2,nh2o] < 20);
        [a,b,c,d] = vpasolve(eqs,[nco,nch4,nh2,nh2o],[1,1,1,1]);

        nco_sol(i,j) = double(a);
        nch4_sol(i,j) = double(b);
        nh2_sol(i,j) = double(c);
        nh2o_sol(i,j) = double(d);
        %           soln(i,:,j) = max(double(real([a,b,c,d]));

    end
end
% toc;
%calculate mole fractions from nmols in composition
ntot = nch4_sol + nh2_sol + nh2o_sol + nco_sol;
ych4 = nch4_sol./ntot;
yh2 = nh2_sol./ntot;
yh2o = nh2o_sol./ntot;
yco = nco_sol./ntot;

if(~supressplots(4))
    %unneeded but cool looking plot
    f10 = figure(10);
 plot(temps,nco_sol,'b',temps,nch4_sol,'m',temps,nh2_sol,'g',temps,nh2o_sol,'k');
    legend('CO','CH4','H2','H2O');

    %plot mole fractions
    f11 = figure(11);
    linestyle = {'-','--',':'};
    hold on
    plot(1,0,'-k',1,0,'--k',1,0,':k');
    hold on
    for i = 1:length(pres)
        plot(temps,yco(:,i),strcat(linestyle{i},'b'),...
            temps,ych4(:,i),strcat(linestyle{i},'m'),...
            temps,yh2(:,i),strcat(linestyle{i},'g'),...
            temps,yh2o(:,i),strcat(linestyle{i},'r'));
        hold on
    end
    hold off
    xlabel('Temperature [K]');
    ylabel('Mole Fraction');
    title('Steam Methane Reforming Composition');

 legend('1atm','10atm','100atm','CO','CH4','H2','H2O','location','West');
    %ylim([0.001,1]);
    grid on;
end

% Part B No. 3
% % Equations we'll need:
%  eqs = [       1  == nco2   + nco;...          carbon atom balance
%                4  == nco2*2 + nco + nh2o; ...  hydrogen atom balance
%                6  == nh2*2   + nh2o*2;...      oxygen atom balance
%                nco.*nh2o.^3./(nco2.*nh2).* ... Nernst atom balance
%                   == f_kp_SMR(t)];
%          % 4 eq, 4 unknown
%          [a,b,c,d] = vpasolve(eqs,[nco,nh2o,nco2,nh2],[1,1,1,1]);
syms nco nco2 nh2 nh2o;
%soln_wgs = zeros(length(temps),4,length(pres));
tic
% ***BROKEN***
parfor i = 1:length(temps)
    warning('off','symbolic:numeric:NumericalInstability');
    t = temps(i);

    eqs = [       1  == nco2   + nco;...carbon atom balance
        3  == nco2*2 + nco + nh2o; ...  oxygen atom balance
        10  == nh2*2   + nh2o*2;...      hydrogen atom balance
        (nco2.*nh2)./(nco.*nh2o) ... Nernst atom balance
        == f_kp_WGS(t)];        %(note no pressure term, as nmols same
 on RHS and LHS)
    % 4 eq, 4 unknown
    assume([nco,nh2o,nco2,nh2],'real');
    assumeAlso([nco,nh2o,nco2,nh2] > 0);
    assumeAlso([nco,nh2o,nco2,nh2] < 20);
    [a,b,c,d] = vpasolve(eqs,[nco,nh2o,nco2,nh2],[1,1,1,1]);
    nco_wgs(i) = double(a);
    nh2o_wgs(i) = double(b);
    nco2_wgs(i) = double(c);
    nh2_wgs(i) = double(d);
    %          soln(i,:,j) = max(double(real([a,b,c,d]));

end
% toc;
ntot_wgs = nco_wgs + nh2_wgs + nh2o_wgs + nco2_wgs;
yco2_wgs = nco2_wgs./ntot_wgs;
yh2_wgs = nh2_wgs./ntot_wgs;
yh2o_wgs = nh2o_wgs./ntot_wgs;
yco_wgs = nco_wgs./ntot_wgs;

if(~supressplots(4))

    f12 = figure(12);
    plot(temps,yco_wgs,'b',...
        temps,yco2_wgs,'m',...
        temps,yh2_wgs,'--g',...
        temps,yh2o_wgs,'r');
    legend('CO','CO2','H2','H2O','location','southwest');
    xlabel('Temperature [K]');
    ylabel('Mole Fraction');
    title('Water Gas Shift Composition');
    %ylim([0.001,1]);
    grid on;
end


% Part B No. 4
% Plot exit composition (mol fractions) vs. 3 system stations
 (Reformer,
% Shift Reactor 1, Shift Reactor 2)
% Note: do this for 2 Different Assumptions: (1) isothermal, (2)
 adiabatic
% SMR: CH4 + 3*H2O --> CO + 3*H2 + 2*H2O <-known because all assume
 all
% methane is used
% WGS: CO  + 2*H2O + 3*H2--> ?CO2 + (3+?)H2 + ?CO + ?H2O <- unknown
 because WGS
% doens't go all the way to completition


% NAMING CONVENTIONS:
% Station Location: 1=Reformer, 2 = 1st Shift Reactor, 3 = 2nd Shift
% Assumption:       iso = isothermal, adi = adiabatic
% Inlet/Exit:       in = inlet, ex = exit

% PSEUDO CODE
% Start with Nernst atom balance for WGS reaction to get composition
% "Start with isothermal cases - adiabatic is a whole different beast"
% Assume first WGS uses up all CH4 and goes fully to completion
% Figure out the products from the WGS
% Use isothermal temperature values given to figure out Qdot
% Go step by step through and get products for each following
 reaction, ...
% Take those products and do isothermal calcs on them

% Inlet Temperatures
Tin_C = [800 400 250];    % [C]
Tin = Tin_C + C_TO_K; % [K]

% Exit Temperatures
Tex_iso_C = [800 400 250];
Tex_adi_C = [800 NaN NaN]; %TODO: solve for Tin_adi_C(2) & (3)
Tex_iso = Tex_iso_C + C_TO_K;
Tex_adi = Tex_adi_C + C_TO_K;

% Heat Addition for Isothermal Reaction (Qin, ASSUME: isothermal)
Qin_iso = [NaN NaN NaN];            % [MJ/(kg of reactants)]

% Percent Methane Burned to Heat Reformer (pct_CH4, ASSUME: adiabatic)
pct_CH4 = [NaN]; % Note: only applies to Reformer! Not Shift Reactors!


% Part 1: Isothermal
% find exit compositions
compositions = zeros(4,3); %co;h2o;c02;h2
for i = 1:3
    compositions(:,i) = compositionsFun(f_kp_WGS(Tin(i)));
end

% find heat addition for each component
% WGS: CO  + 2*H2O + 3*H2--> ?CO2 + (3+?)H2 + ?CO + ?H2O
% SMR: CH4 + 3*H2O --> CO + 3*H2 + 2*H2O
% comps[species, stage]. Species order: CO, H20, CO2, H2
Qin = zeros(1,3);
N_H20_in = 3;
N_CH4_in = 1;
h_react =  hEng(Tin(1), 'h2ovap', N_H20_in) + hEng(Tin(1), 'ch4',
 N_CH4_in);
for s = 1:3 % three stages: reformer and two reactors
    h_prod = hEng(Tin(s), 'co', compositions(1,s)) +
 hEng(Tin(s), 'h2ovap', compositions(2,s)) + hEng(Tin(s), 'co2',
 compositions(3,s)) + hEng(Tin(s), 'h2', compositions(4,s));
    Qin(s) = h_prod - h_react;

    if (s == 3) break; end
    h_react = hEng(Tin(s+1), 'co', compositions(1,s)) + hEng(Tin(s
+1), 'h2ovap', compositions(2,s)) + hEng(Tin(s+1), 'co2',
 compositions(3,s)) + hEng(Tin(s+1), 'h2', compositions(4,s));
end
% Qin_MJkg = ?
% TODO: GET Qin IN MJ/KG (CURRENTLY IN J. STORE IN NEW VARIABLE B/C
 Qin IS USED BELOW)
```

```
% PSEUDOCODE APPROACH
% Determine composition of each (CO H20vap CO2 H2) where we calculate
 Qin
% Use molar mass to get kg of each
% Divide Qin by kg total
% Convert J to MJ by dividing by a constant (10^6)

% currently in J/mol of methane reacted
%should be 3Mj/kg
Qin_perkg = Qin / (MM_ch4 / G_PER_KG) /1e6; % J/mol --> MJ/kg


% Part 2: Adiabatic (only shift reactors)
error = 0.0001;
speedFactor = 1000;
T_guess = zeros(1,3);
comps_out_adi = zeros(4,3);
tic
% PROBLEM IS THAT TEMPS ARE JUST CONVERGING TO TEMP AT H_IN - MISSING
% SOMETHING CONCEPTUAL.
% temps = linspace(273,800,40);
% comps_out = zeros(length(temps),4);
% for i = 1:length(temps)
%    comps_out(i,:) = compositionsFun(f_kp_WGS(temps(i)))';
%    h_out(i) = hEng(temps(i),   'co',    comps_out(i,1)) ...
%              + hEng(temps(i), 'h2ovap',comps_out(i,2)) ...
%              + hEng(temps(i), 'co2',   comps_out(i,3)) ...
%              + hEng(temps(i), 'h2',    comps_out(i,4));
% end

% H_in occurs at stage 2
% comps[species, stage]. Species order: CO, H20, CO2, H2
comps_in(:) = compositions(:,1);
tol = 0.0001;
step = 1;
for s = 2:3 % two stages: hot shift reactor, cold shift reactor
    t = Tin(s);
    h_in = hEng(t, 'co', comps_in(1)) ...
        + hEng(t, 'h2ovap', comps_in(2)) ...
        + hEng(t, 'co2',comps_in(3)) ...
        + hEng(t, 'h2', comps_in(4));
    T_guess(s) = Tin(s) + 20;
    comps_out_adi(:,s) = compositionsFun(f_kp_WGS(T_guess(s)));
    h_out = hEng(T_guess(s),   'co',    comps_out_adi(1,s)) ...
          + hEng(T_guess(s), 'h2ovap',comps_out_adi(2,s)) ...
          + hEng(T_guess(s), 'co2',   comps_out_adi(3,s)) ...
          + hEng(T_guess(s), 'h2',    comps_out_adi(4,s));
    dh = h_out - h_in;
    % set up newton raphson variables
    % need to remember previous state for newton raphson
    tlast = Tin(s);
    dhlast = T_guess(s) - tlast;

    while abs(dh/h_in) > tol %use percentage error for robustness
```

```
        dhprime = (dh - dhlast) ./(T_guess(s) - tlast);
        tlast = T_guess(s);
        T_guess(s) = T_guess(s) - dh ./ dhprime;
        comps_out_adi(:,s) = compositionsFun(f_kp_WGS(T_guess(s)));
        dhlast = dh;
        h_out = hEng(T_guess(s), 'co',comps_out_adi(1,s)) ...
            + hEng(T_guess(s), 'h2ovap',comps_out_adi(2,s)) ...
            + hEng(T_guess(s), 'co2',comps_out_adi(3,s)) ...
            + hEng(T_guess(s), 'h2',comps_out_adi(4,s));
        dh = h_out-h_in;
    end
    comps_in = comps_out_adi(:,s);
end
% toc
pctCO = comps_out_adi(1,:)./sum(comps_out_adi);
comps_out_adi(:,1) = compositions(:,1);
y_out_adi = comps_out_adi./repmat(sum(comps_out_adi),4,1);
y_iso = compositions./repmat(sum(compositions),4,1);
% ^SHOULD GET 740, 569 K FOR T_guess

% plot of exit composition vs system station (2x, isothermal and
 adiabatic)
if(~supressplots(4))
f13 = figure(13);
subplot(1,2,1);
bar(y_iso');
xlabel('State, Isothermal');
ylabel('Mole Fraction');
ylim([0,0.8]);
legend('CO', 'H20', 'CO2', 'H2','location','northwest');

subplot(1,2,2);
bar(y_out_adi');
xlabel('State, Adiabatic');
ylabel('Mole Fraction');
legend('CO', 'H20', 'CO2', 'H2','location','northwest');
ylim([0,0.8]);
set(f13, 'Position', [0 0 400 200])

annotation('textbox', [0 0.8 1 0.2], ...
    'String', 'H2 Reformer Outlet Molecular Composition', ...
    'EdgeColor', 'none', ...
    'HorizontalAlignment', 'center',...
    'FontSize',18); % add title to plot manually, subplots don't
 include an overall title
set(f13, 'Position', [300 800 800 400]) %resize plot
end


% Part 3: Heating reformer w/ methane
% find methane used by reformer - CHECK!
molar_mass_meth = 16.043/1000; % [kg/mol]
molar_mass_h2 = 2.016/1000; % [kg/mol]
LHV_meth = 50050e3*MM_ch4/G_PER_KG; % [J/mol]
```

```
LHV_h2 = 120000e3*molar_mass_h2; %[J/mol]
N_meth_burned = Qin(1)/LHV_meth; %moles of methane burned
perc_meth_burned = N_meth_burned./(N_meth_burned+1) * 100; %1 is the
 mole used for the actual reaction

% find LHV ratio - CHECK!
N_meth_rxn = 1;
LHV_ratio_isoth = LHV_h2*compositions(4,3)/(LHV_meth*(N_meth_burned +
 N_meth_rxn)) * 100;
LHV_ratio_adia = LHV_h2*comps_out_adi(4,3)/(LHV_meth*(N_meth_burned +
 N_meth_rxn)) * 100;


% NEED FOR TABLE:
% isothermal:
% -composition of gases exiting reformer and reactors
% -heat addition reqd for isothermal (do delta h energy balance on
 either
% side of each component)
% adiabatic:
% -adiabatic outlet temperatures of last two reactors
% -exit composition for shift reactors (and reformer, but same as
 above)
% heat part:
% -methane burned to heat reformer
% -LHV ratio (efficiency)
%
% PLOT: exit composition for isothermal and adiabatic at each station

if(sum(supressplots)~=4)
    plotfixer();
end
if(savePlots ==1)
     plotfixer();
    if(~supressplots(1))
        saveas(f1,'../plots5/1-CurrentbyLoad','png');
        saveas(f2,'../plots5/2-VbyLoad','png');
        saveas(f3,'../plots5/3-PowerbyLoad','png');
        saveas(f4,'../plots5/4-massbyload','png');
    end
    if(~supressplots(2))
        saveas(f5,'../plots5/5-Eff','png');
        saveas(f6,'../plots5/6-lambda','png');
        saveas(f7,'../plots5/7-PowerLoss','png');
    end
    if(~supressplots(3))
        saveas(f8,'../plots5/8-CompareToGasoline','png');
        saveas(f9,'../plots5/9-KeqbyT','png');
    end
    if(~supressplots(4))
        saveas(f10,'../plots5/10-SMRcompmol','png');
        saveas(f11,'../plots5/11-SMRcomp','png');
        saveas(f12,'../plots5/12-WGScomp','png');
        saveas(f13,'../plots5/13-ReformerComp','png');
    end
end
% toc(entireTime);
```

Published with MATLAB® R2016a