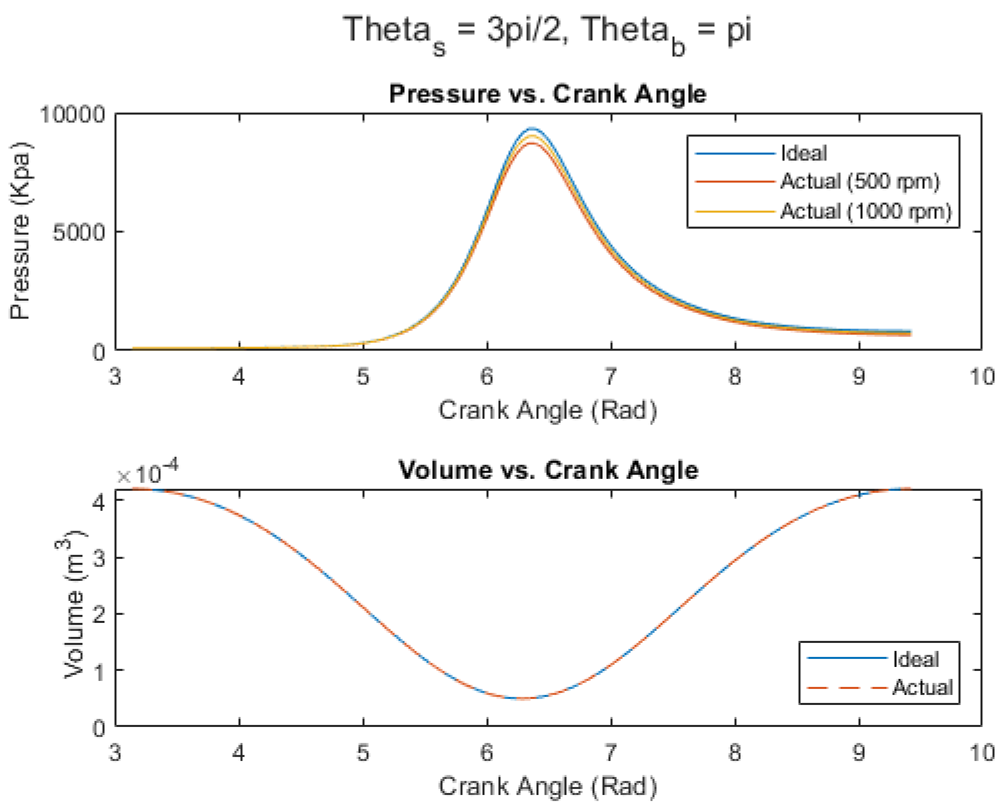
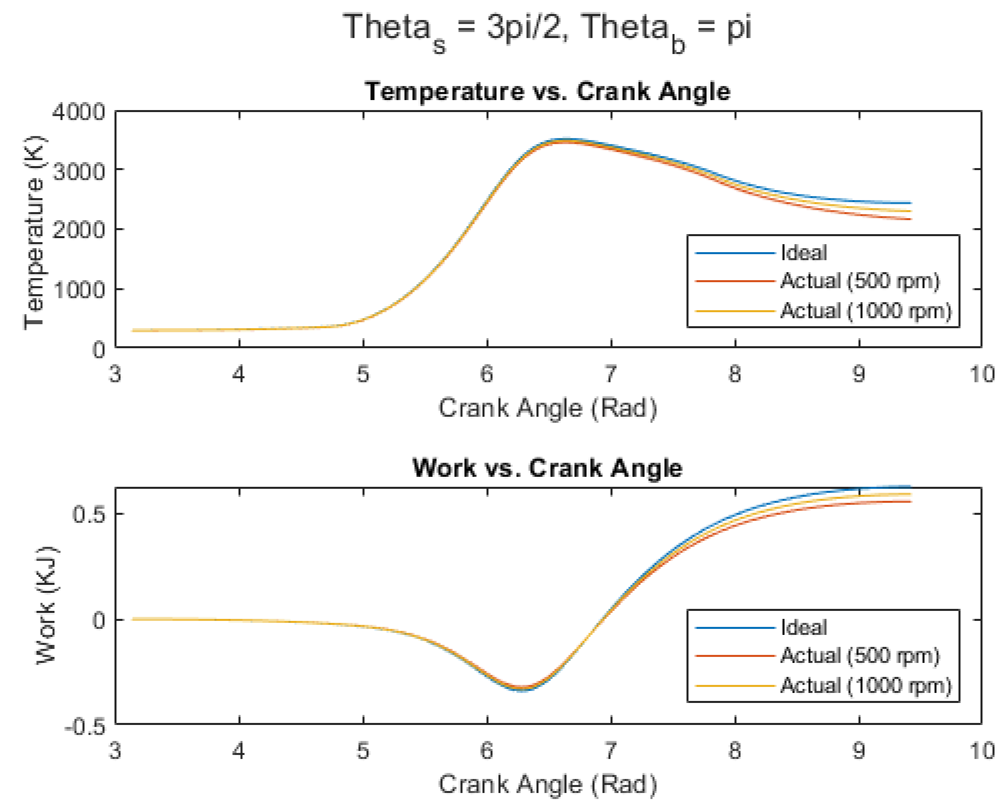
```matlab
% Chloe Blanchard, Carson Cooper, Philip Melton, Trey Nickens, Taylor
% Gautreaux
% Project 2: System of ODEs
% Spring 2023, Professor Sahebsara
clear; close all; clc;

% t = theta for all appearances

% p is our "parameter matrix" containing all given constants and
% information. All of our functions call these constants from the p matrix,
% not necessarily by their names. (EX: the Volume function calls for
% "p(14)" instead of calling for a variable "E")
p = [1.4, .287, 8.4, 101.3, .12, .08, 50*100^(-3), .09, 300, ...
    3*pi/2, pi, 2.8e3, 300, NaN, NaN, 0, 50, 0];

p(14) = (p(6)/(2*p(5))); %calculating E = (s/(2l))
p(15) = p(4)*volume(pi,p)/p(2)/p(9); %calculating the initial mass
% using ideal gas relation

%{
gam = p(1); %specific heat ratio, unitless
R= p(2); %gas constant, kJ/kg/K
r= p(3); %compression ratio, unitless
P1= p(4); %pressure for t = pi, Kpa
l= p(5); %connecting rod length, m
S= p(6); %stroke length, m
V0= p(7); %volume for t = 0, m^3
b = p(8); %piston diameter, m
T1= p(9); %temp for t = pi, K
thetas= p(10); %t for beginning of heat addition, rad
thetab= p(11); %angular interval of heat addition, rad
qin= p(12); %total heat addition, kJ/kg
Tw= p(13); %cylinder wall temperature, K
E= p(14); % (s/(2l)), unitless
M0 = p(15); % mass for t = pi, kg
C = p(16); % mass blowby constant, unitless
omega = p(17); %engine speed, rad/s
hbar = p(18); %convective heat transfer constant, kw/(m^2*K)
%}

% we take symbolic expressions for V and x, and differentiate with respect
% to theta (t) to obtain functions for dV/dt and dx/dt
syms t
symbolicV = (p(7)*(1+(p(3)-1)/(2*p(14))*(1+p(14)*(1-cos(t))- ...
    sqrt(1-p(14)^2*(sin(t))^2))));
dV = diff(symbolicV,t);
dVdt = matlabFunction(dV);

symbolicX = (1/2)*(1-cos(pi*(t-((3*pi)/2))/pi));
dX = diff (symbolicX, t);
dXdt = matlabFunction(dX);
```

```
% Part 1 -----------------------------------------------------------

%initial guess for ODE45 with initial pressure 101.3 kPa, initial work out
%0 kJ
yInitial = [p(4); 0];

% ode45's step size is capped so that the plots are smoother
opt = odeset('MaxStep',0.05);
sol = ode45(@(t,y) steamy(t,y,p,dVdt,dXdt), [pi 3*pi], yInitial,opt);

% Temperature values at desired values of t (theta) are calculated using
% the ideal gas equation, the stored values of pressure (sol.y(1,:)), and
% the volume function.
for i = 1: length(sol.x)
    T(i) = (sol.y(1,i)*volume(sol.x(i),p))/(mass(sol.x(i),p)*(p(2)));
end

% create an array of volume values corresponding to the theta values used
% by ode45 in sol.
for i = 1:length(sol.x)
    vol(i) = volume(sol.x(i), p);
end

% Part 2 ------------------------------------------------------------

% certain parameters are updated in part two for a more realistic model

p(16) = .8; % blow-by constant (1/s)
p(18) = 0.05; % heat loss parameter (kW/(m^2*K))
p(17) = 50; % engine speed (rad/s)


% for omega = 50 rad/s (~500 rpm)
sol2 = ode45(@(t,y) steamy(t,y,p,dVdt,dXdt), [pi 3*pi], yInitial,opt);

for i = 1: length(sol2.x)
    T2(i) = (sol2.y(1,i)*volume(sol2.x(i),p))/(mass(sol2.x(i),p)*(p(2)));
end

    % note ideal and actual plots of volume are exactly the same
for i = 1:length(sol2.x)
    vol2(i) = volume(sol2.x(i), p);
end

% for omega = 100 rad/s (~1000 rpm)
p(17) = 100;
sol3 = ode45(@(t,y) steamy(t,y,p,dVdt,dXdt), [pi 3*pi], yInitial,opt);

for i = 1: length(sol3.x)
    T3(i) = (sol3.y(1,i)*volume(sol3.x(i),p))/(mass(sol3.x(i),p)*(p(2)));
end

% Plots ----------------------------------------------------
% Make tiled layouts
% (1,1)-> P-Theta diagram
% (2,1)-> V-Theta diagram
% (1,1)-> T-Theta diagram
% (2,1)-> W-theta diagram

layout = tiledlayout(2,1);
title(layout,'Theta_s = 3pi/2, Theta_b = pi')

% Pressure vs. Crank Angle
nexttile;
plot(sol.x(:), sol.y(1,:),sol2.x(:),sol2.y(1,:),sol3.x(:),sol3.y(1,:))
% plot(sol2.x(:),sol2.y(1,:))
title  ("Pressure vs. Crank Angle")
xlabel ("Crank Angle (Rad)")
ylabel ("Pressure (Kpa)")
legend('Ideal','Actual (500 rpm)', 'Actual (1000 rpm)')

% Volume vs. Crank Angle
nexttile;
plot (sol.x(:),vol(:))
hold on
plot(sol2.x(:),vol2(:),'--')
title  ("Volume vs. Crank Angle")
xlabel ("Crank Angle (Rad)")
ylabel (" Volume (m^3)")
legend('Ideal', 'Actual',Location='southeast')

figure();
layout2 = tiledlayout(2,1);
title(layout2,'Theta_s = 3pi/2, Theta_b = pi');

% Temperature vs. Crank Angle
nexttile;
plot (sol.x(:),T(:),sol2.x(:),T2(:),sol3.x(:),T3(:))
title  ("Temperature vs. Crank Angle")
xlabel ("Crank Angle (Rad)")
ylabel ("Temperature (K) ")
legend('Ideal','Actual (500 rpm)', 'Actual (1000 rpm)',Location='southeast')

% Work vs. Crank Angle
nexttile;
plot(sol.x(:),sol.y(2,:),sol2.x(:),sol2.y(2,:),sol3.x(:),sol3.y(2,:))
title  ("Work vs. Crank Angle")
xlabel ("Crank Angle (Rad)")
ylabel ("Work (KJ)")
legend('Ideal','Actual (500 rpm)', 'Actual (1000 rpm)',Location='southeast')

% -------------------------------------------------------------

function dy = steamy(t,y,p,dVFunc,dXFunc)
% y(1) = pressure (kPa), y(2) = work,out (kJ);
% takes input column matrix y = [pressure; workOut; ...] and outputs
% the column vector dy/dt = [dP/dt; dW/dt; d.../dt], also takes p, the
% parameter matrix

% dVdt and dXdt are defined anonymously by the "matlabFunction" command,
% and cannot be  directly accessed from within this function. They must
% be supplied as inputs to this function.

% temperature (K) is calculated for use in the fourth term  of the dy(1)
% expression below, which represents heat transfer to the cylinder walls
temp = (y(1)*volume(t,p))/(mass(t,p)*p(2));

dy = zeros(2,1);

% expression for dP/dt
dy(1) = -p(1)*(y(1,1)/volume(t,p))*dVFunc(t) + ... % work term
    ...
    (p(1)-1)*mass(t,p)*p(12)*dheatfraction(t,p,dXFunc)/volume(t,p) ...
    ... % heat addition term ^
    +(-p(1)*p(16)*y(1)) / p(17) - ... % mass blow-by term
    ...
    ((p(1)-1)*p(18)*4*(temp-p(13))) / (p(8)*p(17)); % heat loss term

% expression for dW/dt
dy(2) = y(1,1) * dVFunc(t);

end

% --------------------------------------------------------------

function m = mass(t,p)
% takes two inputs: t (crank angle, radians); and p, the matrix of
% parameters

% outputs the instantaneous mass (kg) of gas contained in the cylinder,
% which is a function of the blow-by constant "C" (p(16)), the engine speed
% "omega" (p(17)), and the initial mass M0 (p(15))

m = p(15)*exp(-(p(16)/p(17))*(t-pi));
end

% ----------------------------------------------------------------

function V = volume(t,p)
% takes two inputs: t (crank angle, radians); and p, the matrix of
% parameters

% outputs the instantaneous volume of the cylinder in m^3, given by the
% expression below.

V= p(7)*(1 + (p(3)-1)/(2*p(14)) * (1+p(14)*(1-cos(t)) - ...
    sqrt(1-p(14)^2*(sin(t))^2)));
end

% ----------------------------------------------------------------
```

```
function x = heatfraction(t,p)
% takes two inputs: t (theta) in in radians (the crank angle); and p, the
% matrix of parameters

% outputs the instantaneous value of x (unitless), the fraction of the
% total heat that has been added at a particlular value of t.

if (pi<=t) && (t<(p(10)))
    % if t is less than the given angle at which "combustion" begins, the
    % fraction x = 0, indicating no heat has been added to the system
    x = 0;

elseif ((p(10))<=t) && (t<= ((p(10))+p(11)))
    % if t is between the starting and ending angles for the combustion
    % process, with the end angle is given by the start angle (thetaS) plus
    % the interval of heat addition (thetaB), then the value of x is given
    % by the expression below
    x = (1/2)*(1-cos(pi*(t-(p(10)))/p(11)));

elseif (( p(10) + p(11) <t)) && (t< 3*pi)
    % a theta greater than the ending angle indicates all heat has been
    % added, x = 1.
    x = 1;
end

end

% ----------------------------------------------------------------

function dxdt = dheatfraction(t,p,dXFunc)
% takes three inputs: t (theta) in radians; p the parameters matrix from
% which we can call all necessary constants, and the function for dX/dt
% (found in the script using the "diff" command)

% outputs dxdt (units 1/rad), the rate of change of the heat fraction with
% respect to theta for a given value of t. dxdt = 0 indicates no heat is
% being added.

if (pi<=t) && (t<(p(10)))
    dxdt = 0;
elseif ((p(10))<=t) && (t<= ((p(10))+p(11)))
    dxdt = dXFunc(t);
elseif (( p(10) + p(11) <t)) && (t<= 3*pi)
    dxdt = 0;
end

end
```


Theta$_s$ = 3pi/2, Theta$_b$ = pi
Pressure vs. Crank Angle
Pressure (Kpa)
10000
5000
0
3
4
5
6
7
8
9
10
Crank Angle (Rad)
Ideal
Actual (500 rpm)
Actual (1000 rpm)
Volume vs. Crank Angle
$\times 10^{-4}$
Volume (m^3)
4
3
2
1
0
Crank Angle (Rad)
Ideal
Actual


Theta$_s$ = 3pi/2, Theta$_b$ = pi
Temperature vs. Crank Angle
Temperature (K)
4000
3000
2000
1000
0
3
4
5
6
7
8
9
10
Crank Angle (Rad)
Ideal
Actual (500 rpm)
Actual (1000 rpm)
Work vs. Crank Angle
Work (KJ)
0.5
0
-0.5
Crank Angle (Rad)
Ideal
Actual (500 rpm)
Actual (1000 rpm)

Published with MATLAB® R2022a