```
%This Program is intended to calculate the power output of a Thermal-
Lag
%Cycle engine

% Nathan Payne Summer 2012 ME4444 Senior Design
clc
clear

%DataMatrix: All variables are stored in respective positons
% 1: Stroke Position Degree 0-360, 0&360=BDC, 180=TDC
% 2: Stroke Position Multiplier, 0-1, 0=TDC, 1=BDC
% 3: Pressure of Hot side of motor (K)
% 4: Pressure of Cold side of motor (K)
% 5: Temperature of permanetly hot (K)
% 6: Temperature of cold side of motor (K)
% 7: Temperature of displaced fluid in hot side (K)
% 8:
% 9: Current total volume of engine (m^3)
% 10:
% 11:
% 12: Energy Transfer due to convection, Permanetly hot fluid (watts)
% 13: Energy Transfer due to convection, Permanetly cold fluid (watts)
% 14: Energy Transfer due to convection, Permanetly displaced fluid
(watts)
% 15: Energy Transfer due to compression, Permanetly hot fluid (watts)
% 16: Energy Transfer due to compression, Permanetly hot fluid (watts)
% 17: Energy Transfer due to compression, Permanetly hot fluid (watts)
% 18: Stroke Mass multiplier, 0-1, 1= mass in hot side, 0= mass in
cold side
% 19:
% 20:
% 21:
% 22: Enthalpy Hot (J/kg)
% 23: Enthalpy Cold (J/kg)
% 24: Enthalpy Displaced (J/kg)
% 25: Density Hot (Kg/m^3)
% 26: Density Cold (Kg/m^3)
% 27: Density Displaced (Kg/m^3)
% 28: Entropy Hot (J/kg/K)
% 29: Entropy Cold (J/kg/K)
% 30: Entropy Displaced (J/kg/K)


%% Inputs-Dynamic

%Miscilaneous Parameters
%cycles to steady state opperation(cycle final values = initial cycle
values)
cyclestoaccuracy=3;
% number of steps in cycle,(ex 72 = incriments of 5 degrees)
steps=100;
```

```
%Engine State
RPM=1000;
Pressure_start=1000; %bar
T_h_wall=600;% outside tempeture of incoming fluid
T_c_wall=300;%Tempeture of water in water jacket, K

%Engine Parameters
L_h=.1;%Length of Hot Side Tube(m);
D_h=.03;% Inner Diameter of Hot Side Collector(m);
thick_h=.004;%thickness of Hot Side Tube (m);
T_h_start=500%starting inside temp of run
% Nozzle
L_n=.02;% Length of Nozzle (m)
% Cold Side of motor
L_clearance=.02;%Clearence between TDC and nozzle in cold (m)
L_stroke=.05;%length of piston stroke
D_c=.03;% Inner Diameter of Cold Side Collector/Piston(m)
H_c=100;% rate of heat transfer(water jacketed aluminum) W/m^2*K
T_c_start=300;



%% Inputs-Static
K_h=401; %Thermal conductivity of hot side, W/m*K (aluminum=250,
copper =401)


%% Coded inputs-engine parameters
CrossSection_Area_h=pi*D_h^2*.25;%cross-section area of hot side of
engine
CrossSection_Area_c=pi*D_c^2*.25;%cross-section area of of piston
SA_h=3.1415*D_h*L_h;%Surface Area of hot side
SA_clearance=3.1415*D_c*L_clearance;
SA_stroke=3.1415*D_c*L_stroke;
V_h_total=CrossSection_Area_h*L_h
V_clearance=CrossSection_Area_c*(L_clearance);%Volume of stroke
V_stroke=CrossSection_Area_c*L_stroke;
V_TDC=V_h_total+V_clearance;
V_BDC=V_h_total+V_clearance+V_stroke;
H_h=K_h/(thick_h*1000);
H_c=100;% number to recomputed at a later time.
time=(RPM/60)/72; %Time per iterative cycle
%% Pre-Cycle while loops


% Functions for HOT program
data=janload('nasa.fit')
species={'N2' 'O2'}
mix=[102.76;16]
Pressure_start_Pa=Pressure_start*100000; %bar to pascal conversion
density_h=density(data,species,mix,T_h_start,Pressure_start_Pa);
%(kg/m3)
```

```
density_c=density(data,species,mix,T_c_start,Pressure_start_Pa);
%(kg/m3)
density_d=density_c;
mass_h_starting=density_h*V_h_total;%always hot (Kg)
mass_c_starting=density_c*(V_stroke+V_clearance);%always cold (Kg)
mass_stroke=V_stroke*((density_c+density_h)/2);
mass_clearance=density_c*V_clearance;
mass_d=0;
mass_total=mass_h_starting+mass_c_starting+mass_d;

%Initial Values @ BDC
DataMatrix=zeros(72,28);
n=1;
DataMatrix(1,2)=1;%BDC
DataMatrix(steps+1,2)=1;
DataMatrix(steps,2)=1;
DataMatrix(1,3)=Pressure_start_Pa;
DataMatrix(1,4)=Pressure_start_Pa;
DataMatrix(1,5)=T_h_start;%hot side starting temp
DataMatrix(1,6)=T_c_start;%cold side starting temp
DataMatrix(1,7)=T_c_start;
DataMatrix(1,8)=0;%Pressure accross nozzle.
DataMatrix(1,18)=1;%Percent of displaced volume
DataMatrix(1,19)=mass_h_starting;
DataMatrix(1,20)=mass_c_starting;
DataMatrix(1,21)=mass_d;
DataMatrix(1,22)=enthalpy(data,species,mix,T_h_start,Pressure_start_Pa
);
DataMatrix(1,23)=enthalpy(data,species,mix,T_c_start,Pressure_start_Pa
);
DataMatrix(1,24)=enthalpy(data,species,mix,T_c_start,Pressure_start_Pa
);
DataMatrix(1,25)=density_h;
DataMatrix(1,26)=density_c;
DataMatrix(1,27)=density_d;
DataMatrix(1,28)=entropy(data,species,mix,T_h_start,Pressure_start_Pa)
;
DataMatrix(1,29)=entropy(data,species,mix,T_c_start,Pressure_start_Pa)
;
DataMatrix(1,30)=entropy(data,species,mix,T_c_start,Pressure_start_Pa)
;


%% INITIAL WHILE LOOP
%DataMatrix is loaded, 0-360 @ increments of 5 degrees
stepsize=360/steps;
n=2;
while n <= steps+2;
    ii=((n-2)*stepsize);
    DataMatrix(n,1)=ii;
    DataMatrix(n,2)=.5-.5*(sin((-90+ii)*(3.1415/180)));
    DataMatrix(n,9)=V_BDC+DataMatrix(n,2)*V_stroke;
```

```
    n=n+1;
end
DataMatrix(1,9)=DataMatrix(2,9);
n=2;
count=1;
while count <= cyclestoaccuracy;

    %####################
    %% MASTER WHILE LOOP
    % Loop runs though cycle of engine
    % Properties of the previous state are used for new calculations.
    % Permanetly cold and displaced fluid are assumed to be turbulent
and
    % fully mixed on the cold side of the engine.
    % Permanetly Hot is assumed to be laminar and seperate from
displaced
    % hot.
    % TL/DR three bodies of fluid are moddled
    %(Permanetly Hot: Hot mass permanetly in motor)
    %(Cold: Permanetly Cold + Displaced Hot blended together in cold
side)
    %(Displaced Hot: Hot fluid moving accross nozzle boundary in hot
side)

    while n<=steps+2;
        %% INPUTS FROM PREVIOUS CYCLE VIA DATA MATRIX
        stroke_position=DataMatrix(n-1,2);%Ratio BDC:1 TDC:0
        P_previous_h=DataMatrix(n-1,3);
        P_previous_c=DataMatrix(n-1,4);
        T_h_previous=DataMatrix(n-1,5);
        T_c_previous=DataMatrix(n-1,6);
        T_d_previous=DataMatrix(n-1,7);
        Delta_Pressure=DataMatrix(n-1,8);
        R_displaced_volume=DataMatrix(n-1,18);
        mass_h_previous=DataMatrix(n-1,19);
        mass_c_previous=DataMatrix(n-1,20);
        mass_d_previous=DataMatrix(n-1,21);
        enthalpy_h_previous=DataMatrix(n-1,22);
        enthalpy_c_previous=DataMatrix(n-1,23);
        enthalpy_d_previous=DataMatrix(n-1,24);
        density_h=DataMatrix(n-1,25);
        density_c=DataMatrix(n-1,26);
        density_d=DataMatrix(n-1,27);
        entropy_h_previous=DataMatrix(n-1,28);
        entropy_c_previous=DataMatrix(n-1,29);
        entropy_d_previous=DataMatrix(n-1,30);

        %Calculations of constants

cv_h=spheat(data,species,mix,T_h_previous)/(spratio(data,species,mix,T
_h_previous)*1000);%Specific Heat constant based on Constant volume
(J/g*K)
```

```
cv_c=spheat(data,species,mix,T_c_previous)/(spratio(data,species,mix,T
_c_previous)*1000);%Specific Heat constant based on Constant volume
(J/g*K)

cv_d=spheat(data,species,mix,T_d_previous)/(spratio(data,species,mix,T
_c_previous)*1000);%Specific Heat constant based on Constant volume
(J/g*K)
        cr_h=spratio(data,species,mix,T_h_previous);
        cr_c=spratio(data,species,mix,T_c_previous);
        cr_d=spratio(data,species,mix,T_d_previous);

cr_blend=(cr_h*mass_h_previous+cr_c*mass_c_previous+cr_d*mass_d_previo
us)/(mass_h_previous+mass_c_previous+mass_d_previous);



        %% CALCULATION OF MASS MOVEMENT AND TEMP MIXTURE: Volume
        % mass multiplier lags behind stroke multipler as a function
of
        % Pressure across nozzle boundary
        % mass movement though nozzle is a function of stroke
displacement
        % and pressure accross nozzle boundary
        if Delta_Pressure == 0;
            P_h_comp=P_previous_h*((DataMatrix(n-
1,9)/DataMatrix(n,9))^(cr_blend));
            P_c_comp=P_h_comp;
            T_h_comp=(T_h_previous)*(P_h_comp/P_previous_h)^(1-
1/cr_h);
            T_c_comp=(T_c_previous)*(P_c_comp/P_previous_c)^(1-
1/cr_c);
            T_d_comp=(T_d_previous)*(P_h_comp/P_previous_h)^(1-
1/cr_d);
        else
        end
        %% CALCULATIONS OF STATE CHANGE DUE TO COMPRESSION
        mass_h=mass_h_starting;
        mass_c=mass_clearance+mass_stroke*(stroke_position);
        mass_d=mass_stroke*(1-stroke_position);
        mass_movement=mass_stroke*abs(DataMatrix(n-1,2)-
DataMatrix(n,2));


        % Tempeture modifier of displaced fluid
        if n <(steps/2) %mass movement from cold to hot side

T_d_comp=T_d_comp*((cv_d*mass_d)/(cv_d*mass_d+cv_c*mass_movement))+T_c
_comp*((cv_c*mass_movement)/(cv_c*mass_d+cv_c*mass_movement));
            %density_d=density(data,species,mix,T_d_comp,P_h_comp);
        end
        if n > (steps/2)% mass movement from hot to cold side
```

```
T_c_comp=T_c_comp*((cv_c*mass_c)/(cv_c*mass_c+cv_d*mass_movement))+T_d
_comp*((cv_d*mass_movement)/(cv_c*mass_c+cv_d*mass_movement));
            %density_c=density(data,species,mix,T_c_comp,P_c_comp);
        end


        %Recalculation for accuracy;
        V_c=V_clearance+(V_stroke*stroke_position);

V_h=(mass_h/density_h)/((mass_h/density_h)+(mass_d/density_d))*V_h_tot
al;
        V_d=V_h_total-V_h;
        R_displaced_volume=(V_h_total-V_d)/(V_h_total);




        %         %% CALCULATIONS OF ENERGY CHANGE DUE TO COMPRESSION

enthalpy_h_compression=enthalpy(data,species,mix,T_h_comp,P_h_comp)*ma
ss_h;%change in enthalpy_per kg

enthalpy_c_compression=enthalpy(data,species,mix,T_c_comp,P_c_comp)*ma
ss_c;%change in enthalpy_per kg

enthalpy_d_compression=enthalpy(data,species,mix,T_d_comp,P_h_comp)*ma
ss_d;%change in enthalpy_per kg


        E_h_compression=enthalpy_h_previous-enthalpy_h_compression;
        E_c_compression=enthalpy_c_previous-enthalpy_c_compression;
        E_d_compression=enthalpy_d_previous-enthalpy_d_compression;

        %% CALCULATIONS OF STATE CHANGE DUE TO CONVECTION
        % always hot
        E_convection_h=(time)*H_h*SA_h*R_displaced_volume*(T_h_wall-
T_h_comp);%jouls gained by heat per kg

        % always cold

E_convection_c=(time)*H_c*(SA_clearance+stroke_position*SA_stroke)*(T_
c_wall-T_c_comp);%jouls gained by heat per kg

        %displaced hot
        if(mass_d>0)
            E_convection_d=(time)*H_h*SA_h*(1-
R_displaced_volume)*(T_h_wall-T_d_comp);%jouls gained by heat per kg
        else
            E_convection_d=0;
        end
```

```
        %% CALCULATION OF PRESSURE CHANGE DUE TO NOZZLE

        Delta_Pressure=0; %Difference of pressure accross nozzle.
        %Always Hot
        %DataMatrix(n,15)=(cv_h*(T_h_previous-T_h_comp))/time; % comp
in watts
        %Always Cold
        %DataMatrix(n,16)=(cv_c*(T_c_previous-T_c_comp))/time; %
Compression in watts


        %% Final Calculations
        T_h_ending=T_h_comp+E_convection_h*cv_h;

T_c_ending=T_c_comp+E_convection_c*cv_c;%+T_delta_c_conduction;
        T_d_ending=T_d_comp+E_convection_d*cv_c;
        P_h_ending=P_h_comp;%*(T_h_ending/T_h_comp);
        P_c_ending=P_c_comp;%*(T_c_ending/T_c_comp);

entropy_h=entropy(data,species,mix,T_h_comp,P_h_comp)*mass_h;%change
in enthalpy_per kg

entropy_c=entropy(data,species,mix,T_c_comp,P_c_comp)*mass_c;%change
in enthalpy_per kg

entropy_d=entropy(data,species,mix,T_d_comp,P_h_comp)*mass_d;%change
in enthalpy_per kg
        entropy_sum=entropy_h+entropy_c+entropy_d;

temp_average=T_h_ending*mass_h+T_c_ending*mass_c+T_d_ending*mass_d;
        %% CURRENT CYCLE STATE OUTPUTS TO DATAMATRIX
        DataMatrix(n,3)=P_h_ending;
        DataMatrix(n,4)=P_c_ending;
        DataMatrix(n,5)=T_h_ending;
        DataMatrix(n,6)=T_c_ending;
        DataMatrix(n,7)=T_d_ending;
        DataMatrix(n,8)=Delta_Pressure; %Pressure accross nozzle, + in
hot side.
        DataMatrix(n,10)=mass_movement;%kg per step
        DataMatrix(n,11)=P_h_ending;
        DataMatrix(n,12)=E_convection_h*mass_h/time;%Heat conduction
though watts
        DataMatrix(n,13)=E_convection_c*mass_c/time;% Cold conduction
though watts
        DataMatrix(n,14)=E_convection_d*mass_d/time;% Cold conduction
though watts
        DataMatrix(n,15)=-E_h_compression*mass_h/time;%Compression
output in watts
```

```
        DataMatrix(n,16)=-E_c_compression*mass_c/time;%Compression
output in watts
        DataMatrix(n,17)=-E_d_compression*mass_d/time;%Compression
output in watts
        DataMatrix(n,18)=R_displaced_volume;% 0-1: 1 fully permanetly
hot
        DataMatrix(n,19)=mass_h;
        DataMatrix(n,20)=mass_c;
        DataMatrix(n,21)=mass_d;
        DataMatrix(n,22)=enthalpy_h_compression;
        DataMatrix(n,23)=enthalpy_c_compression;
        DataMatrix(n,24)=enthalpy_d_compression;

DataMatrix(n,25)=density(data,species,mix,T_h_ending,P_h_ending);

DataMatrix(n,26)=density(data,species,mix,T_c_ending,P_c_ending);

DataMatrix(n,27)=density(data,species,mix,T_d_ending,P_h_ending);
        DataMatrix(n,28)=entropy_h;
        DataMatrix(n,29)=entropy_c;
        DataMatrix(n,30)=entropy_d;
        DataMatrix(n,31)=V_h;
        DataMatrix(n,32)=V_c;
        DataMatrix(n,33)=V_d;
        DataMatrix(n,34)=entropy_sum;
        DataMatrix(n,35)=temp_average;

        n=n+1;

        %percent done
        clc
        Completed =(n+(count-1)*steps)/(cyclestoaccuracy*steps)*100
    end
    count=count+1;
    if cyclestoaccuracy > 1
        DataMatrix(1,:)=DataMatrix(steps-1,:);
        n=2;
    end
end
DataMatrix(:,3:4)=DataMatrix(:,3:4)/100000;
DataMatrix(:,11)=DataMatrix(:,11)/100000;
DataMatrix(:,12:14)=DataMatrix(:,12:14)*100;
cla
clc
%plot
figure(1)
plot(DataMatrix(2:steps-2,1),DataMatrix(2:steps-
2,5),DataMatrix(2:steps-2,1),DataMatrix(2:steps-
2,6),DataMatrix(2:steps-2,1),DataMatrix(2:steps-2,7))
title('Tempeture of fluids')
ylabel('Temperature (K)')
xlabel('Crank position')
```

```
hleg1 = legend('Hot fluid','Cold fluid','Displaced fluid');

figure(2)
plot(DataMatrix(2:steps-2,1),DataMatrix(2:steps-
2,12),DataMatrix(2:steps-2,1),DataMatrix(2:steps-
2,13),DataMatrix(2:steps-2,1),DataMatrix(2:steps-2,14))
title('Heat Flow')
ylabel('W transfered')
xlabel('crank position')
hleg1 = legend('hot fluid-heat','cold fluid-heat','displaced fluid-
heat');

figure(3)
plot(DataMatrix(2:steps-2,1),DataMatrix(2:steps-
2,15),DataMatrix(2:steps-2,1),DataMatrix(2:steps-
2,16),DataMatrix(2:steps-2,1),DataMatrix(2:steps-2,17))
title('Crank Work')
ylabel('W transfered')
xlabel('Crank position')
hleg1=legend('Hot air','Cold air','Displaced air');

figure(4)
plot(DataMatrix(2:steps-2,31),DataMatrix(2:steps-
2,3),DataMatrix(2:steps-2,32),DataMatrix(2:steps-
2,4),DataMatrix(2:steps-2,33),DataMatrix(2:steps-2,11))
title('PV Diagram')
ylabel('Pressure (Bar)')
xlabel('Volume M^3')
hleg1=legend('Hot air','Cold air','Displaced air');

figure(4)
plot(DataMatrix(2:steps-2,31),DataMatrix(2:steps-2,3));
title('PV Diagram')
ylabel('Pressure (Bar)')
xlabel('volume (m^3)')


figure(5)
plot(DataMatrix(2:steps-2,1),DataMatrix(2:steps-
2,2),DataMatrix(2:steps-2,1),(DataMatrix(2:steps-2,18)))
title('Movement of Mass and Piston')
ylabel('Piston displacement vs Air displacement')
hleg1=legend('Piston','Air');

figure(6)
plot(DataMatrix(2:steps-2,1),DataMatrix(2:steps-
2,3),DataMatrix(2:steps-2,1),DataMatrix(2:steps-2,4))
title('Pressure of System')
ylabel('Pressure in Bar')
hleg1=legend('Hot Chamber','Cold Chamber');

figure(7)
```

```
plot(DataMatrix(2:steps-2,34),DataMatrix(2:steps-2,35))
title('T-S Diagram')
ylabel('Mean Tempeture K')
xlabel('s J/Kg/K')


Net_Work=sum(DataMatrix(:,15))+sum(DataMatrix(:,16))+sum(DataMatrix(:,
17))


%Net_Heat=sum(;
```