POLITECNICO DI TORINO

MSc Automotive Engineering

A.Y. 2023/24

Propulsion systems and their applications to vehicles

Laboratory report

Carlo Vittorio Colucci 329703

February 27, 2024

Contents

1 Laboratory 1: Flywheel dimensioning

The goal of this laboratory is to design an engine flywheel for both a single cylinder and a multi-cylinder engine, starting from the characterization of the reference air-fuel cycle.

1.1 Parameters setup

Different engine and flywheel reference parameters are provided from the outset:

```
Ncyl=4; % number of cylinder [-]
B=70.8; % bore [mm]
S=78.95; % stroke [mm]
rc=11; % compression ratio [-]
LAM_rod=0.306; % crank slider parameter [-]
n=6000; % engine speed [rpm]
mrec=1.2; % reciprocating masses [kg/dm^3]
mrot=0.85; % equivalent rotating masses [kg/dm^3]
rho_met=7.7; % density of the metal of the flywheel [kg/dm^3]
delta_k = 0.01; % kinematic irregularity [-]
```

Some starting data relative to the reference air-fuel cycle are provided as well:

- Data for calculation of Point 1:

```
pa=100; % ambient pressure [kPa]
Ta=293; % ambient temperaure [K]
lambdav=0.82; % volumetric efficiency [-]
pr_ratio=1.15; % residual to environmental pressure pr/pa [-]
Tr=900; % residual gas temperature [K]
alfast=14.7; % stoichiometric air-fuel ratio [-]
lam=0.95; % relative air-fuel ratio [-]
cf=2500; % fuel specific heat [J/(kg*K)]
xf=1; % fuel vaporized fraction mf(vap)/mf [-]
rf=320; % fuel vaporization heat [kJ/kg]
deltaT=30; % temperature increment for the air-fuel-residual
   mixture during the intake stroke [K]
Ra=287.2; % air elastic constant [J/(kg*K)]
Rr=288; % residuals elastic constant [J/(kg*K)]
R1=271; % air-fuel-residual mixture elastic constant [J/(kg*K)]
cpa=1009; % air specific heat at constant pressure [J/(kg*K)]
```

```
cpr=1150; % residuals specific heat at constant pressure
   [J/(kg*K)]
```

- Data for calculation of point 2:

```
m_com=1.35; % compression polytropic index
cvQLHVu=744; % air specific heat at constant volume [J/(kg*K)]
cvQLHVb=824; % burned gas specific heat at constant volume
   [J/(kg*K)]
QLHVvT0=44000; % lower heating value at constant volume at the
   reference temperature T0 [kJ/kg]
T0=288; % reference temperature T0 [K] for lower heating value
```

- Data for calculation of point 3:

```
dq=0.54; % coefficient of dissociation heat [J/(kg*K^2)]
Td=1850; % starting temperature of disoociation [K]
Rb=288; % burned g
% as elastic constant [J/(kg*K)]
cpb=1328; % burned gas specific heat at constant pressure
   [J/(kg*K)]
delta_HT=0.06; % heat transfer coefficient [-]
```

- Data for calculation of point 4:

```
m_exp=1.27; % expansion polytropic index
```

1.2 Air-fuel cycle characterization

The first step of the design is the characterization of the reference air-fuel cycle by evaluating the thermodynamic state in its key-points:

```
alpha = lam*alfast;
alpha_first = alpha*pr_ratio*(1/(rc-1))*(Ra/Rr)*(Ta/Tr)*(1/lambdav);
      % [-]
T1 = deltaT + ((cpa*Ta*alpha + cf*Ta + alpha_first*cpr*Tr
   -xf*rf*1000)/(alpha*cpa + cf + alpha_first*cpr)) % [K]
p1 = pa*1000*(lambdav*(rc-1)*((1+alpha)/alpha)*(1/(Ra*Ta)) +
   pr_ratio*(1/(Rr*Tr)))*((R1*T1)/rc) % [Pa]
```

```
p2 = p1 * rc^m_com  % [Pa]
T2 = T1 * rc^(m_com-1)  % [K]
```

The lower heating value in point 2 is computed as well:

```
QLHVvT2 = QLHVvT0*1000 + (cvQLHVu - cvQLHVb)*(T2-T0)*(alfast+1)
```

$Q_{LHV,2} = 43.4 MJ/kg$;

```
Acoeff = dq;
Bcoeff = cvQLHVb - 2*dq*Td;
Ccoeff = -cvQLHVb*T2 + dq*Td^2 -
   (1-delta_HT)*lam*(QLHVvT2/(1+alpha+alpha_first));
T3_1 = (-Bcoeff + sqrt(Bcoeff^2 - 4*Acoeff*Ccoeff))/(2*Acoeff); % [K]
T3_2 = (-Bcoeff - sqrt(Bcoeff^2 - 4*Acoeff*Ccoeff))/(2*Acoeff); % [K]
T3 = max(T3_1,T3_2) % [K]
p3 = p2*T3*Rb/(T2*R1)   % [Pa]
p4 = p3 * rc^(-m_exp)   % [Pa]
T4 = T3 * rc^(1-m_exp)  % [K]
```

Summarising the thermo-dynamic state in the key-points:

Point	Pressure [KPa]	Temperature [K]
1	330.1	88.5
2	764.2	2254.4
3	2964.8	9295.1
4	1551.7	442.3

Table 1: Cycle relevant points

To graphically represent the cycle it's necessary to describe the pressure to volume functions along the cycle:

```
Vd = S * pi * (B^2) / 4;    % [mm^3]
Vc = Vd/(rc-1); % [mm^3]
pResidual = pr_ratio * pa * 1000;    % [Pa]
V1 = Vd +Vc % [mm^3]

Vgas = [];
pGas =[];

```

```
for CAcycle = 0:720
    cosBeta = sqrt(1 - (LAM_rod^2) * (sin(CAcycle*pi/180))^2);  % [-]
    x = (S/2)*((1-cos(CAcycle*pi/180)) + (1/LAM_rod)*(1-cosBeta));
       % [mm]
    V = Vc + x*((pi/4)*B^2);     % [mm^3]
    if CAcycle <= 180
        p34 = p3*((Vc/V)^m_exp);  % [Pa];
        pGas = [pGas; p34*10^-5];   % [bar]
        Vgas = [Vgas;V*10^-6];     % [dm^3]
    elseif CAcycle > 180 && CAcycle <= 360
        Vgas = [Vgas;V*10^-6]; % [dm^3]
        pGas = [pGas; pResidual*10^-5]; % [bar]
    elseif CAcycle > 360 && CAcycle <= 540
        Vgas = [Vgas;V*10^-6]; % [dm^3]
        pGas = [pGas; p1*10^-5];     % [bar]
    elseif CAcycle > 540 && CAcycle < 720
        p12= (p1*((V1/V)^m_com)); % [Pa]
        pGas = [pGas;p12*10^-5];   % [bar]
        Vgas = [Vgas;V*10^-6]; % [dm^3]
    elseif CAcycle == 720
        pGas = [pGas; p3*10^-5];     % [bar]
        Vgas = [Vgas;Vc*10^-6]; % [dm^3]
    end
end

figure('Name','Air-fuel cycle')
plot(Vgas,pGas); grid minor
xlabel('Volume [L]');
ylabel('Pressure [bar]');
legend('Air-fuel cycle')
title('Air-fuel cycle')
set(gca,'fontsize', 20)
```

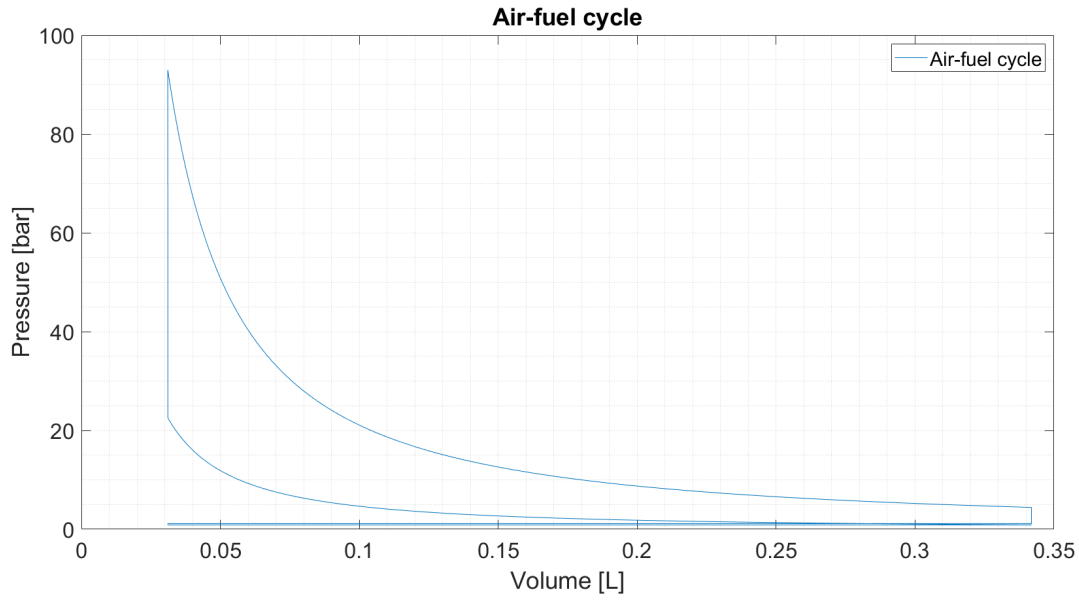

Figure 1: Air-fuel Cycle

The indicated mean effective pressure is a relevant indicator of the cycle performance capabilities:

```
imep = p1*(10^-5)*(rc/(rc-1))*(1/(1-m_com))*(rc^(m_com-1) - 1) +
   p3*(10^-5)*(1/(rc-1))*(1/(1-m_exp))*(rc^(1-m_exp) -1) +(p1 -
   pResidual)*10^-5  % [bar]
```

$imep = 12.5bar$.

1.3 Single cylinder pressures evaluation

Once the pressure trend along the cycle is described, starting from the computation of the inertia pressure acting on the piston, the effective pressure and the resistant pressure (with the relative works) are evaluated as a result:

```
u = 2*(S)*(n/60);    % [mm/s]
rConrod = S/2;  % [mm]
lConrod = rConrod/LAM_rod;  % [mm]

inertiaPressure = [];
tangentialCoefficient = [];
```

```
for CAcycle = 0:720
    beta = asin(LAM_rod*sind(CAcycle)); % [rad]
    inertiaPressureCAi =
       -(mrec*10^3*((pi^2)/2)*(u*10^-3)^2)*(cosd(CAcycle) +
       LAM_rod*(cosd(2*CAcycle))/cos(beta))*10^(-5);   % [bar]
    tangentialCoefficientCAi = sin(beta + deg2rad(CAcycle)) /
       cos(beta);    % [-]
    inertiaPressure = [inertiaPressure; inertiaPressureCAi]; % [bar]
    tangentialCoefficient = [tangentialCoefficient;
       tangentialCoefficientCAi];  % [-]
end

effectivePressure = pGas - pa*10^-2 + inertiaPressure;  % [bar]
tangentialPressure = effectivePressure.*tangentialCoefficient;  %
   [bar]

crankShaftCycle = linspace(0,720,721);
figure;
plot(crankShaftCycle,pGas)
hold on
plot(crankShaftCycle,inertiaPressure)
plot(crankShaftCycle,effectivePressure),grid minor
xlabel('\theta [ CA ]');
ylabel('Pressure [bar]');
legend('Gas pressure','Inertia pressure','Effective Pressure')
title('Pressure contributions')
set(gca,'fontsize', 20)
```

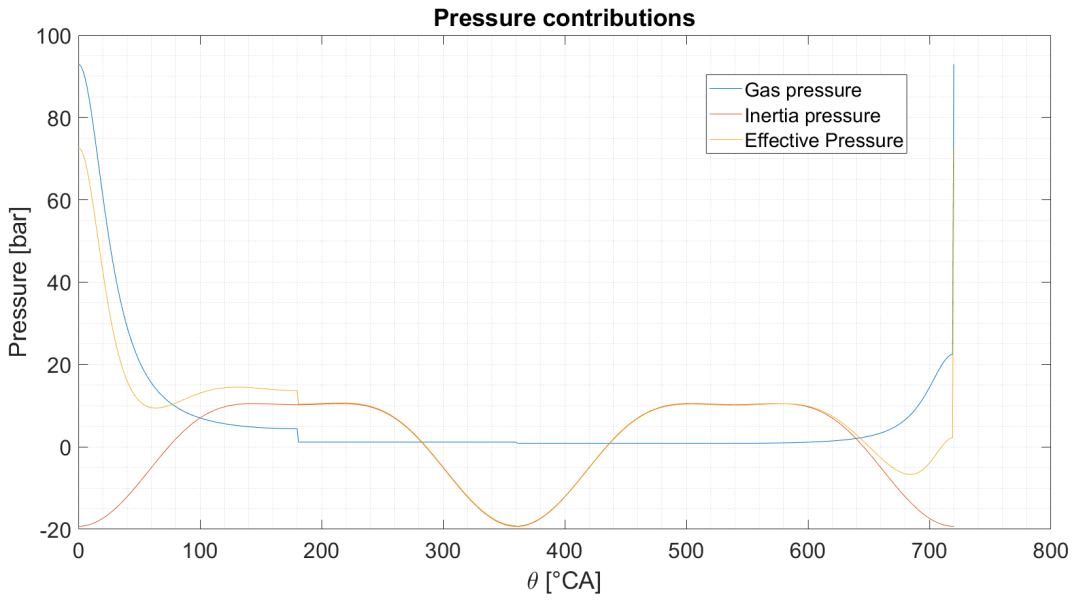

Figure 2: Pressure contributions

```
figure;
plot(Vgas,pGas); hold on
plot(Vgas,-inertiaPressure); grid minor
xlabel('Volume [L]');
ylabel('Pressure [bar]');
lgd = legend('Air-fuel cycle','Inertia pressure');
title('Inertia pressure trend')
set(gca,'fontsize', 20)
```

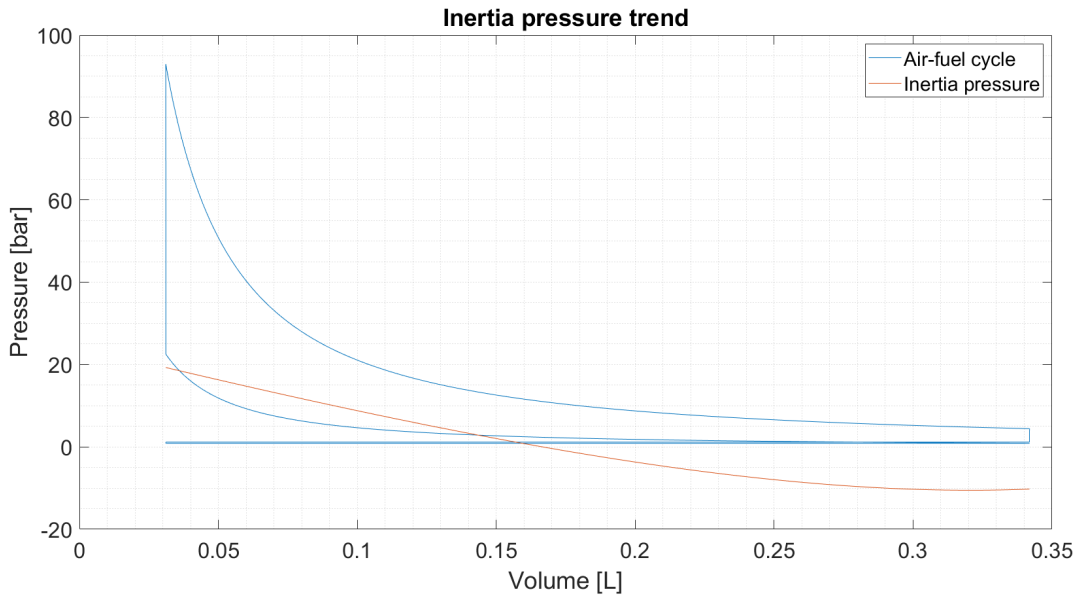

Figure 3: Inertia pressure trend

```
Mcs = tangentialPressure * Vd/2 * 10^-4;     % [N*m]
Mr =  ones(721,1) * imep * Vd / (4*pi)* 10^-4;  % [N*m]

resistantPressure = Mr/(Vd/2*10^-4);     % [bar]

McsRatio = max(Mcs) / mean(Mcs) % [-]
```

$M_{CS,ratio} = 9.5$;

```
Wcs = cumtrapz(deg2rad(crankShaftCycle), Mcs);  % [J]
Wr = cumtrapz(deg2rad(crankShaftCycle), Mr);    % [J]

figure;
yyaxis left
plot(crankShaftCycle,tangentialPressure); hold on
plot(crankShaftCycle,resistantPressure); hold on
ylim([-20 50])
ylabel('Pressure [bar]');
yyaxis right
plot(crankShaftCycle,Wcs); hold on
plot(crankShaftCycle,Wr); grid minor
```

```
ylim([-200 500])
xlim([0 720])
ylabel('Work [J]');
xlabel('\theta [ CA ]');
lgd = legend('Tangential pressure','Resistant pressure','Crankshaft work',
    'Resistant work');
title('Tangential VS resistant characteristics')
set(gca,'fontsize', 20)
```

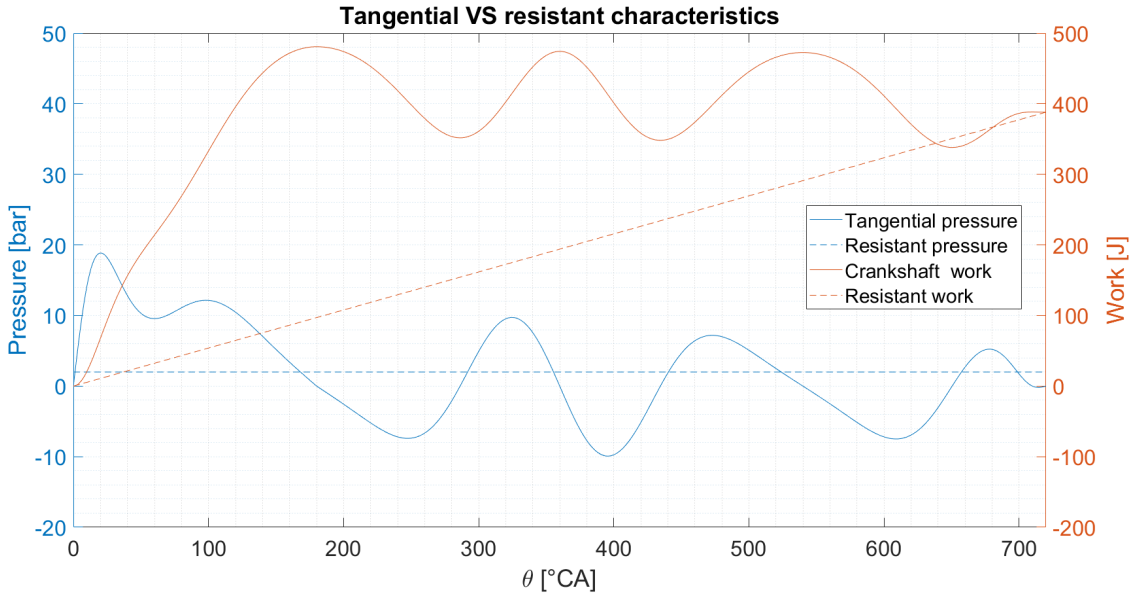

Figure 4: Tangential VS resistant characteristics

1.4 Single cylinder flywheel design

The first relevant parameter for the design is the dynamic irregularity, calculated as follows:

```
dWmax = max(Wcs - Wr);  % [J]
dWmin = min(Wcs - Wr);

dynamicIrregularity = (dWmax+abs(dWmin))/(imep * 10^-4 * Vd)     % [-]
```

$\zeta = 1.0342$;

The next step consists in computing the power train total inertia and the engine one, in order to evaluate the flywheel inertia, needed for the component dimensioning:

```
Wavg = n*pi/30; % [rad/s]
r = S/2;    % [mm]
Jtot = (dynamicIrregularity * imep * Vd *10^-4) / (delta_k *
   Wavg^2);    % [kg*m^2]
Jeng = mrot *10^3 * Vd*10^-9 * (r*10^-3) ^2;    % [kg*m^2]
Jfw = Jtot-Jeng;    % [kg*m^2]
fwDiameter = (Jfw*320 / (pi*rho_met*10^3))^(1/5)    % [m]
L = r/LAM_rod;  % [mm]

if fwDiameter > 5*S*10^-3
    disp('The flywheel diameter is too big: change the shape.')
elseif fwDiameter < 2*S*10^-3
    fwDiameter = (Jfw*32*15 / (pi*rho_met*10^3))^(1/5)  % [m]
    if fwDiameter > 5*S*10^-3
        disp('The flywheel diameter is too big: change the shape.')
    end
else
    disp('The flywheel diameter is fine.')
end
```

$d_{fw} = 26.63cm$;

The final step is the evaluation of the engine speed along the cycle:

```
W = sqrt(Wavg^2 + 2/Jtot * (Wcs-Wr));   % [rad/s]
Wavg1 = 1/(4*pi)*trapz(deg2rad(crankShaftCycle),W); % [rad/s]
shift = Wavg1-Wavg; % [rad/s]
if abs(shift/Wavg) < delta_k
    disp('The angular velocity estimation is good.')
    Wavg1   % [rad/s]
else
    disp('The angular velocity estimation failed: it is computed
       again.')
    W = W-shift;    % [rad/s]
    Wavg1 = 1/(4*pi)*trapz(deg2rad(crankShaftCycle),W); % [rad/s]
    shift = Wavg1-Wavg; % [rad/s]
    if abs(shift/Wavg) < delta_k
        disp('The angular velocity estimation is good.')
```

```
        Wavg1% [rad/s]
    else
        disp('The angular velocity estimation failed.')
    end
end
WavgVector = ones(721,1) * Wavg;     % [rad/s]

figure;
plot(crankShaftCycle,W); hold on
plot(crankShaftCycle,WavgVector); grid minor
xlabel('\theta [ CA ]');
ylabel('Angular speed [rad/s]');
lgd = legend('\omega','\omega_{avg}');
xlim([0 720])
title('Crankshaft angular speed')
set(gca,'fontsize', 20)
```

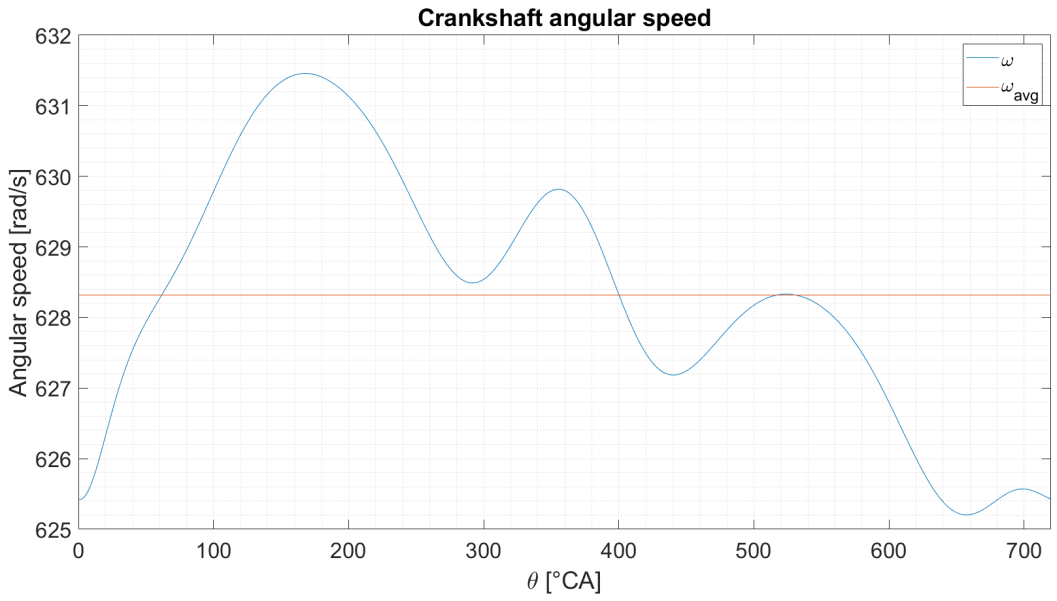

Figure 5: Crankshaft angular speed

1.5 Multi-cylinder pressures evaluation

The multi-cylinder engine procedure for flywheel design is almost similar to the single cylinder case. Thus the starting phase consists in evaluating the inertia, effective and resistant pressures, with the relative works for a 4-cylinders engine:

```
PS = 2*360/Ncyl;    % [ CA ]
piston2coeff = 1;   % [-]
piston3coeff = 3;
piston4coeff = 2;

Mcs1 = Mcs; % [N*m]
Mcs2 = [Mcs(PS*piston2coeff+1:end);Mcs(1:PS*piston2coeff)];
Mcs3 = [Mcs(PS*piston3coeff+1:end);Mcs(1:PS*piston3coeff)];
Mcs4 = [Mcs(PS*piston4coeff+1:end);Mcs(1:PS*piston4coeff)];
McsMC = Mcs1+Mcs2+Mcs3+Mcs4;

tangentialPressure2 = Mcs2/(Vd/2)*10^4; % [bar]
tangentialPressure3 = Mcs3/(Vd/2)*10^4;
tangentialPressure4 = Mcs4/(Vd/2)*10^4;
tangentialPressureMC = McsMC/(Vd/2)*10^4;

figure;
plot(crankShaftCycle,tangentialPressure); hold on
plot(crankShaftCycle,tangentialPressure2); hold on
plot(crankShaftCycle,tangentialPressure3); hold on
plot(crankShaftCycle,tangentialPressure4); hold on
plot(crankShaftCycle,tangentialPressureMC,LineWidth=2); grid minor
xlabel('\theta [ CA ]');
ylabel('Pressure [bar]');
lgd = legend('Cylinder 1','Cylinder 2','Cylinder 3','Cylinder
   4','Multi-cylinder');
xlim([0 720])
title('Multi-cylinder CS tangential pressure contributions')
set(gca,'fontsize', 20)
```

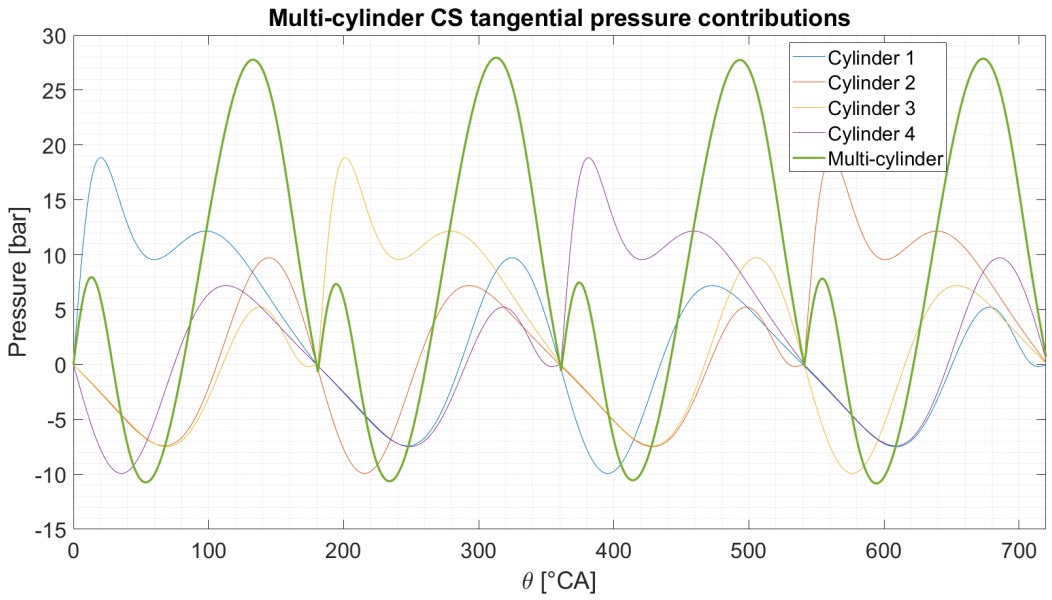

Figure 6: Multi-cylinder CS tangential pressure contributions

```
McsMCratio = max(McsMC) / mean(McsMC)    % [-]
MrMC =  Ncyl * Mr;  % [N*m]
resistantPressureMC = MrMC/(Vd/2)*10^4; % [bar]
WcsMC = cumtrapz(deg2rad(crankShaftCycle),McsMC);    % [J]
WrMC = cumtrapz(deg2rad(crankShaftCycle),MrMC); % [J]

figure;
yyaxis left
plot(crankShaftCycle,tangentialPressureMC); hold on
plot(crankShaftCycle,resistantPressureMC); hold on
ylabel('Pressure [bar]');
ylim([-15 30])
yyaxis right
plot(crankShaftCycle,WcsMC); hold on
plot(crankShaftCycle,WrMC); grid minor
xlabel('\theta [ CA ]');
ylabel('Work [J]');
lgd = legend('Tangential pressure','Resistant pressure','Engine
    work','Resistant work');
xlim([0 180])
```

```
ylim([-150 300])
title('Tangential␣VS␣resistant␣MC␣characteristics')
set(gca,'fontsize', 20)
```

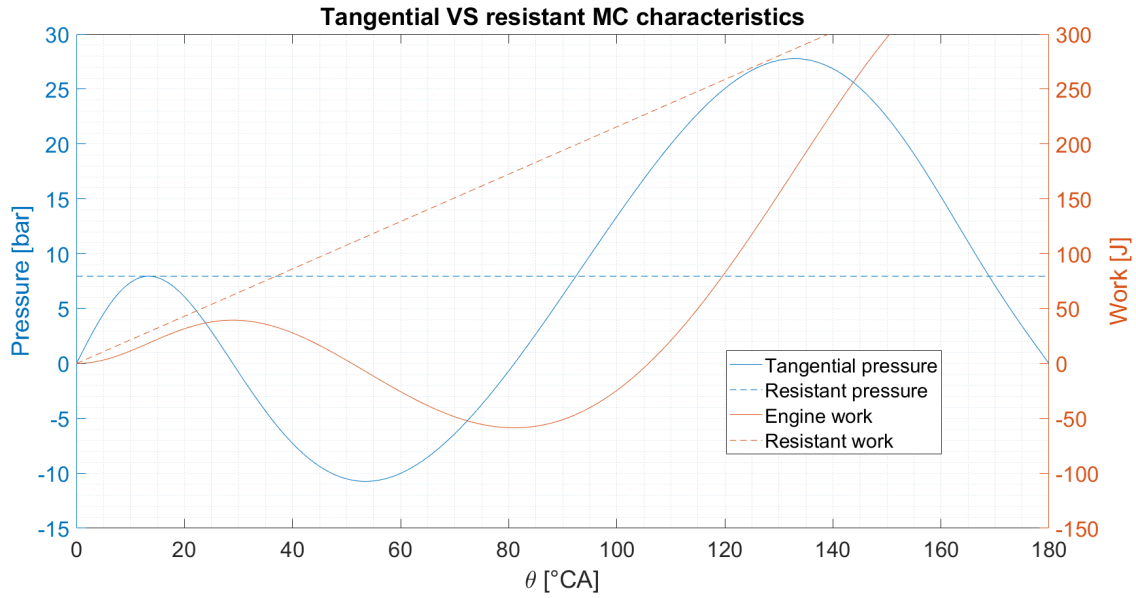

Figure 7: Tangential VS resistant MC characteristics

1.6 Multi-cylinder flywheel design

As in the single cylinder case, dynamic irregularity is the first parameter to be retrieved:

```
dWmaxMC = max(WcsMC - WrMC);
dWminMC = min(WcsMC - WrMC);

dynamicIrregularityMC = (dWmaxMC+abs(dWminMC))/(imep * 10^-4 *
   Vd*Ncyl)     % [-]
if dynamicIrregularityMC<dynamicIrregularity/Ncyl
    disp('The␣dynamic␣irregularity␣for␣the␣multi-cylinder␣engine␣is␣
       verified.')
else
    disp('The␣dynamic␣irregularity␣for␣the␣multi-cylinder␣engine␣is␣
       not␣verified.')
end
```

$\zeta_{MC} = 0.17$;
The expected relation between this value and the single-cylinder relative one is verified:$\zeta_{MC} < \zeta/i$.
Finally the flywheel inertia is computed, as well as its diameter:

```
JtotMC = (dynamicIrregularityMC * Ncyl * imep * Vd *10^-4) /
   (delta_k * Wavg^2);    % [kg*m^2]
JengMC = Ncyl * mrot *10^3 * Vd*10^-9 * (r*10^-3) ^2;    % [kg*m^2]
JfwMC = JtotMC-JengMC;    % [kg*m^2]
fwDiameterMC = (JfwMC*320 / (pi*rho_met*10^3))^(1/5)    % [m]

if fwDiameterMC > 5*S*10^-3
    disp('The flywheel diameter is too big: change the shape.')
elseif fwDiameterMC < 2*S*10^-3
    fwDiameterMC = (JfwMC*32*15 / (pi*rho_met*10^3))^(1/5)  % [m]
    if fwDiameterMC > 5*S*10^-3
        disp('The flywheel diameter is too big: change the shape.')
    end
else
    disp('The flywheel diameter is fine.')
end
```

Also the angular speed of the multi-cylinder engine is evaluated following the method already presented above:

```
WMC = sqrt(Wavg^2 + 2/JtotMC * (WcsMC-WrMC));   % [rad/s]
Wavg1MC = 1/(4*pi)*trapz(deg2rad(crankShaftCycle),WMC); % [rad/s]
shiftMC = Wavg1MC-Wavg; % [rad/s]
if abs(shiftMC/Wavg) < delta_k
    disp('The angular velocity estimation is good.')
    Wavg1MC % [rad/s]
else
    disp('The angular velocity estimation failed: it is computed
       again.')
    WMC = WMC-shiftMC;% [rad/s]
    Wavg1MC = 1/(4*pi)*trapz(deg2rad(crankShaftCycle),WMC); % [rad/s]
    shiftMC = Wavg1MC-Wavg; % [rad/s]
    if abs(shiftMC/Wavg) < delta_k
        disp('The angular velocity estimation is good.')
```

```
        Wavg1MC% [rad/s]
    else
        disp('The angular velocity estimation failed.')
    end
end

figure;
plot(crankShaftCycle,WMC); hold on
plot(crankShaftCycle,WavgVector); grid minor
xlabel('\theta [ CA ]');
ylabel('Angular speed [rad/s]');
lgd = legend('\omega','\omega_{avg}');
xlim([0 180])
title('MC crankshaft angular speed')
set(gca,'fontsize', 20)
```

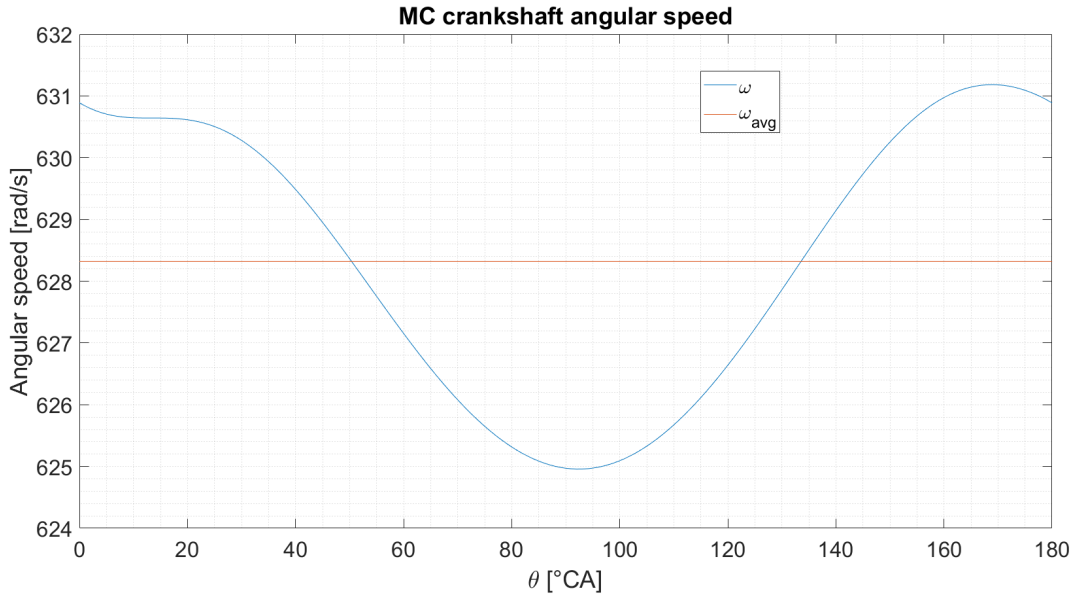

Figure 8: MC crankshaft angular speed

2 Laboratory 2: Engine testing and HRR analysis

The goals of this laboratory are to:

- correctly handle given engine test data collected at variable ambient conditions, bearing them to standard conditions;
- analyse the heat release rate during an engine combustion phase.

The reference engine is a compression ignition one embedded with an exhaust gas recirculation system.

2.1 Parameters setup

All the engine data are setup in the following section:

```
n_engine = SS_data(:,1);% [rpm]
Texp = SS_data(:,2);    % [Nm]
Pexp = Texp .* n_engine * 2*pi/60;
qm_fuel = SS_data(:,3); % [kg/h]
qm_air = SS_data(:,4);  % [kg/h]
qm_egr = SS_data(:,5);  % [kg/h]
pa = SS_data(:,6)/10;  % [kPa]
p0_dry = 99;           % [kPa]
Ta = SS_data(:,7)+273.15;% [K]
T0 = 298;              % [K]
RelativeHumidityAir = SS_data(:,8)/100; % [-]
Ti_MF = SS_data(:,9)+273.15;% [K]
pi_MF = SS_data(:,10)/10;% [kPa]
WSPcoefficients = xlsread(file_name,'Steady-state tests','N3:N7'); %
   [-] Water saturation pressure coefficients
Qlhv = xlsread(file_name,'Steady-state tests','N10');              %
   [MJ/kg]
R = xlsread(file_name,'Steady-state tests','N11');                 %
   [J/(kg*K)]
stroke = xlsread(file_name,'Steady-state tests','N21');            %
   [mm]
bore = xlsread(file_name,'Steady-state tests','N23');              %
   [mm]
```

```
cylinderDisplacement = stroke * pi * (bore^2)/4;                    % [mm^3]
rc = xlsread(file_name,'Steady-state tests','N24');                 % [-]
cylindersNumber = xlsread(file_name,'Steady-state tests','N25');    % [-]
m = 2;
R_air = xlsread(file_name,'Steady-state tests','N14');  % [J/(kg*K)]
cpEGR = xlsread(file_name,'Steady-state tests','N17');  % [J/(kg*K)]
gammaEGR = xlsread(file_name,'Steady-state tests','N18');   % [-]
```

2.2 Correction factor determination

In order to take into account the variable atmospheric conditions and to correct the power data referred to them, obtaining data referred to standard conditions, the ISO 1585 standard is adopted. Correction factors for variable speeds are computed to carry out the conversion:

```
p_satH2O = [];
for i = 1:size(SS_data,1)
    p_satH2Oi = 0;
    for j = 1:length(WSPcoefficients)
        p_satH2Oi = p_satH2Oi +
           WSPcoefficients(j)*(Ta(i)-273.15)^(j-1);    % [kPa]
    end
    p_satH2O = [p_satH2O; p_satH2Oi];   % [kPa]
end

pa_dry = pa - RelativeHumidityAir .* p_satH2O;  % [kPa]

fa_TC = ((p0_dry./pa_dry).^0.7).*((Ta./T0).^1.2);   % [-]
n_engineConverted = n_engine/60; % [rev/s]
n_engineCorrected = n_engineConverted/m; % [cycle/s]
qm_fuelConverted = qm_fuel.*(10^6)./3600; % [mg/s]
engineDisplacement = cylinderDisplacement * cylindersNumber *
   10^(-6);
qf = qm_fuelConverted ./ n_engineCorrected ./ engineDisplacement;
   % [mg/(L*cycle)]
```

```
18
19 qc = qf./((pi_MF+pa)./pa);  % [mg/(L*cycle)]
20
21 fm = [];
22 for i=1:size(SS_data,1)
23     if qc(i) >= 37.2 & qc(i) <= 65
24         fmi = 0.036*qc(i)-1.14; % [-]
25     end
26     if qc(i) < 37.2
27         fmi = 0.2;  % [-]
28     end
29     if qc(i) > 65
30         fmi = 1.2;  % [-]
31     end
32     fm = [fm; fmi]; % [-]
33 end
34
35 CorrectionFactorCI_TC = fa_TC.^fm;  % [-]
36
37 for i=1:length(CorrectionFactorCI_TC)
38     if CorrectionFactorCI_TC<0.9 | CorrectionFactorCI_TC>1.1
39         disp('Error in the correction factor')
40     end
41 end
42
43 figure
44 plot(n_engine, CorrectionFactorCI_TC); grid minor
45 xlabel('n [rpm]')
46 ylabel('\mu_c [-]')
47 xlim([500 4000])
48 ylim([0.98 1.02])
49 title('Correction factor trend')
```

Figure 9: Correction factor trend

Once these factors are available, all the relevant parameters to carry out a proper analysis of the engine performance can be evaluated:

```
Tcorrected = CorrectionFactorCI_TC .* Texp; % [Nm]
Pcorrected = CorrectionFactorCI_TC .* Pexp / 1000; % [kW]

figure
yyaxis left
plot(n_engine, Pexp/1000); hold on
plot(n_engine, Pcorrected); grid minor;
ylabel('P [kW]')
yyaxis right
plot(n_engine, Texp); hold on
plot(n_engine, Tcorrected); hold on
ylabel('T [Nm]')
title('ISO 1585 power adjustment to the environment conditions')
lgd = legend('Uncorrected power', 'Corrected power','Uncorrected
    torque', 'Corrected torque');
xlabel('n [rpm]')
set(gca,'fontsize', 20)

```

```
18
19 eta_f = Pcorrected ./ (qm_fuel/3600 * (Qlhv)*10^3); % [-]
20
21 figure
22 plot(n_engine, eta_f); grid minor
23 xlabel('n␣[rpm]')
24 ylabel('\eta_{f}␣[-]')
25 xlim([500 4000])
26 ylim([0 0.5])
27 title('Fuel␣conversion␣efficency␣trend')
28 set(gca,'fontsize', 20)
29
30 bsfc = qm_fuel * 10^3 ./ Pcorrected;    % [g/kWh]
31
32 figure
33 plot(n_engine, bsfc); grid minor
34 xlabel('n␣[rpm]')
35 ylabel('bsfc␣[g/kWh]')
36 xlim([500 4000])
37 ylim([200 245])
38 title('Brake␣specific␣fuel␣consumption␣trend')
39 set(gca,'fontsize', 20)
40
41 R_egr = cpEGR * (1-1/gammaEGR); % [J/(kg*K)]
42 Rmix = (qm_air/3600*R_air + qm_egr/3600*R_egr) ./
      (qm_air/3600+qm_egr/3600);    % [J/(kg*K)]
43 rho_a = (pa_dry+pi_MF) * 10^3 ./ (Rmix.*Ti_MF);
44
45 lambda_v = (qm_air+qm_egr) ./ (3600 * n_engineCorrected .* rho_a *
      cylindersNumber * cylinderDisplacement*10^(-9)); % [-]
46
47 figure
48 plot(n_engine, lambda_v); grid minor
49 xlabel('n␣[rpm]')
50 ylabel('\lambda_{v}␣[-]')
51 xlim([500 4000])
52 ylim([0 1])
```

```
53 title('Volumetric efficiency trend')
54 set(gca,'fontsize', 20)
```

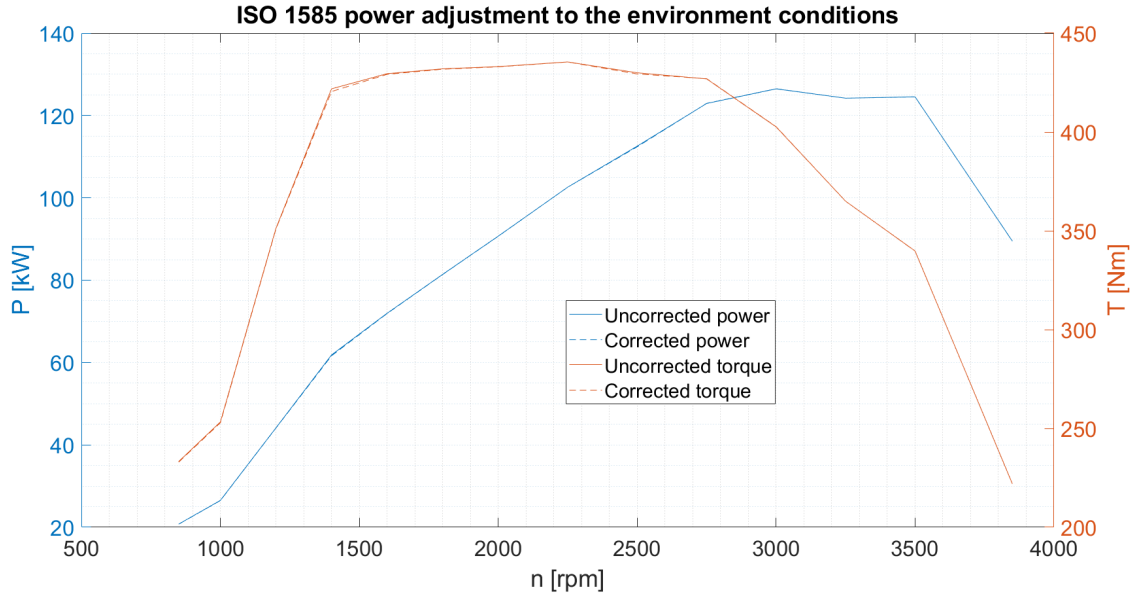

Figure 10: ISO 1585 power adjustment to the environment conditions

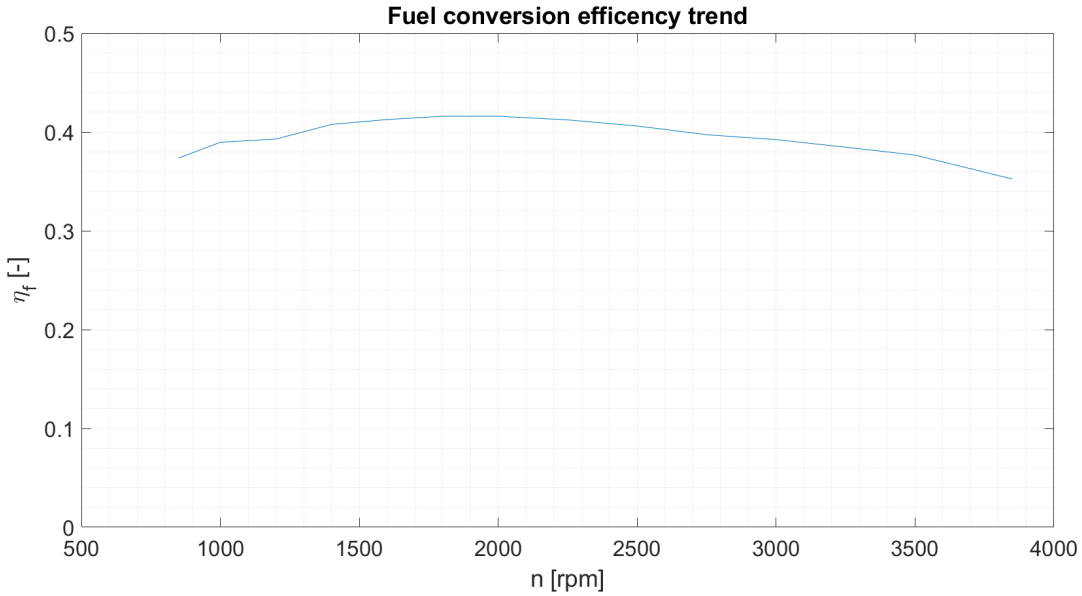

Figure 11: Fuel conversion efficency trend

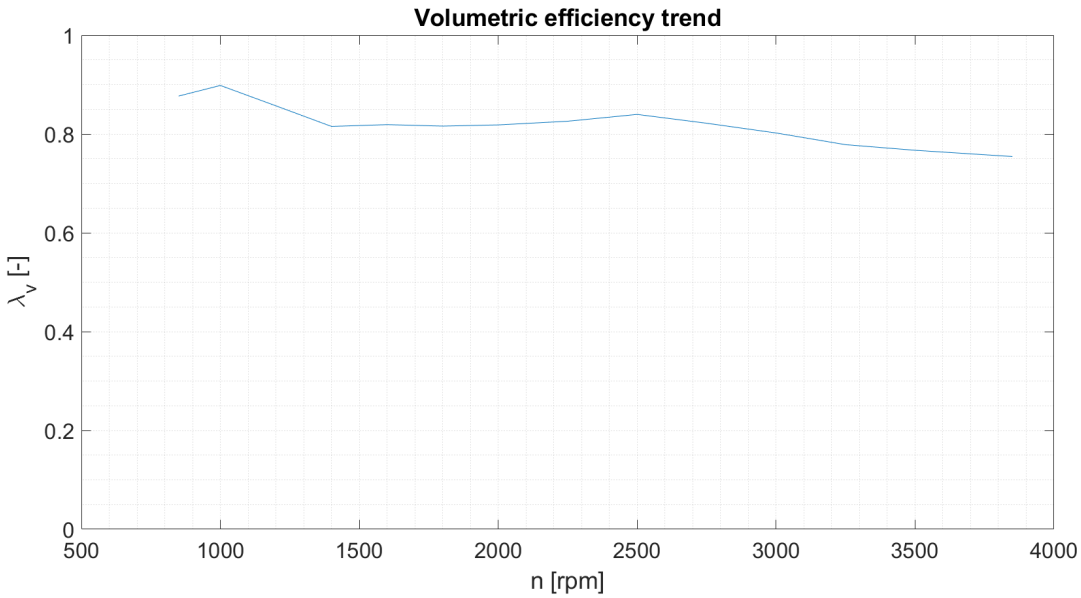

Figure 13: Volumetric efficiency trend

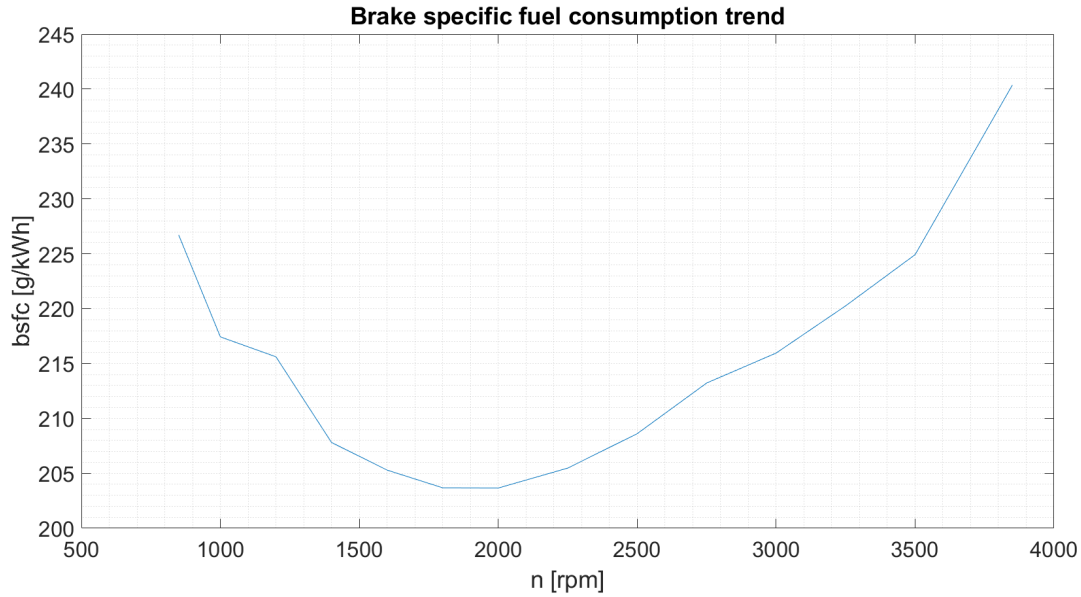

Figure 12: Brake specific fuel consumption trend

2.3 In-cylinder pressure analysis

During the data collection phase, a piezoelectric sensor was positioned inside the combustion chamber, collecting data over a cycle:

```
load('ifile_2000FL');
HRR_data_add = xlsread('datasheet.xlsx', 'HRR point','B:B');
deltaCrank = double(ifile.PMAN1.axis)+360;  % [ CA ]
p_man = double(ifile.PMAN1.data);   % [bar]
p_raw1 = double(ifile.PCYL1.data);   % [bar]
p_raw2 = double(ifile.PCYL2.data);   % [bar]
p_raw3 = double(ifile.PCYL3.data);   % [bar]
p_raw4 = double(ifile.PCYL4.data);   % [bar]
```

The first step consists in scaling the in-cylinder pressure raw data collected, exploiting the intake manifold pressure data:

```
shift1 = mean(p_man(1801,:)) - mean(p_raw1(1801,:));    % [bar]
shift2 = mean(p_man(1801,:)) - mean(p_raw2(1801,:));
shift3 = mean(p_man(1801,:)) - mean(p_raw3(1801,:));
shift4 = mean(p_man(1801,:)) - mean(p_raw4(1801,:));

pScaled1 = p_raw1 + shift1; % [bar]
pScaled2 = p_raw2 +shift2;
pScaled3 = p_raw3 +shift3;
pScaled4 = p_raw4 +shift4;

figure
plot(deltaCrank, p_man);hold on
plot(deltaCrank, p_raw1); hold on
plot(deltaCrank, pScaled1); grid minor
xlabel('\theta [ CA ]')
ylabel('Pressure [bar]')
xlim([0 720])
lgd = legend('Intake manifold pressure','Raw data', 'Raw scaled
   data');
title('Cylinder 1 raw pressure data')
set(gca,'fontsize', 20)
```

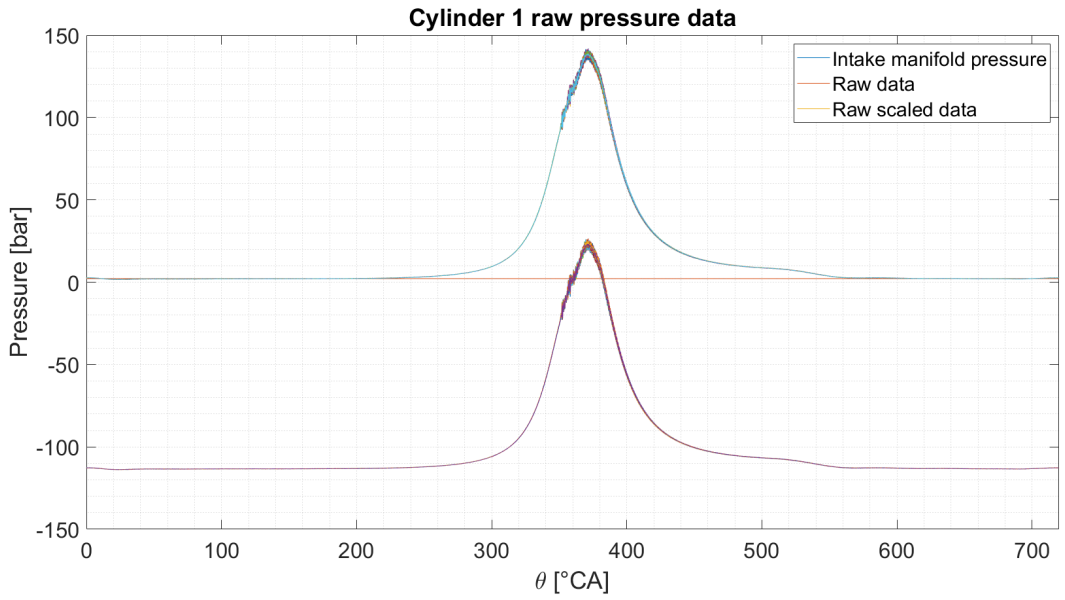

Figure 14: Cylinder 1 raw pressure data

The scaled data need to be averaged:

```
for i=1:length(deltaCrank)
    pCyl1(i,1) = mean(pScaled1(i,:));    % [bar]
    pCyl2(i,1) = mean(pScaled2(i,:));
    pCyl3(i,1) = mean(pScaled3(i,:));
    pCyl4(i,1) = mean(pScaled4(i,:));
end
figure
plot(deltaCrank, p_man);hold on
plot(deltaCrank, pCyl1); grid minor
xlabel('\theta [ CA ]')
ylabel('Pressure [bar]')
xlim([0 720])
title('Cylinder 1 averaged pressure data')
set(gca,'fontsize', 20)
```

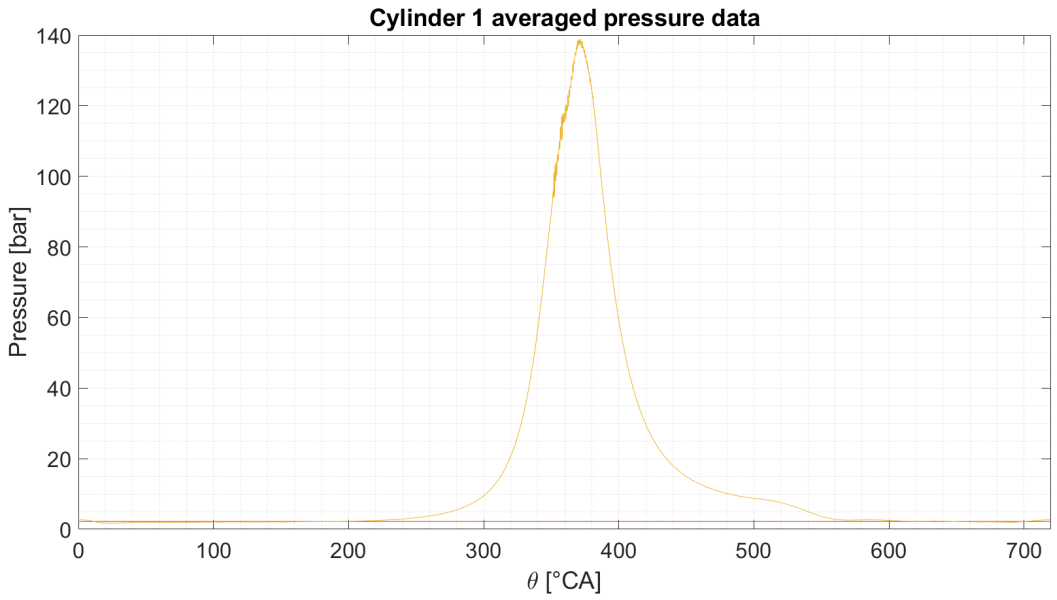

Figure 15: Cylinder 1 averaged pressure data

Once the in-cylinder pressure data are scaled and averaged, it's possible to go on filtering them. Three different methodologies are presented:

1. moving-average filter;
2. generic digital filter;
3. zero-phase digital filter.

1st method:

```
k=10;   % The higher the value the smoother the resulting curve
pCyl1MAfilter = movmean(pCyl1, k);  % [bar]
```

2nd method:

```
windowSize = 10;
b = (1/windowSize)* ones(1,windowSize);    a = 1;
pCyl1DFfilter = filter(b,a,pCyl1);  % [bar]
```

3rd method (Butterworth filter):

```
engSpeed = mean(double(ifile.SPEED.data)); % [rpm]
CAstep = deltaCrank(2)-deltaCrank(1);   % [ CA ]
```

```
fs = (engSpeed/60) *(360/CAstep);   % [Hz]
fc = 4000;  % [Hz] cutoff frequency
Wn = fc/(fs/2); % Ratio between cutoff and Nyquist frequency
n=2;    % order of the filter
[b,a] = butter(n,Wn);
pCyl1BUTfilter = filtfilt(b,a,pCyl1);   % [bar]

figure
plot(deltaCrank, pCyl1); hold on
plot(deltaCrank, pCyl1MAfilter); hold on
plot(deltaCrank, pCyl1DFfilter); hold on
plot(deltaCrank, pCyl1BUTfilter); grid minor
xlabel('\theta [ CA ]')
ylabel('Pressure [bar]')
xlim([366 377])
ylim([133 140])
legend('Oringal averaged data','Moving-average filter','Generic
   digital filter','Zero-phase digital filter');
title('Comparison of the different filters usage')
set(gca,'fontsize', 20)
```

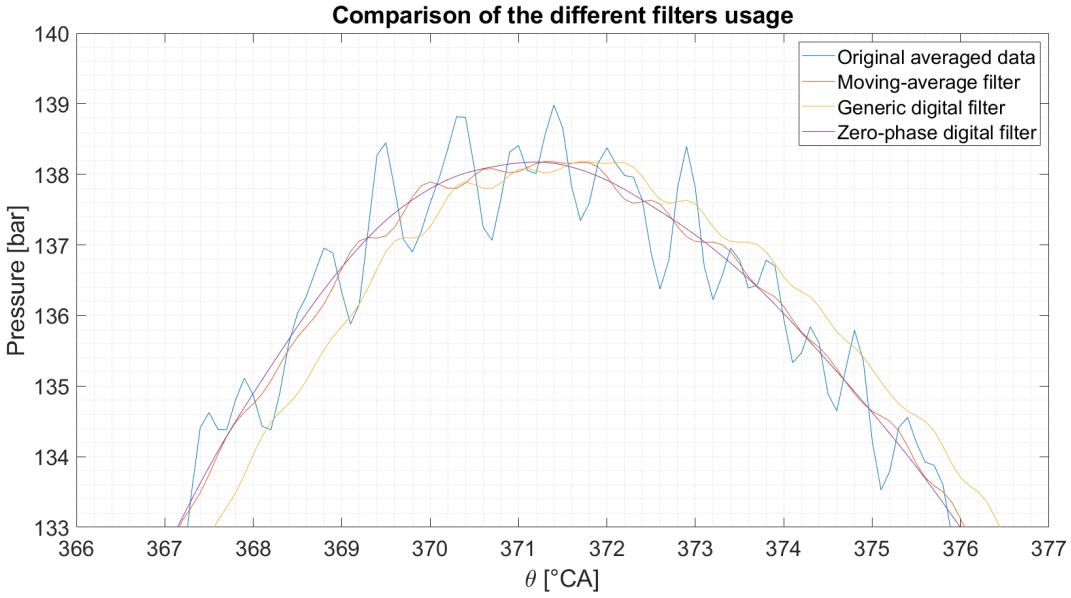

Figure 16: Comparison of the different filters usage

The Butterworth filter shows the highest results resolution, thus it is used for the filtering of all the other pressures:

```
pCyl2BUTfilter = filtfilt(b,a,pCyl2);   % [bar]
pCyl3BUTfilter = filtfilt(b,a,pCyl3);
pCyl4BUTfilter = filtfilt(b,a,pCyl4);
```

Finally an average pressure is retrieved starting from the 4 cylinders filtered pressures:

```
for i=1:length(deltaCrank)
    pCylBUTfilter = [pCyl1BUTfilter(i,1), pCyl3BUTfilter(i,1),
       pCyl3BUTfilter(i,1), pCyl4BUTfilter(i,1)];
    pCyl(i,1) = mean(pCylBUTfilter, 'all'); % [bar]
end

figure;
plot(deltaCrank, pCyl1BUTfilter); hold on
plot(deltaCrank, pCyl); grid minor
xlabel('\theta [ CA ]')
ylabel('Pressure [bar]')
xlim([366 377])
ylim([133 140])
legend('Cylinder 1 pressure', 'Average cylinder pressure');
title('Filtered pressure data averaged for the cylinders')
set(gca,'fontsize', 20)
```

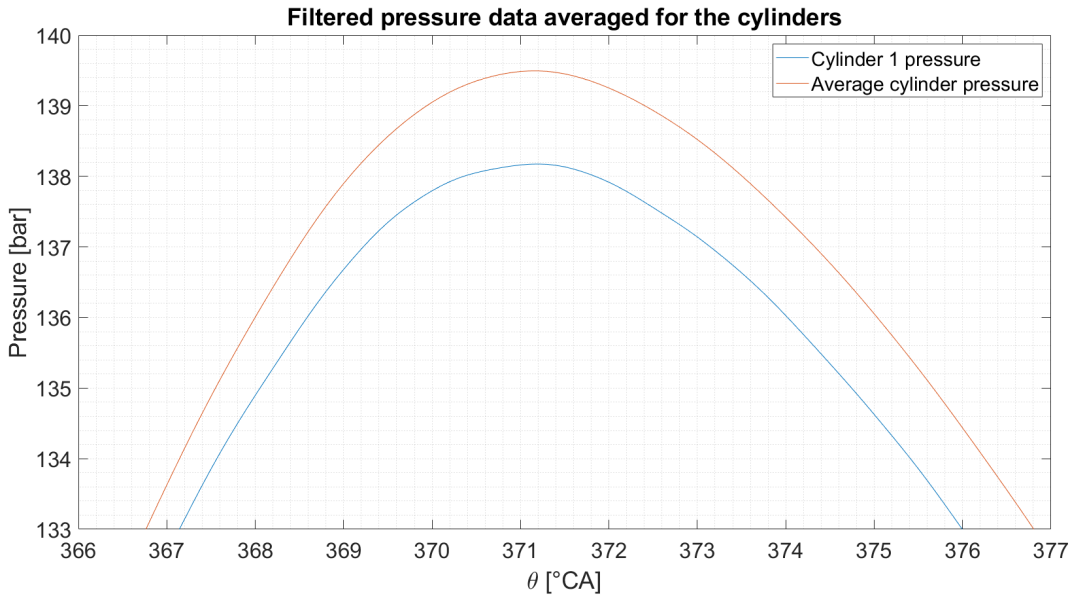

Figure 17: Filtered pressure data averaged for the cylinders

2.4 HRR analysis

The preliminary steps for undertaking this kind of analysis consists in extrapolating the injection current data and evaluating the in-cylinder volume trend over the cycle. Meanwhile, characteristic points of the cycle are extrapolated, such as the start of injection (SOI).

```
injCurrent = double(ifile.INJ1.data);    % [A]
possibleCrankAngles = [];
for i= 1:length(deltaCrank)
    if injCurrent(i,1) > 1e-1
        possibleCrankAngles = [possibleCrankAngles, i]; % [-]
    end
end
SOIindex = min(possibleCrankAngles);     % [-]
SOI = deltaCrank(SOIindex)  % [ CA ]
```

$SOI = 345.4CA$;

```
bore = double(ifile.engine.bore);
stroke = double(ifile.engine.stroke);
r = stroke/2;    % [mm]
l_rod = double(ifile.engine.conrod_length); % [mm]
```

```
rc = double(ifile.engine.compression_ratio);    % [-]
Vd = stroke * bore^2 *pi /4*10^-9; % [m^3]
Vc = Vd/(rc-1); % [m^3]
m = 1.3;    % Polytropic coefficent
cpEGR = HRR_data_add(6);    gammaEGR = 1.4;
R_air = 287.05;     R_egr = cpEGR * (1-1/gammaEGR); % [J/(kg*K)]
qm_air = HRR_data_add(3);   qm_egr = HRR_data_add(5);   % [kg/h]
m_air = qm_air / (engSpeed *60*2);  m_egr = qm_egr / (engSpeed
   *60*2);  % [kg/s]
Rmix = m_air * R_air + m_egr * R_egr;    % [J/(kg*K)]

for i = 1 : length(deltaCrank)
    deltaCrankRad(i) = deg2rad(deltaCrank(i));  % [rad]
    beta(i) = asin(r/l_rod * sin(deltaCrankRad(i)));    % [rad]
    x(i) = r*((1-cos(deltaCrankRad(i)))+ l_rod/r *(1-cos(beta(i))));
          % [mm]
    Vx(i) =  Vc + (x(i) * pi/4 *bore^2)*10^-9;  % [m^3]
end
figure
plot(Vx*10^3,pCyl); grid minor
xlabel('Volume [L]')
ylabel('Pressure [bar]')
title('Cycle representation')
set(gca,'fontsize', 20)
```

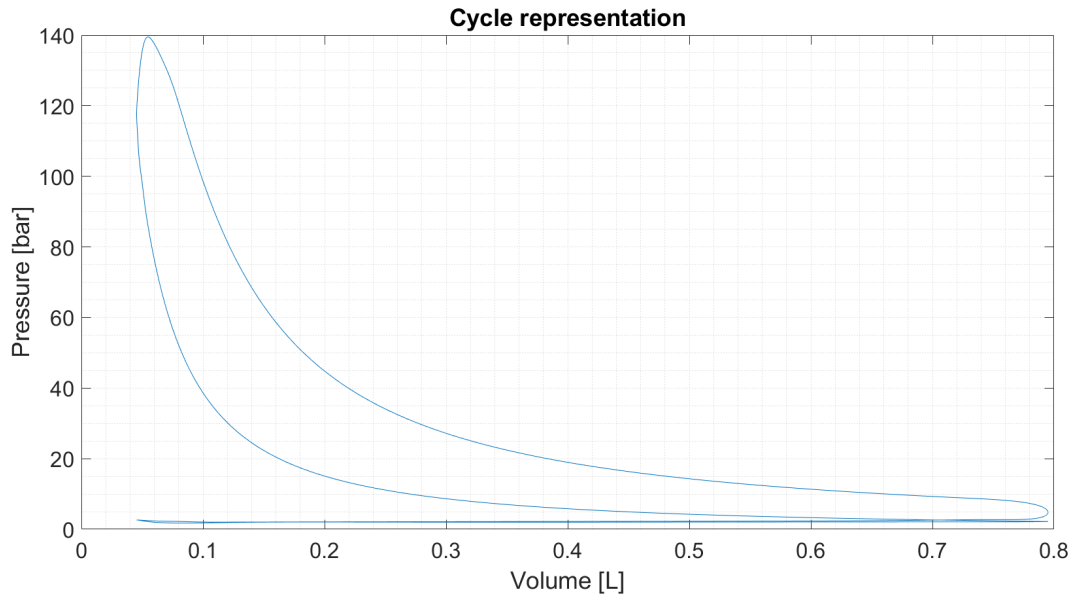

Figure 18: Cycle representation

All these computed values are the bases for the calculation of the differential value of the net heat release rate:

```
for i=1:length(deltaCrank)
    if i == 1
        dQn = 0;
    else
        dV = Vx(i)-Vx(i-1); % [m^3]
        dp = (pCyl(i)-pCyl(i-1))*10^5;  % [Pa]
        T = pCyl(i)*10^5 * Vx(i) / Rmix;    % [K]
        Tdata(i) = T;   % [K]
        gamma = 1.338-(6e-5)*T+(1e-8)*T^2;  % [-]
        gammaData(i) = gamma;   % [-]
        dQn = (gamma/(gamma-1)) * pCyl(i)*10^5 * dV + (1/(gamma-1))
           * Vx(i) * dp;   % [J/ CA ]
    end
    dQnData(i) = dQn;   % [J/ CA ]
end

figure
plot(deltaCrank, dQnData); grid minor
```

```
xlabel('\theta [ CA ]')
ylabel('HRR_{n} [J/ CA ]')
xlim([300 450])
title('Net heat release rate per cycle')
set(gca,'fontsize', 20)
```

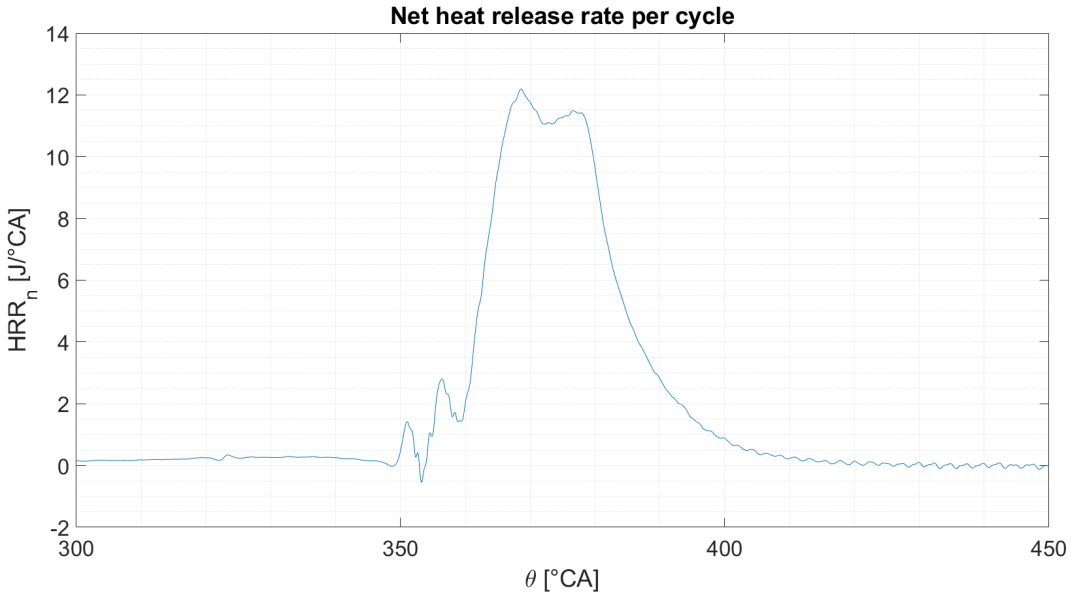

Figure 19: Net heat release rate per cycle

To represent the mass fraction burned during the cycle, the cumulative sum of the infinitesimal net heat release rate is exploited:

```
Qlhv = HRR_data_add(7); % [MJ/(kg*K)]
qm_fuel = HRR_data_add(4);  % [kg/h]
m_fuel = qm_fuel / (engSpeed *60*2);  % [kg]
QnCumulative = cumsum(dQnData(3450:4500));
xb = QnCumulative/max(QnCumulative);
```

To conclude, the remaining cycle characteristic points are retrieved and a summarising plot of the engine combustion process is shown:

```
for i= 1:length(xb)
    if abs(0.02-xb(i)) <= 0.0005
        SOC = deltaCrank(i+3450)    % [ CA ]
    elseif abs(0.1-xb(i)) <= 0.001
```

```
        MFB10 = deltaCrank(i+3450)  % [ CA ]
    elseif abs(0.5-xb(i)) <= 0.002
        MFB50 = deltaCrank(i+3450)  % [ CA ]
    elseif abs(0.9 - xb(i)) <= 0.0005
        MFB90 = deltaCrank(i+3450)  % [ CA ]
    end
end
```

- $SOC = 356.1CA$;
- $MFB_{10} = 363.4CA$;
- $MFB_{50} = 374CA$;
- $MFB_{90} = 388.2CA$.

```
for i = SOIindex:4500
    if i == SOIindex
        pMotored(i-SOIindex+1) = pCyl(SOIindex);    % [bar]
    else
        pMotored(i-SOIindex+1) = pMotored(i-SOIindex) *
           (Vx(i-1)/Vx(i))^gammaData(i);   % [bar]
    end
end

verticalLines = linspace(0,1);
SOIline=ones(1,100)*SOI;  SOCline=ones(1,100)*SOC;
   MFB10line=ones(1,100)*MFB10;    MFB50line=ones(1,100)*MFB50;
   MFB90line=ones(1,100)*MFB90;

figure
yyaxis left
plot(deltaCrank, pCyl); hold on
plot(deltaCrank(SOIindex:4500), pMotored); hold on
plot(deltaCrank, injCurrent(:,1),LineWidth=2); hold on
ylim([0 inf])
ylabel('Pressure [bar]')
yyaxis right
```

```
plot(deltaCrank(3450:4500), xb);hold on
plot(SOIline, verticalLines); hold on
plot(SOCline, verticalLines); hold on
plot(MFB10line, verticalLines); hold on
plot(MFB50line, verticalLines); hold on
plot(MFB90line, verticalLines); grid minor
ylabel('Mass fraction of fuel burned [-]')
xlabel('\theta [ CA ]')
ylim([0 inf])
xlim([320 430])
legend('Filtered in-cylinder pressure [bar]','Motored pressure
    [bar]','Injection current [A]','Mass fraction of fuel burned
    [-]','Start of ignition','Start of
    combustion','MFB10','MFB50','MFB90');
title('Engine combustion process')
set(gca,'fontsize', 20)
```

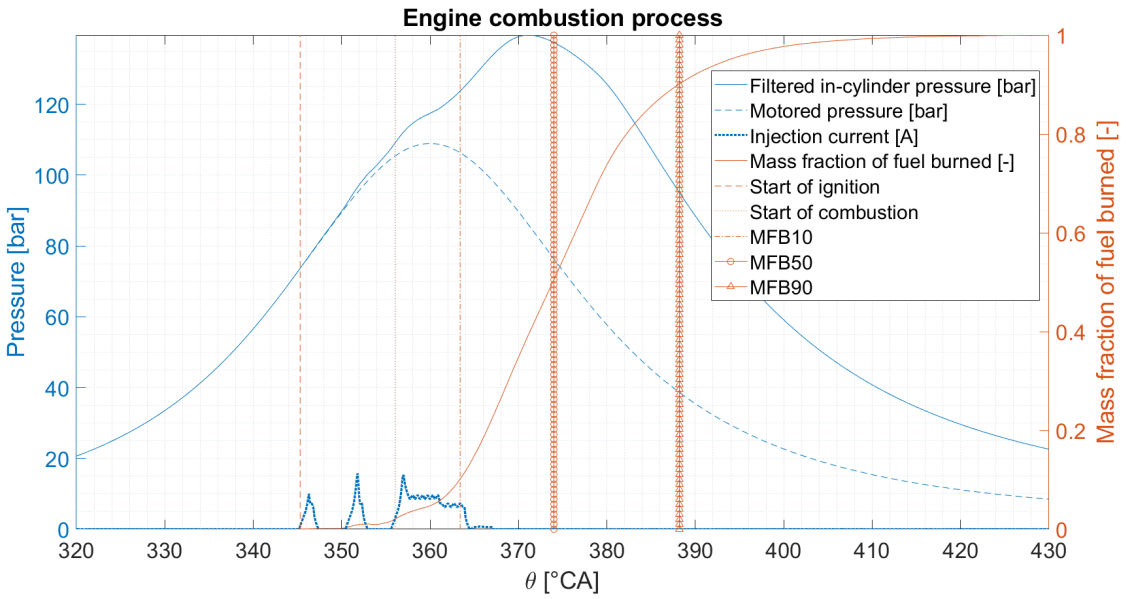

Figure 20: Engine combustion process