

# Contemporary Powertrains

## Combustion and Emission Study: Modelling Fuels and Emissions

Alexander Cook - 2104599

### Task 1 - Comparative Combustion Analysis of 3 Engines

This task processes the supplied engine datasets to generate consistent pressure–crank angle and pressure–volume outputs on a common crank grid. The objective is to reproduce the required diagnostic plots (P– $\theta$ , P–V, indicated work, IMEP, polytropic indices, and 3D surfaces) and confirms the workflow is numerically stable and comparable across engines.

```
clc; clear; close all;

desired_crank = (-360:1:359)';
P_units = 'bar';
V_units = 'cm^3';

FordFOX_P_mat = 'FordFOX_Data.mat';
FordFOX_V_mat = 'FordFOX_VData.mat';
CLAAS_P_mat = 'CLAAS_Data.mat';
McLaren_P_mat = 'McLaren_Data.mat';

% ENGINE DATA

% CLAAS geometry
CLAAS.bore_mm = 104;
CLAAS.stroke_mm = 132;
CLAAS.rod_mm = 195;
CLAAS.CR = 17.5;
CLAAS.Ncyl = 6;

% McLaren geometry
McL.stroke_mm = 69.9;
McL.rod_mm = 110;
McL.CR = 10.5;
McL.Ncyl = 8;
McL.disp_cc = 3799;

% FordFOX (measured pressure + measured volume)
FordFOX = build_FordFOX(FordFOX_P_mat, FordFOX_V_mat, desired_crank);

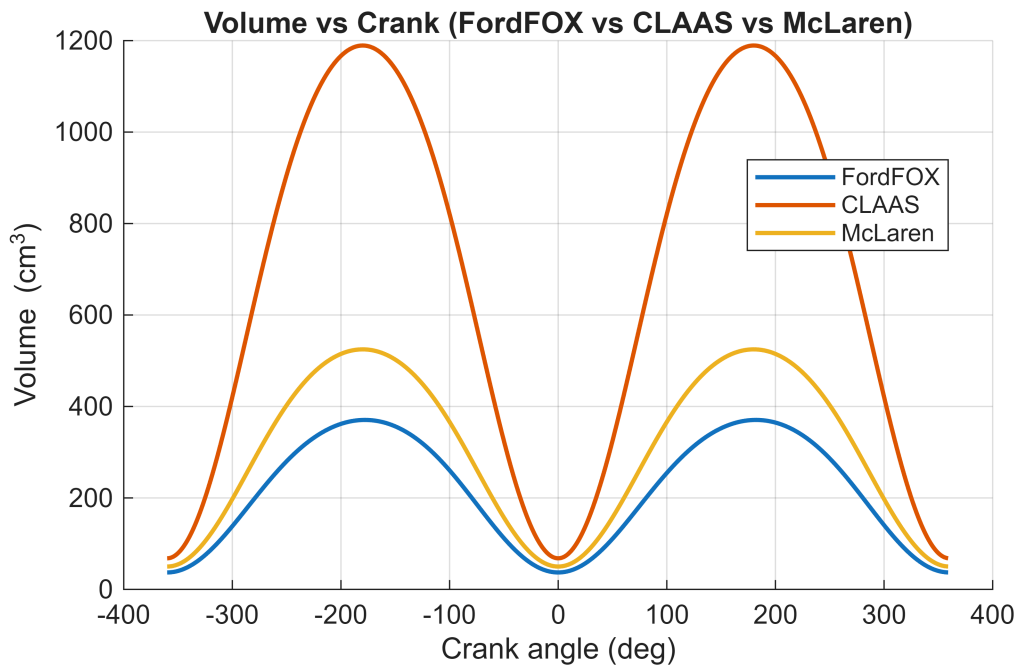
% CLAAS (measured pressure + geometry-based volume)
CLAASeng = build_geometry_engine('CLAAS', CLAAS_P_mat, ...
    CLAAS.bore_mm, CLAAS.stroke_mm, CLAAS.rod_mm, CLAAS.CR, desired_crank);

% McLaren (measured pressure + geometry-based volume)
McLaren = build_geometry_engine_mclaren('McLaren', McLaren_P_mat, ...
```

```
McL.stroke_mm, McL.rod_mm, McL.CR, McL.Ncyl, McL.disp_cc, desired_crank);
```

```
% VOLUME vs CRANK (ALL 3 ENGINES)
```

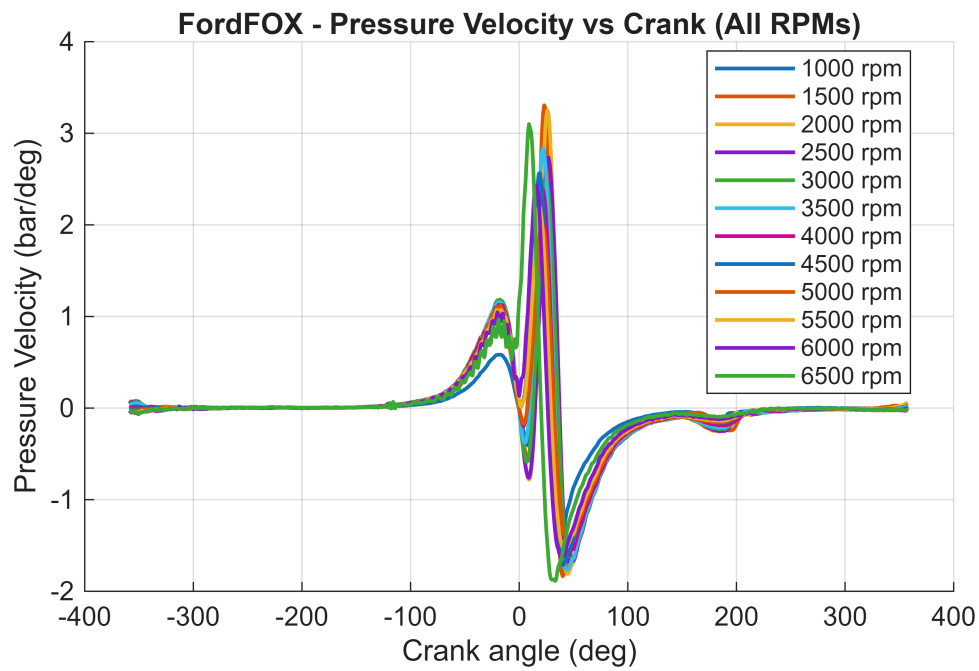
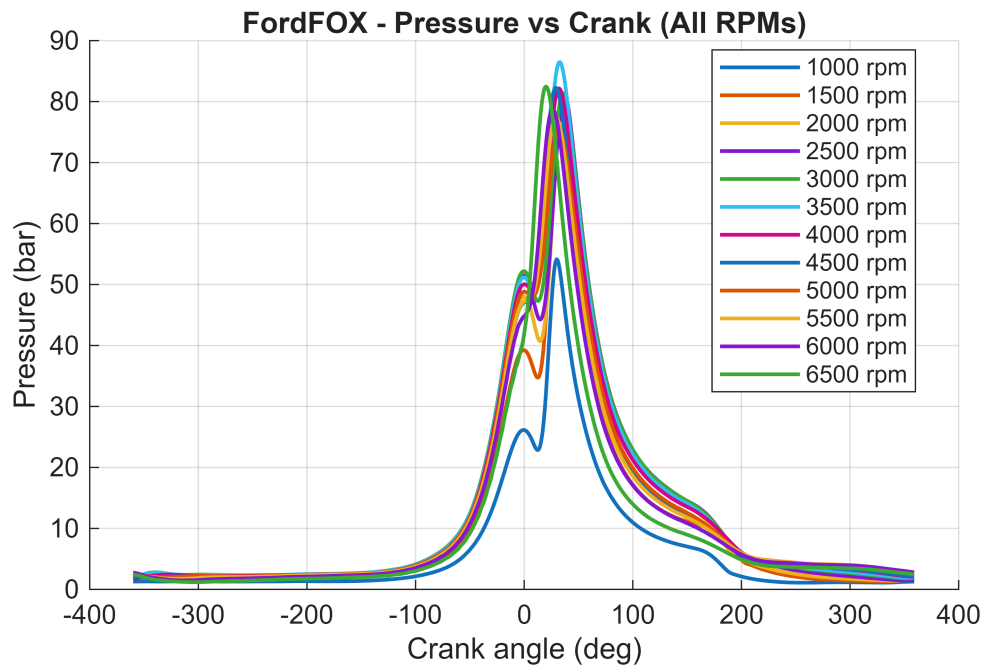
```
figure('Name','Volume vs Crank (All Engines)');
hold on; grid on;
plot(desired_crank, FordFOX.V_i, 'LineWidth', 1.6);
plot(desired_crank, CLAASeng.V_i, 'LineWidth', 1.6);
plot(desired_crank, McLaren.V_i, 'LineWidth', 1.6);
xlabel('Crank angle (deg)');
ylabel(['Volume (', V_units, ')']);
title('Volume vs Crank (FordFOX vs CLAAS vs McLaren)');
legend({'FordFOX', 'CLAAS', 'McLaren'}, 'Location', 'best');
```

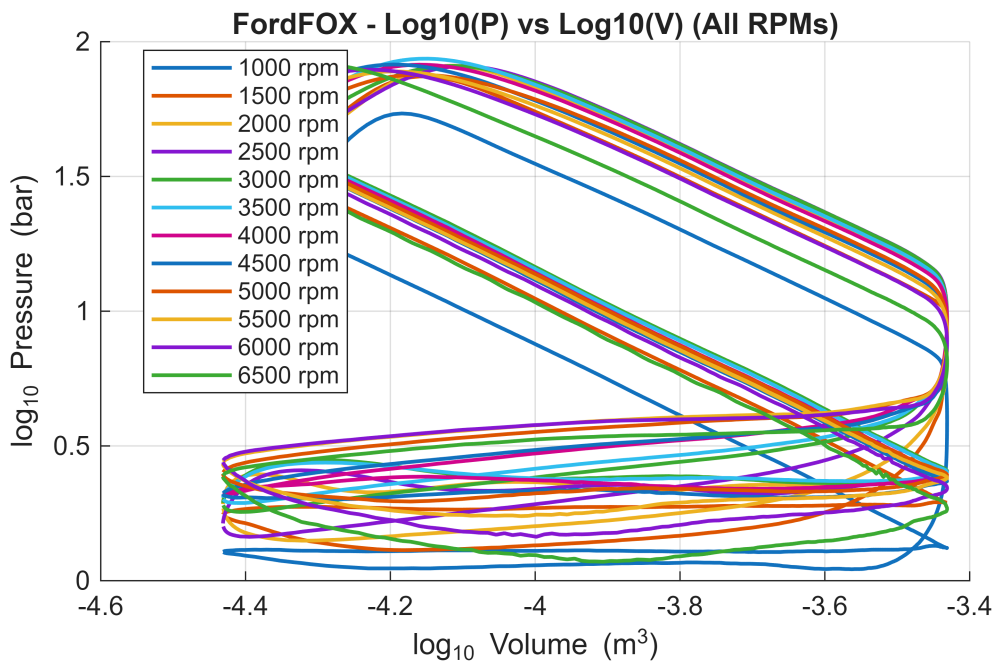
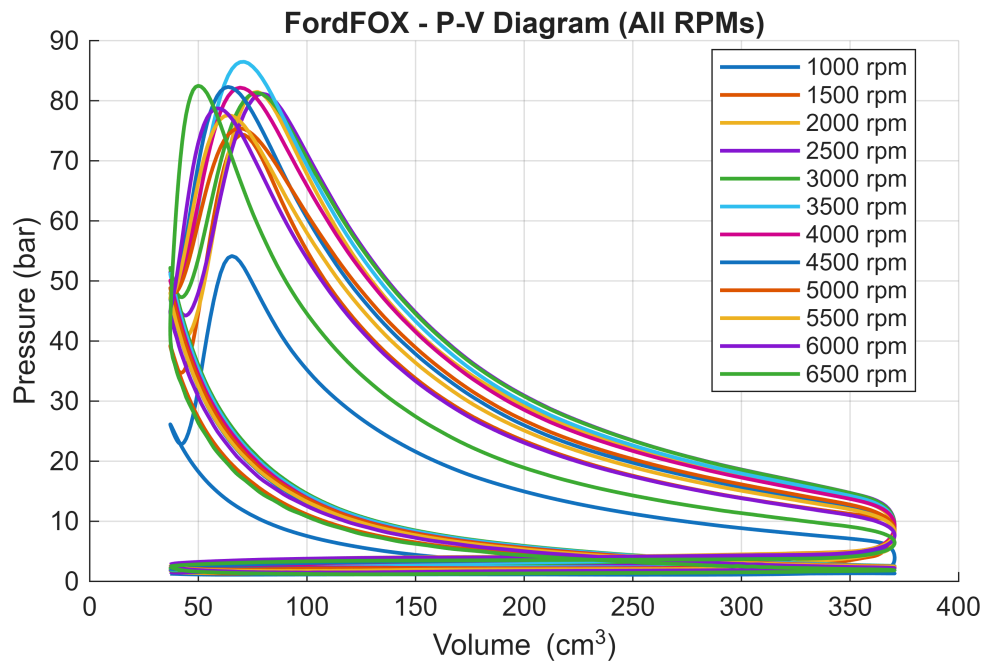


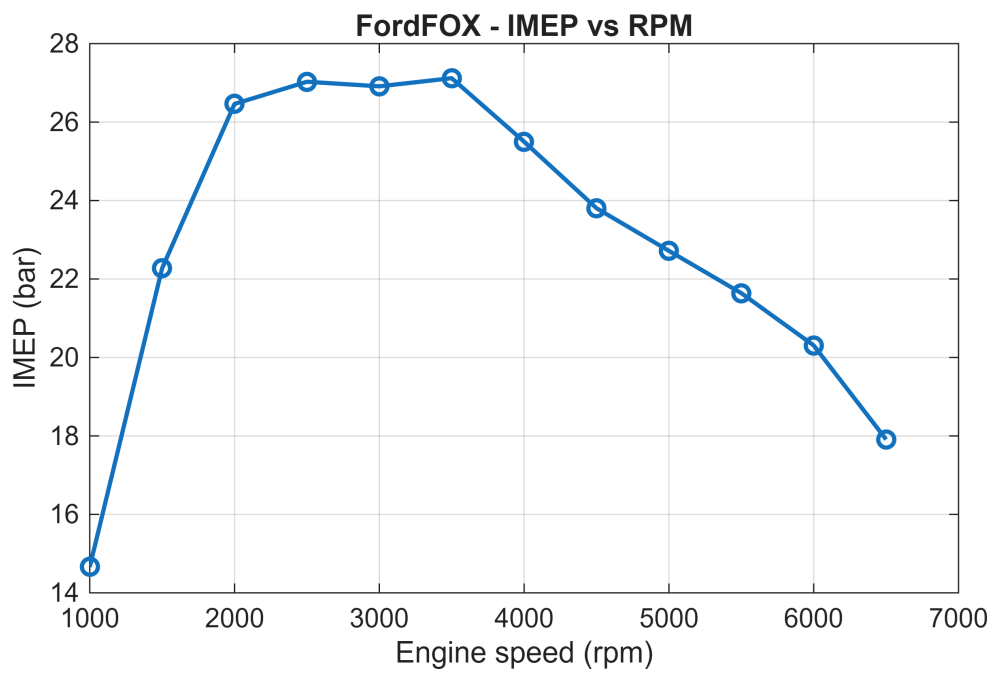
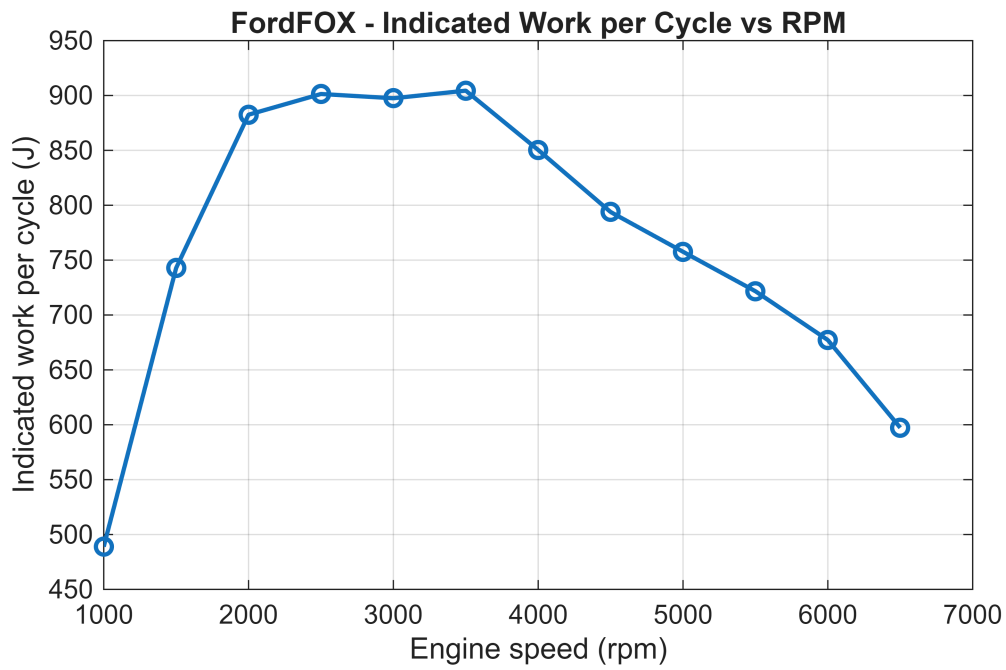
### Cylinder volume versus crank angle

The volume–crank angle comparison highlights the geometric differences between the three engines. The CLAAS engine shows the largest swept and clearance volumes, while the FordFOX has the smallest overall displacement, with the McLaren engine lying between the two. All three profiles are symmetric about TDC and BDC, confirming correct slider–crank volume implementation and crank phasing alignment across datasets. This validates that subsequent pressure–volume and work calculations are performed on a consistent geometric basis.

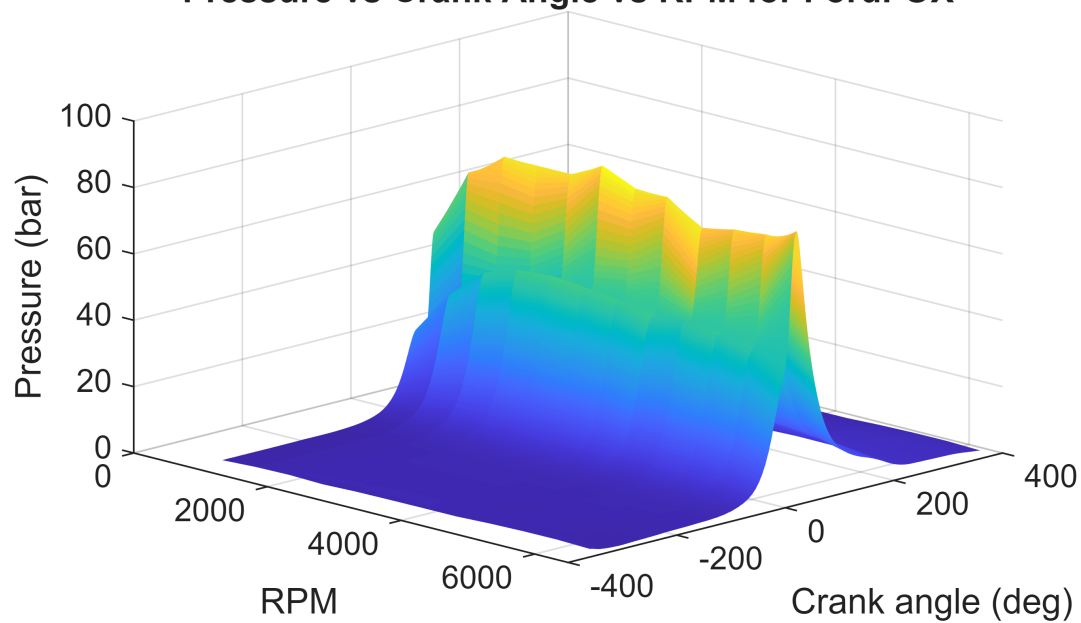
```
plot_engine_suite(FordFOX, 'FordFOX', desired_crank, P_units, V_units);
```



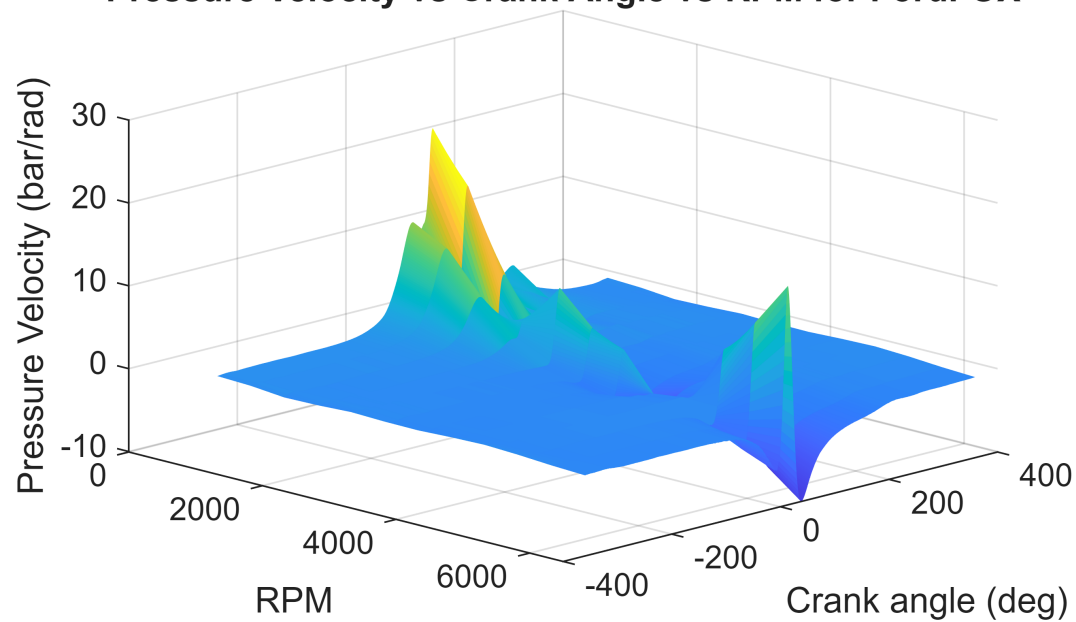


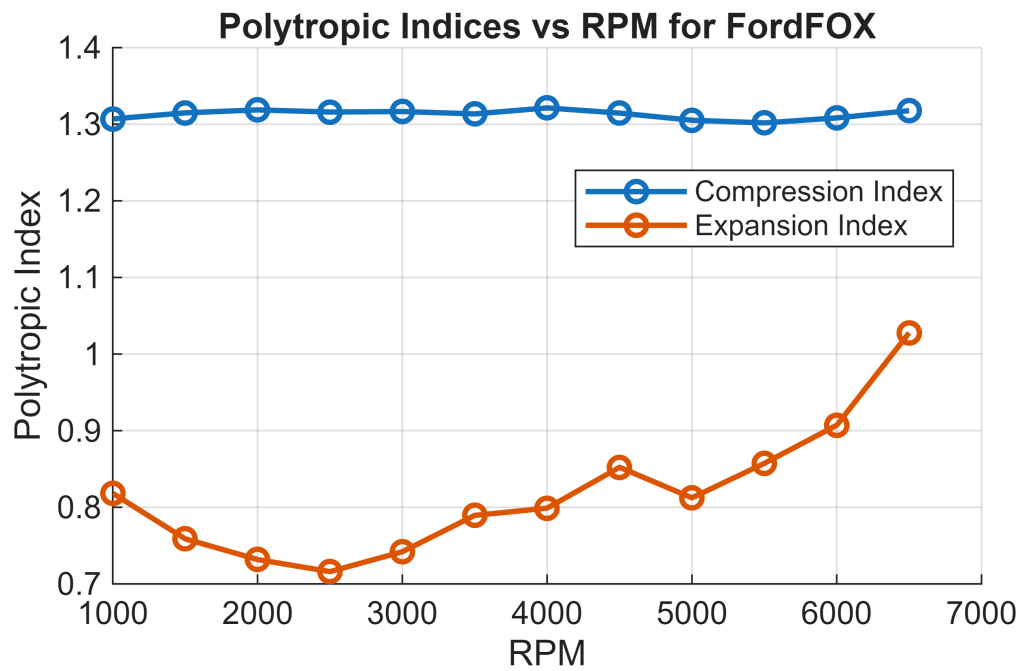


**Pressure vs Crank Angle vs RPM for FordFOX**

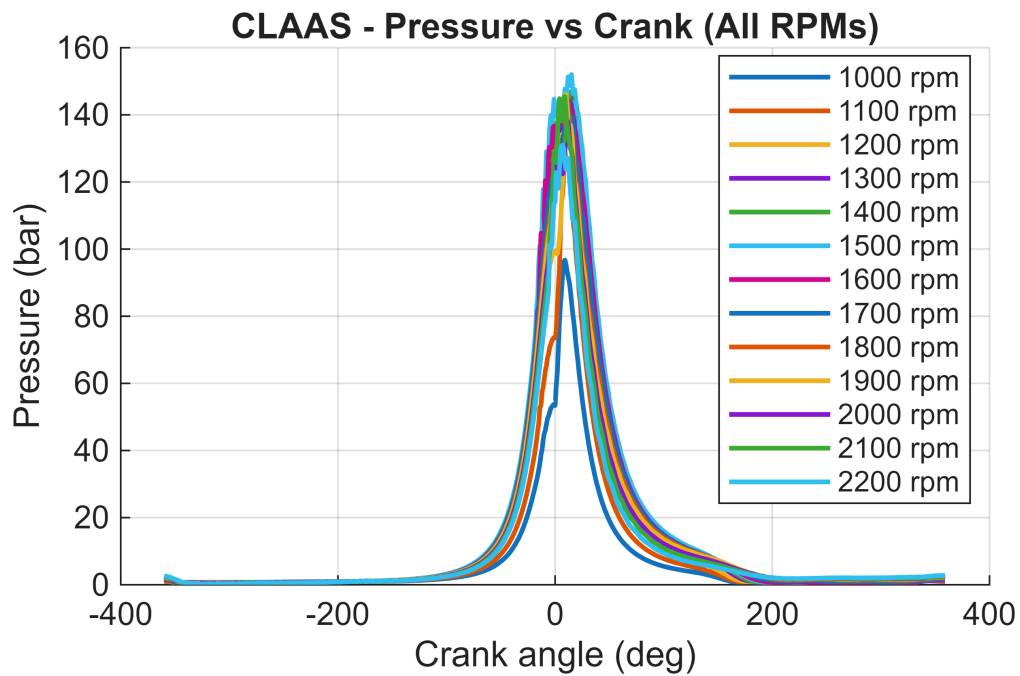


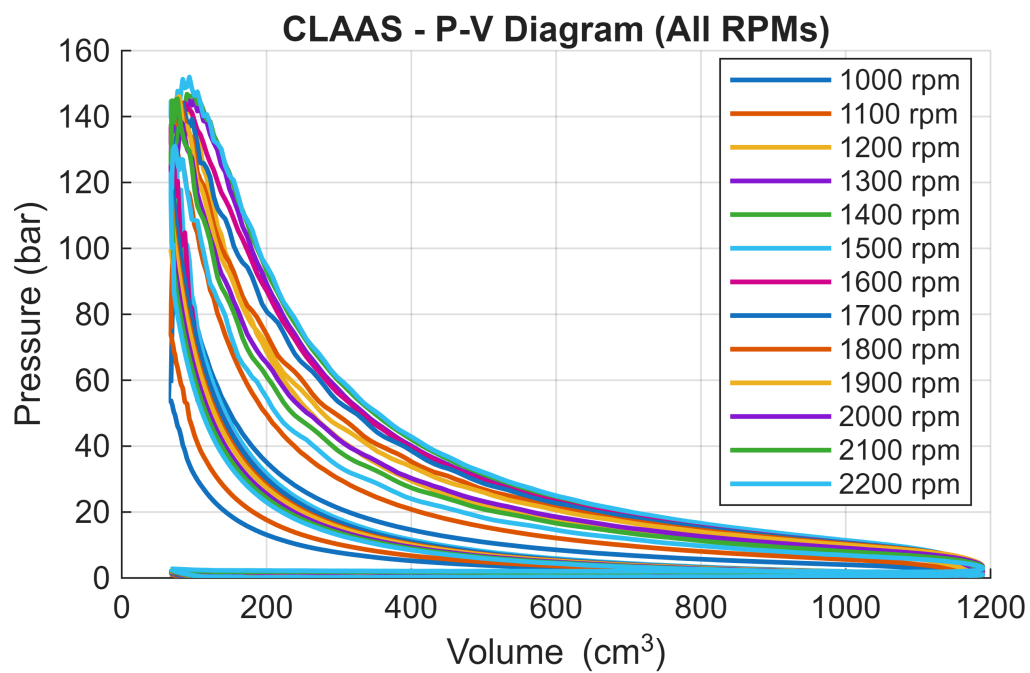
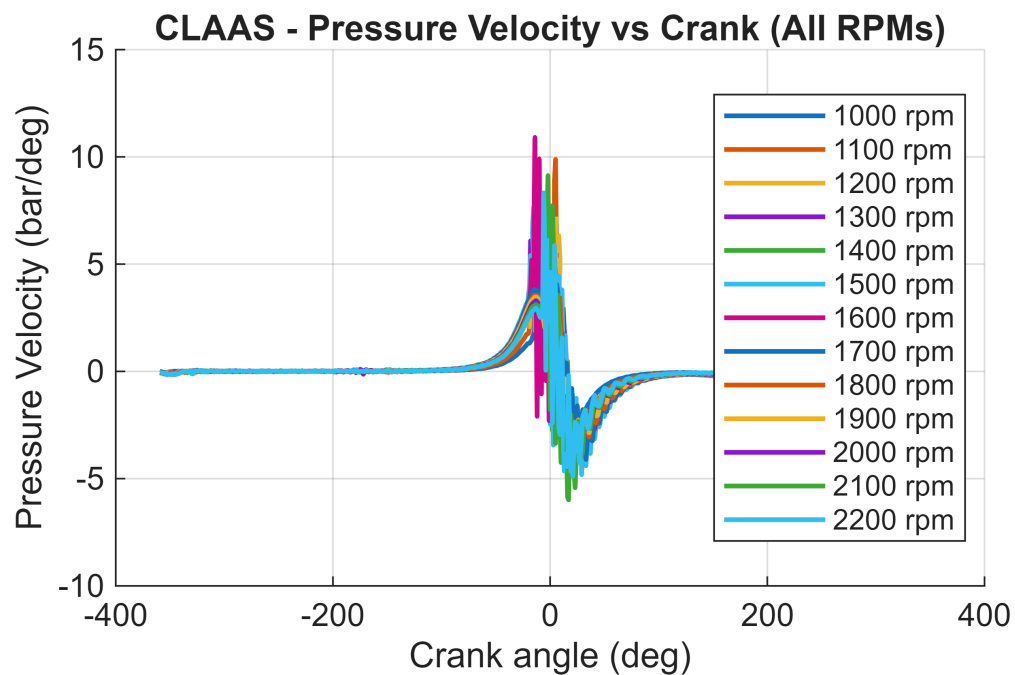
**Pressure Velocity vs Crank Angle vs RPM for FordFOX**

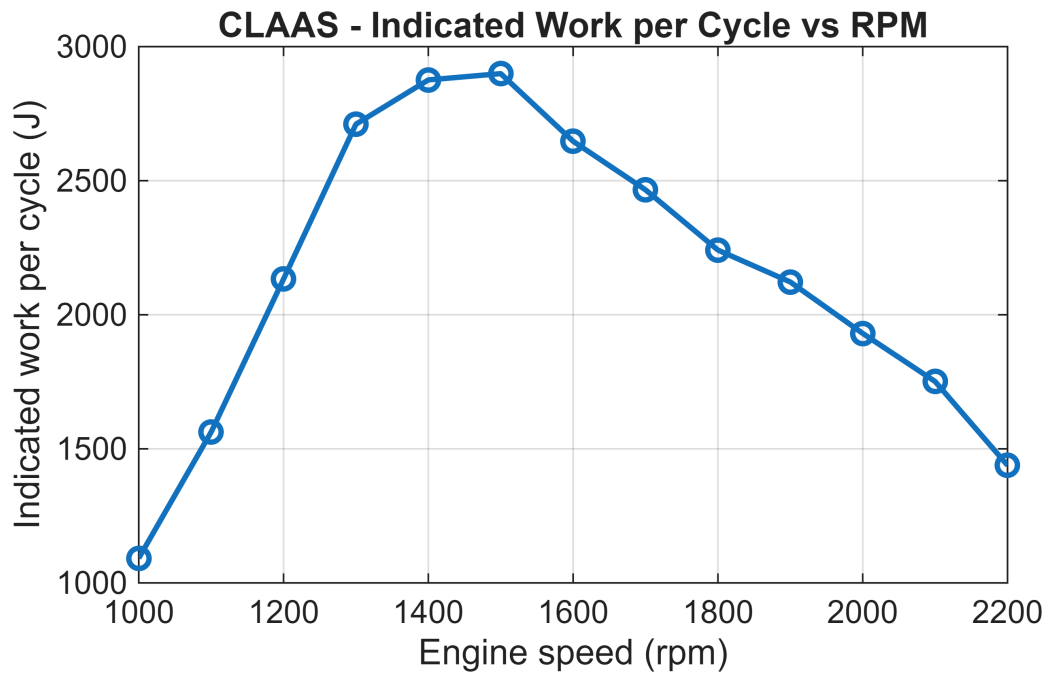
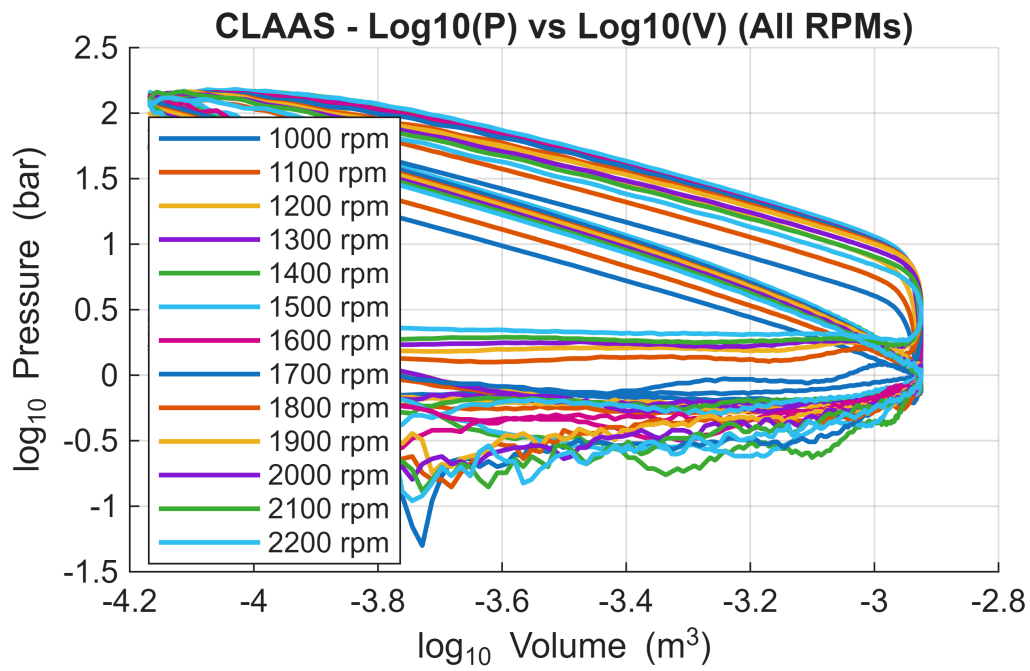


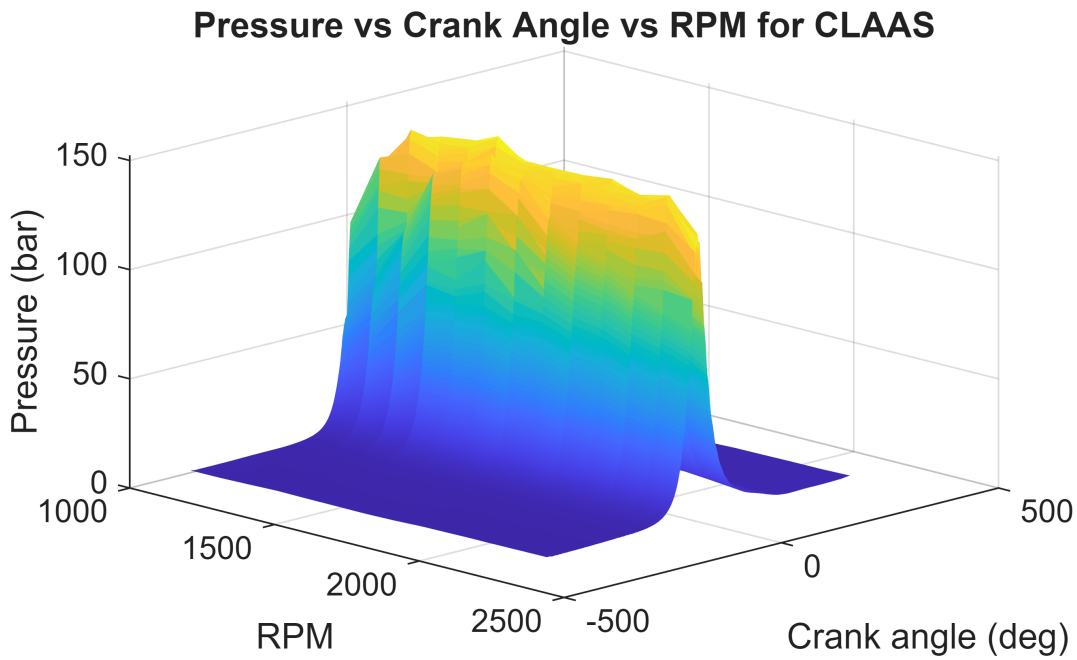
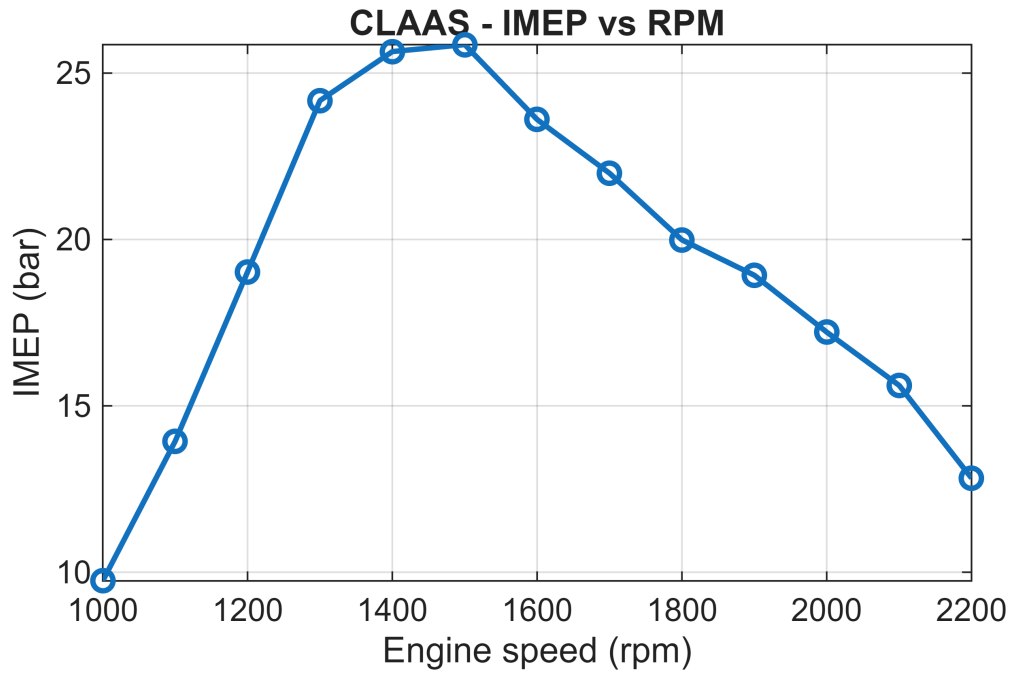


```
plot_engine_suite(CLAASeng, 'CLAAS', desired_crank, P_units, V_units);
```

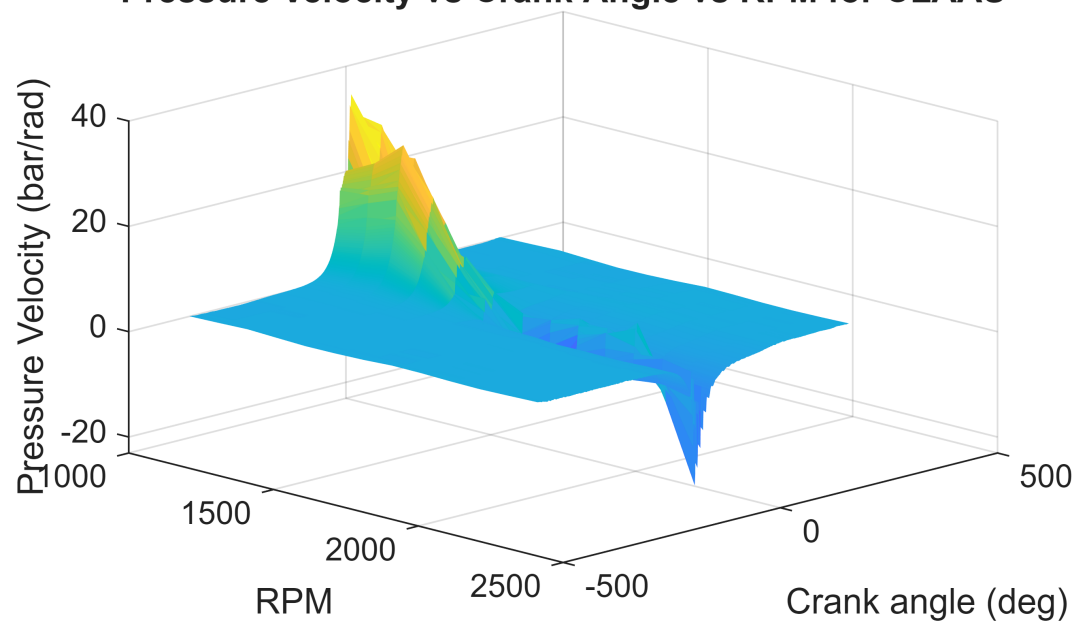




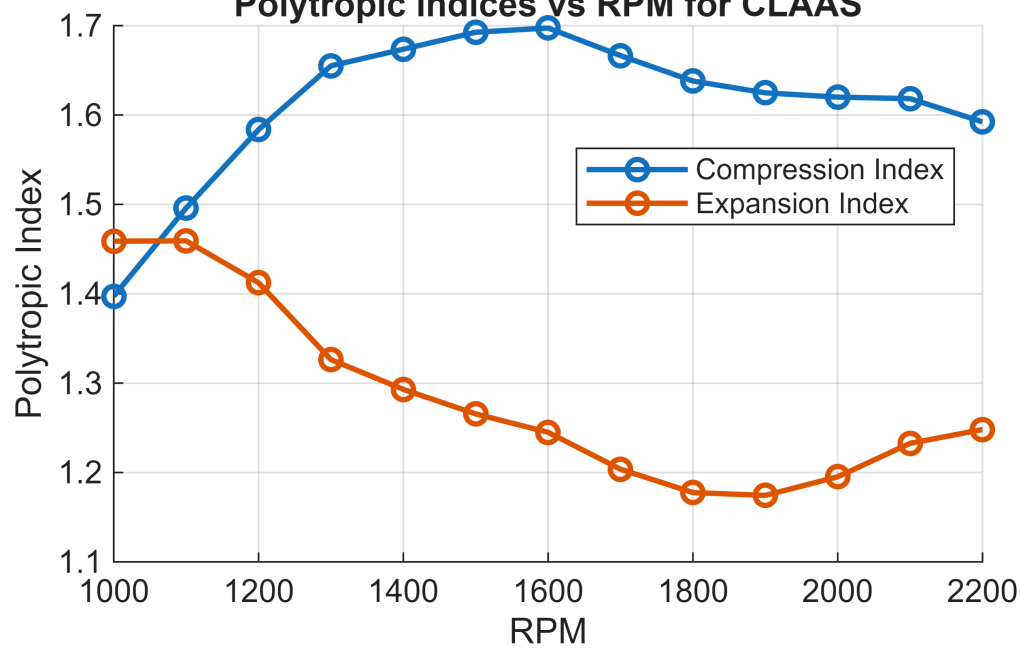




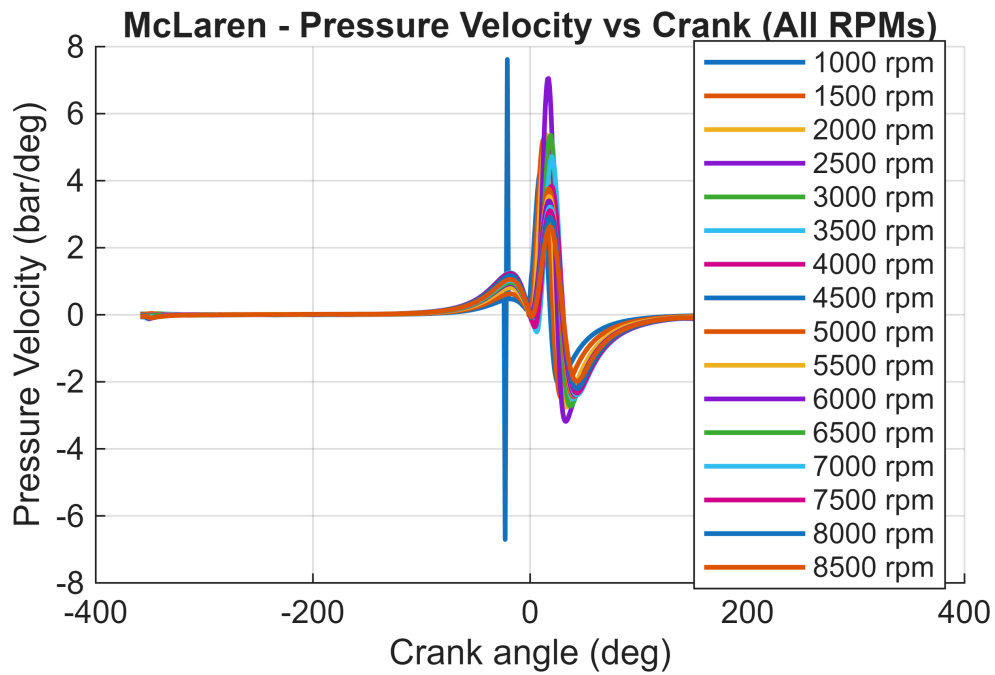
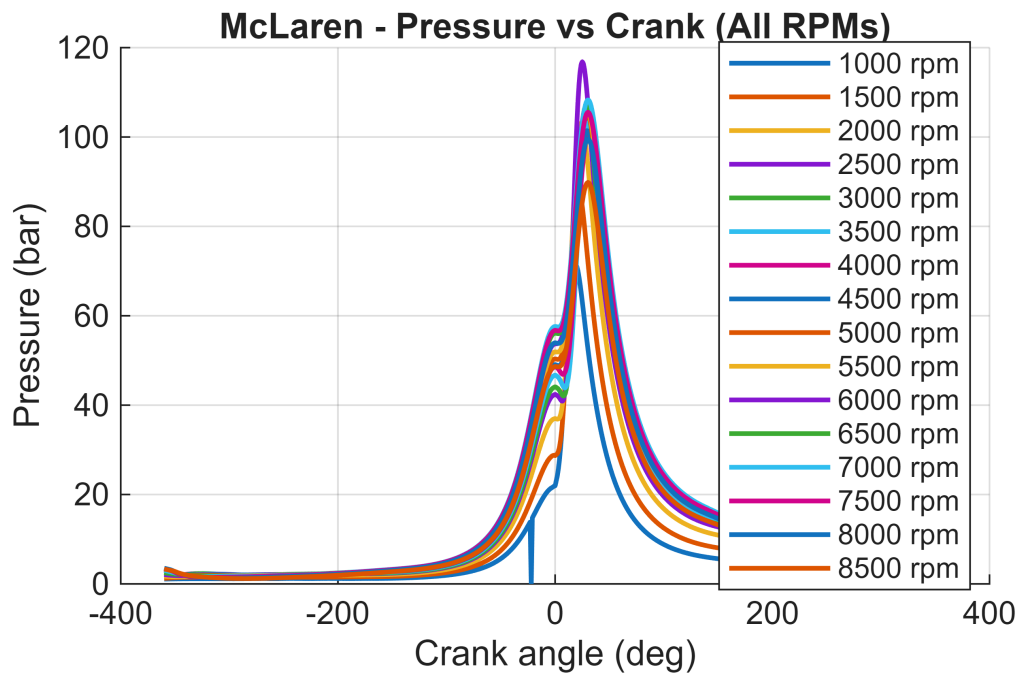
### Pressure Velocity vs Crank Angle vs RPM for CLAAS

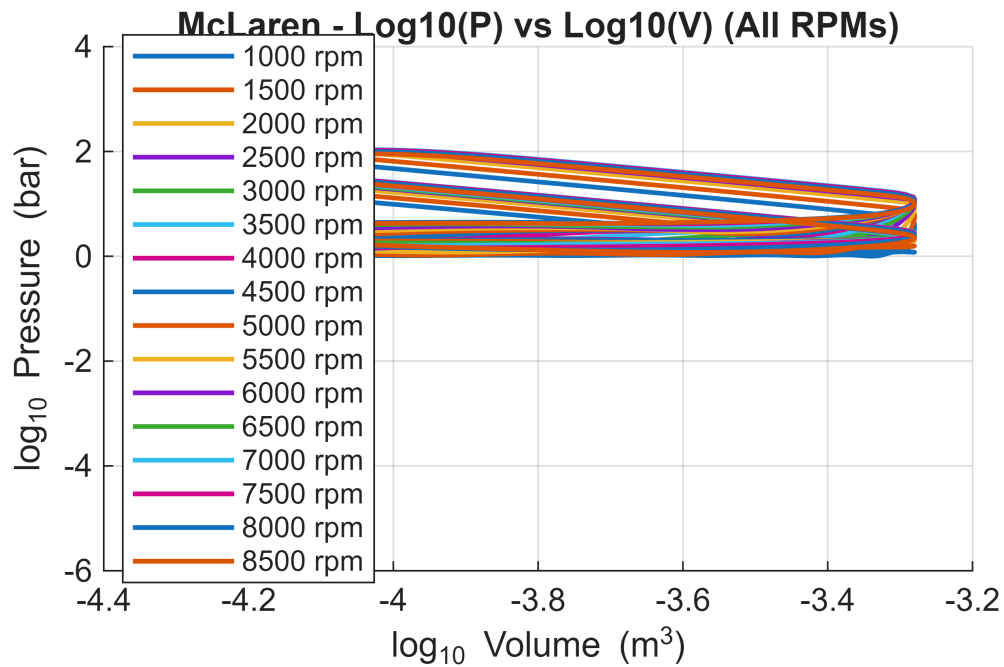
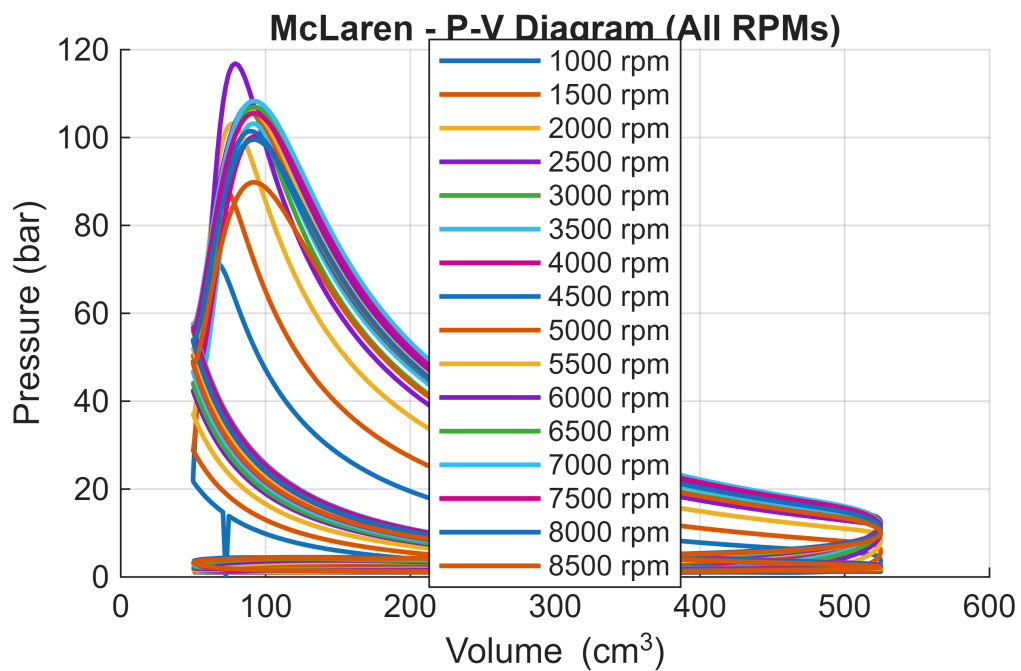


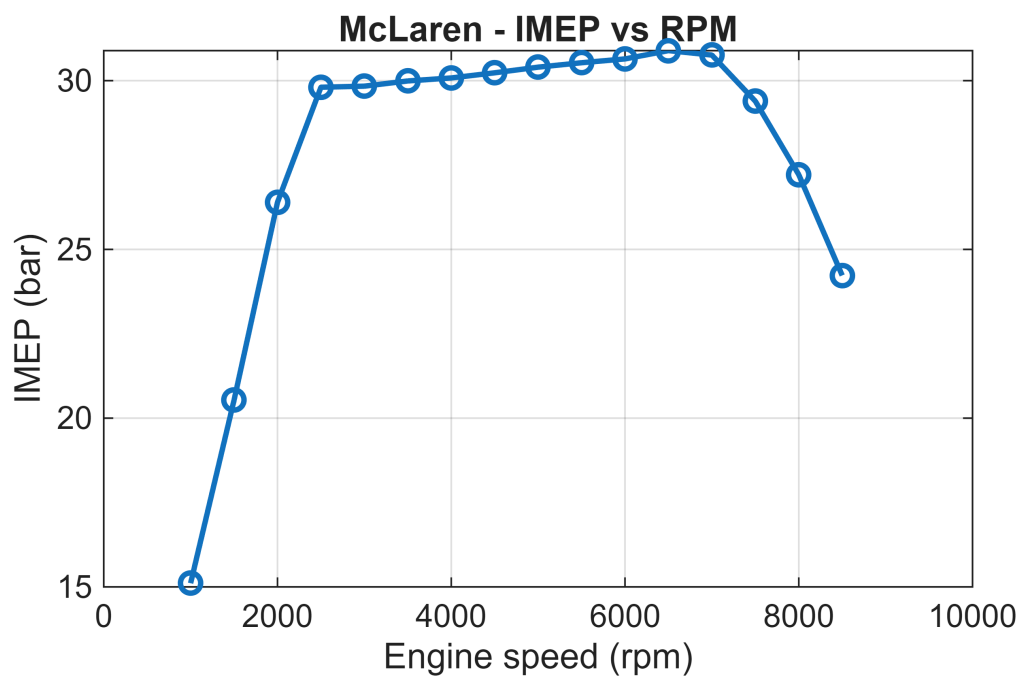
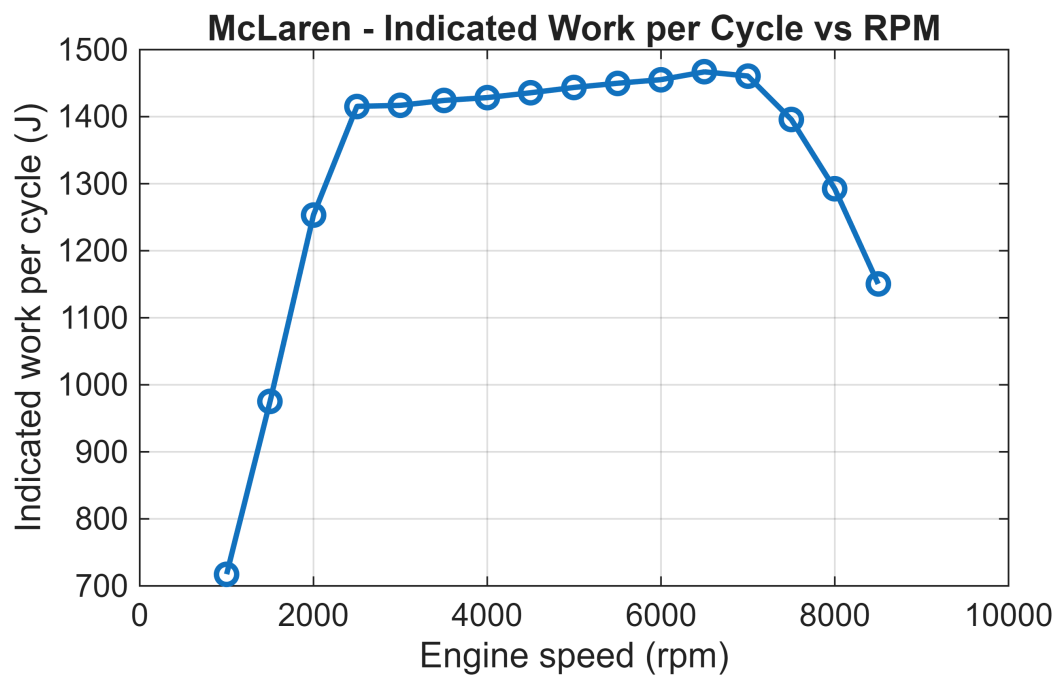
### Polytropic Indices vs RPM for CLAAS



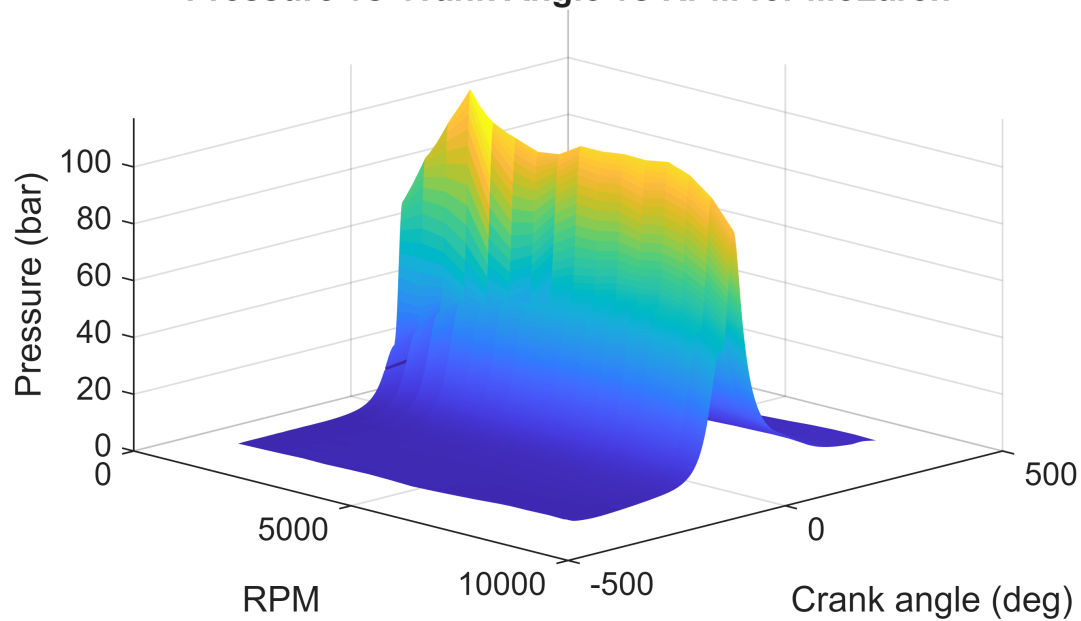
```
plot_engine_suite(McLaren, 'McLaren', desired_crank, P_units, V_units);
```



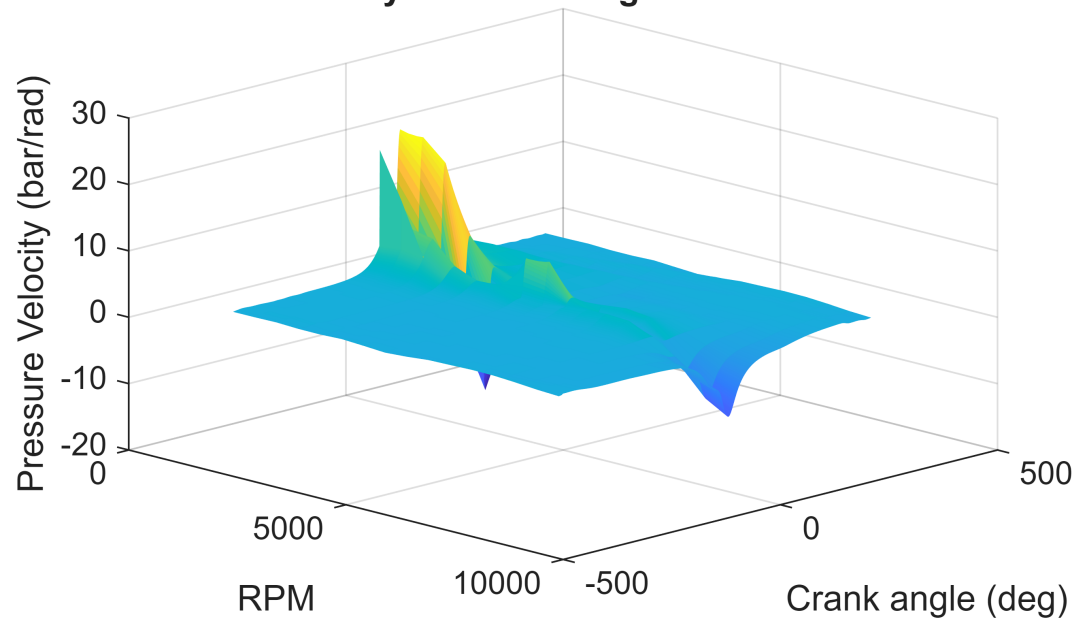


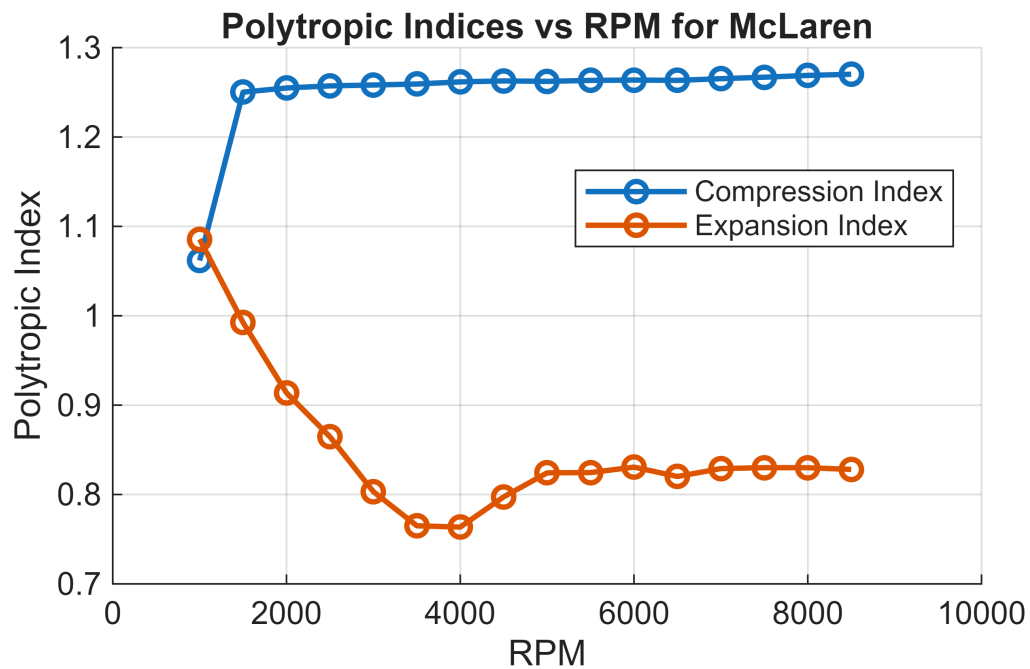


**Pressure vs Crank Angle vs RPM for McLaren**



**Pressure Velocity vs Crank Angle vs RPM for McLaren**





```
function eng = build_FordFOX(P_matfile, V_matfile, desired_crank)

    eng.name = 'FordFOX';

    S = load(P_matfile);
    fn = fieldnames(S);

    Tpress = S.(fn{2});
    A = toNumericArray(Tpress);

    rpm    = A(1, 2:end);
    crank  = A(2:end, 1);
    P      = A(2:end, 2:end);

    okP = isfinite(crank);
    crank = crank(okP);
    P      = P(okP,:);

    [crank, idxP] = sort(crank);
    P = P(idxP,:);

    S = load(V_matfile);
    fn = fieldnames(S);

    Tvol = S.(fn{3});
    Vtab = toNumericArray(Tvol);

    crank_V = Vtab(:,1);
    V        = Vtab(:,2);
```

```

okV = isfinite(crank_V) & isfinite(V);
crank_V = crank_V(okV);
V      = V(okV);

[crank_V, idxV] = sort(crank_V);
V = V(idxV);

eng.rpm = rpm;
eng.P_i = P;
eng.V_i = V(:);

if size(eng.P_i,1) ~= numel(desired_crank)
    warning('FordFOX: pressure rows (%d) do not match desired_crank (%d).',
size(eng.P_i,1), numel(desired_crank));
end
if numel(eng.V_i) ~= numel(desired_crank)
    warning('FordFOX: volume length (%d) does not match desired_crank (%d).',
numel(eng.V_i), numel(desired_crank));
end

eng = add_metrics(eng);
eng = add_polytropic_indices(eng, desired_crank);
end

function eng = build_geometry_engine(name, P_matfile, bore_mm, stroke_mm, rod_mm,
CR, desired_crank)

eng.name = name;

S = load(P_matfile);
fn = fieldnames(S);
Tpress = [];
for k = 1:numel(fn)
    c = S.(fn{k});
    if istable(c) || isnumeric(c)
        Tpress = c; break;
    end
end
if isempty(Tpress)
    error('No suitable table/numeric array found in %s.', P_matfile);
end

A = toNumericArray(Tpress);
rpm  = A(1, 2:end);
crank = A(2:end, 1);
P     = A(2:end, 2:end);

okP = isfinite(crank);
crank = crank(okP);

```

```

P      = P(okP,:);

[crank, idxP] = sort(crank);
P = P(idxP,:);

[crank, ia] = unique(crank, 'stable');
P = P(ia,:);

P_i = interp1(crank, P, desired_crank, 'linear', 'extrap');

bore_cm    = bore_mm/10;
stroke_cm  = stroke_mm/10;
rod_cm     = rod_mm/10;

r  = stroke_cm/2;
l  = rod_cm;
Ap = (pi/4)*bore_cm^2;

Vd = Ap*stroke_cm;           % swept volume (cm^3) per cylinder
Vc = Vd/(CR - 1);           % clearance volume (cm^3)

theta = deg2rad(desired_crank);
x = r*(1 - cos(theta)) + l - sqrt(l^2 - (r*sin(theta)).^2);
V_i = Vc + Ap.*x;

eng.rpm = rpm;
eng.P_i = P_i;
eng.V_i = V_i(:);

eng = add_metrics(eng);
eng = add_polytropic_indices(eng, desired_crank);
end

function eng = build_geometry_engine_mclaren(name, P_matfile, stroke_mm, rod_mm,
CR, Ncyl, disp_cc, desired_crank)

eng.name = name;

S = load(P_matfile);
fn = fieldnames(S);
Tpress = [];
for k = 1:numel(fn)
    c = S.(fn{k});
    if istable(c) || isnumeric(c)
        Tpress = c; break;
    end
end
if isempty(Tpress)
    error('No suitable table/numeric array found in %s.', P_matfile);
end

```

```

A = toNumericArray(Tpress);
rpm  = A(1, 2:end);
crank = A(2:end, 1);
P     = A(2:end, 2:end);

okP = isfinite(crank);
crank = crank(okP);
P     = P(okP,:);

[crank, idxP] = sort(crank);
P = P(idxP,:);

[crank, ia] = unique(crank, 'stable');
P = P(ia,:);

P_i = interp1(crank, P, desired_crank, 'linear', 'extrap');

Vd = disp_cc / Ncyl;
Vc = Vd/(CR - 1);

stroke_cm = stroke_mm/10;
rod_cm    = rod_mm/10;

Ap = Vd / stroke_cm;

r = stroke_cm/2;
l = rod_cm;

theta = deg2rad(desired_crank);
x = r*(1 - cos(theta)) + l - sqrt(l^2 - (r*sin(theta)).^2);
V_i = Vc + Ap.*x;

eng.rpm = rpm;
eng.P_i = P_i;
eng.V_i = V_i(:);

eng = add_metrics(eng);
eng = add_polytropic_indices(eng, desired_crank);
end

function eng = add_metrics(eng)

rpm = eng.rpm(:).';
P_i = eng.P_i;
V_i = eng.V_i(:);

W_bar_cm3 = zeros(length(rpm),1);
for j = 1:length(rpm)
    W_bar_cm3(j) = trapz(V_i, P_i(:,j));

```

```

end

W_J = 0.1 * abs(W_bar_cm3);

Vd_cm3 = max(V_i) - min(V_i);
IMEP_bar = abs(W_bar_cm3) / Vd_cm3;

eng.W_J = W_J;
eng.IMEP_bar = IMEP_bar;
eng.Vd_cm3 = Vd_cm3;
end

function plot_engine_suite(eng, label, desired_crank, P_units, V_units)
    % Pressure vs crank (all RPMs)
    figure('Name',[label ' - Pressure vs Crank (All RPMs)']);
    hold on; grid on;
    for j = 1:length(eng.rpm)
        plot(desired_crank, eng.P_i(:,j), 'LineWidth', 1.4);
    end
    xlabel('Crank angle (deg)');
    ylabel(['Pressure (', P_units, ')']);
    title([label ' - Pressure vs Crank (All RPMs)']);
    legend(compose('%0f rpm', eng.rpm), 'Location', 'best');
end

```

### Pressure versus crank angle (all RPMs)

For all engines, peak cylinder pressure occurs shortly after TDC and increases with engine speed, indicating increasing trapped charge energy and combustion intensity. The CLAAS engine consistently exhibits the highest peak pressures ( $\approx 140\text{--}150$  bar at higher RPM), while the FordFOX shows lower absolute peak pressures ( $\approx 80\text{--}90$  bar), consistent with its smaller displacement and operating range. The McLaren engine demonstrates intermediate peak pressures but maintains a relatively narrow pressure rise region, suggesting faster combustion phasing at higher speeds.

```

%Pressure Velocity vs crank (all RPMs)

dtheta = desired_crank(2) - desired_crank(1);
Pmat = eng.P_i;

dP_dtheta = NaN(size(Pmat));
dP_dtheta(2:end-1,:) = (Pmat(3:end,:) - Pmat(1:end-2,:)) ./ (2*dtheta);

dP_dtheta([1 end],:) = NaN;

figure('Name',[label ' - Pressure Velocity vs Crank (All RPMs)']);
hold on; grid on;
for j = 1:length(eng.rpm)
    plot(desired_crank, dP_dtheta(:,j), 'LineWidth', 1.4);
end
xlabel('Crank angle (deg)');
ylabel('Pressure Velocity (bar/deg)');

```

```
title([label ' - Pressure Velocity vs Crank (All RPMs)']);
legend(compose('%0f rpm', eng.rpm), 'Location', 'best');
```

### Pressure velocity (dP/dθ) and 3D surface plots

Pressure-velocity plots show a sharp positive peak near TDC followed by a negative region during expansion for all engines. The magnitude of the positive pressure rise rate increases with RPM, particularly for the McLaren engine, reflecting faster combustion at higher speeds. The 3D pressure-velocity and pressure surface plots clearly visualise this RPM dependence and confirm correct numerical differentiation and surface construction across the crank angle domain.

```
% P-V diagram (all RPMs)
figure('Name',[label ' - P-V Diagram (All RPMs)']);
hold on; grid on;
for j = 1:length(eng.rpm)
    plot(eng.V_i, eng.P_i(:,j), 'LineWidth', 1.4);
end
xlabel(['Volume (', V_units, ')']);
ylabel(['Pressure (', P_units, ')']);
title([label ' - P-V Diagram (All RPMs)']);
legend(compose('%0f rpm', eng.rpm), 'Location', 'best');
% LOG10(P) vs LOG10(V) (ALL RPMs)

P_floor_bar = 1e-6;
V_floor_m3 = 1e-12;

V_m3 = eng.V_i(:) * 1e-6;

figure('Name',[label ' - Log10(P) vs Log10(V) (All RPMs)']);
hold on; grid on;

x = log10(max(V_m3, V_floor_m3));
for j = 1:length(eng.rpm)
    P_bar = eng.P_i(:,j);
    y = log10(max(P_bar, P_floor_bar));
    plot(x, y, 'LineWidth', 1.4);
end
```

### Polytropic indices (log–log P–V analysis)

Compression polytropic indices remain relatively stable with RPM for each engine, typically around 1.25–1.35, indicating repeatable compression behaviour dominated by heat transfer and real-gas effects rather than ideal isentropic compression. Expansion polytropic indices are consistently lower ( $\approx 0.75$ –1.0) and show greater RPM dependence, reflecting increasing heat losses and blow-down effects during expansion. Differences between engines are evident, with the CLAAS engine exhibiting higher compression indices, consistent with higher absolute pressures and temperatures.

```
xlabel('log_{10} Volume (m^3)');
ylabel('log_{10} Pressure (bar)');
```

```

title([label ' - Log10(P) vs Log10(V) (All RPMs)']);
legend(compose('%0f rpm', eng.rpm), 'Location', 'best');

% Indicated work vs RPM
figure('Name',[label ' - Indicated Work per Cycle vs RPM']);
plot(eng.rpm, eng.W_J, '-o', 'LineWidth', 1.6);
grid on;
xlabel('Engine speed (rpm)');
ylabel('Indicated work per cycle (J)');
title([label ' - Indicated Work per Cycle vs RPM']);

```

## Pressure–volume (P–V) diagrams and indicated work

The P–V diagrams show increasing loop area with RPM up to a mid-range maximum for all engines, after which the loop area reduces at higher speeds. This trend is reflected directly in the indicated work per cycle plots. For the CLAAS and FordFOX engines, indicated work peaks in the mid-RPM range ( $\approx 1400$ – $1600$  rpm for CLAAS and  $\approx 2500$ – $3000$  rpm for FordFOX), before decreasing due to reduced volumetric efficiency and increased losses at higher speeds. The McLaren engine maintains a relatively flat indicated work plateau across a wider RPM range, consistent with a performance-oriented operating envelope.

```

% IMEP vs RPM
figure('Name',[label ' - IMEP vs RPM']);
plot(eng.rpm, eng.IMEP_bar, '-o', 'LineWidth', 1.6);
grid on;
xlabel('Engine speed (rpm)');
ylabel('IMEP (bar)');
title([label ' - IMEP vs RPM']);

```

## IMEP versus RPM

IMEP trends closely follow the indicated work behaviour. The CLAAS engine achieves peak IMEP of approximately 25–26 bar at mid-range speeds, while the FordFOX reaches around 26–27 bar before declining steadily at higher RPM. The McLaren engine sustains higher IMEP values ( $\approx 30$ – $31$  bar) across a broader speed range, indicating superior high-speed charge utilisation and combustion effectiveness. These trends confirm that IMEP provides a displacement-independent comparison of engine performance and aligns with expectations for the respective engine applications.

```

%3D SURFACES: P vs Crank vs RPM, and dP/dθ vs Crank vs RPM

CrankGrid = repmat(desired_crank(:), 1, length(eng.rpm));
RPMGrid   = repmat(eng.rpm(:).',    length(desired_crank), 1);
Pgrid     = eng.P_i;

figure('Name',[label ' - 3D Surface: Pressure vs Crank vs RPM']);
surf(RPMGrid, CrankGrid, Pgrid, 'EdgeColor', 'none');
grid on;
xlabel('RPM');
ylabel('Crank angle (deg)');
zlabel(['Pressure (' P_units ')']);
title(['Pressure vs Crank Angle vs RPM for ' label]);

```

```

view(45, 25);

dtheta = mean(diff(desired_crank));
dP_dtheta = gradient(Pgrid, dtheta);

figure('Name',[label ' - 3D Surface: Pressure Velocity vs Crank vs RPM']);
surf(RPMGrid, CrankGrid, dP_dtheta, 'EdgeColor', 'none');
grid on;
xlabel('RPM');
ylabel('Crank angle (deg)');
zlabel('Pressure Velocity (bar/rad)');
title(['Pressure Velocity vs Crank Angle vs RPM for ' label]);
view(45, 25);

%POLYTROPIC INDICES vs RPM

figure('Name',[label ' - Polytropic Indices vs RPM']);
hold on; grid on;
plot(eng.rpm, eng.n_comp, '-o', 'LineWidth', 1.6);
plot(eng.rpm, eng.n_exp, '-o', 'LineWidth', 1.6);
xlabel('RPM');
ylabel('Polytropic Index');
title(['Polytropic Indices vs RPM for ' label]);
legend({'Compression Index','Expansion Index'}, 'Location','best');

end

function A = toNumericArray(T)
    if istable(T)
        A = table2array(T);
    else
        A = T;
    end
end

function eng = add_polytropic_indices(eng, desired_crank)

P_floor = 1e-6;
V_floor = 1e-12;

comp_mask = (desired_crank >= -180) & (desired_crank <= 0);
exp_mask = (desired_crank >= 0) & (desired_crank <= 180);

Nrpm = numel(eng.rpm);
n_comp = NaN(Nrpm,1);
n_exp = NaN(Nrpm,1);

V = eng.V_i(:);

```

```

for j = 1:Nrpm
    Pj = eng.P_i(:,j);

    Vc = max(V(comp_mask), V_floor);
    Pc = max(Pj(comp_mask), P_floor);
    okc = isfinite(Vc) & isfinite(Pc);

    if nnz(okc) >= 5
        a = polyfit(log(Vc(okc)), log(Pc(okc)), 1);
        n_comp(j) = -a(1);
    end

    Ve = max(V(exp_mask), V_floor);
    Pe = max(Pj(exp_mask), P_floor);
    oke = isfinite(Ve) & isfinite(Pe);

    if nnz(oke) >= 5
        a = polyfit(log(Ve(oke)), log(Pe(oke)), 1);
        n_exp(j) = -a(1);
    end
end

eng.n_comp = n_comp;
eng.n_exp = n_exp;
end

```

## Overall Task 1 assessment

Task 1 successfully demonstrates a robust MATLAB post-processing workflow capable of handling multiple engines and RPM ranges. The pressure, P–V, work, IMEP, and polytropic index trends are physically consistent and align with expectations based on engine size and operating intent. This provides a validated foundation for the subsequent fuel comparison (Task 2) and combustion model refinement (Task 3).

## Task 2 - Alternative Fuels Investigation in Ricardo Wave

```

file = 'E85-Cyl1-Pressures.csv';

caseIdx = 7;
rpm = 7000 - 500*caseIdx;

T = readtable(file, 'VariableNamingRule','preserve');

T(1,:) = [];

crankColIdx = 2*caseIdx - 1;
pressColIdx = 2*caseIdx;

theta_raw = double(T(:, crankColIdx));
P_raw = double(T(:, pressColIdx));

```

```

mask = ~isnan(theta_raw) & ~isnan(P_raw);
theta_raw = theta_raw(mask);
P_raw = P_raw(mask);

[theta_raw, sortIdx] = sort(theta_raw);
P_raw = P_raw(sortIdx);

[theta_raw, uniqIdx] = unique(theta_raw, 'stable');
P_raw = P_raw(uniqIdx);

desired_crank = (-360:1:359)';

P_pa = interp1(theta_raw, P_raw, desired_crank, 'linear', NaN);
P_bar = P_pa / 1e5;

[Pk_bar, idxPk] = max(P_bar, [], 'omitnan');
theta_Pk = desired_crank(idxPk);

dP = nan(size(P_bar));
dP(2:end-1) = (P_bar(3:end) - P_bar(1:end-2)) ./ (desired_crank(3:end) -
desired_crank(1:end-2));

[dPmax_bar_per_deg, idxdP] = max(dP, [], 'omitnan');
theta_dPmax = desired_crank(idxdP);

fprintf('E85 Cyl1 | %d rpm | Case %d\n', rpm, caseIdx);

```

```
E85 Cyl1 | 3500 rpm | Case 7
```

```
fprintf(' Pmax = %.2f bar at %.0f deg\n', Pk_bar, theta_Pk);
```

```
Pmax = 79.47 bar at 22 deg
```

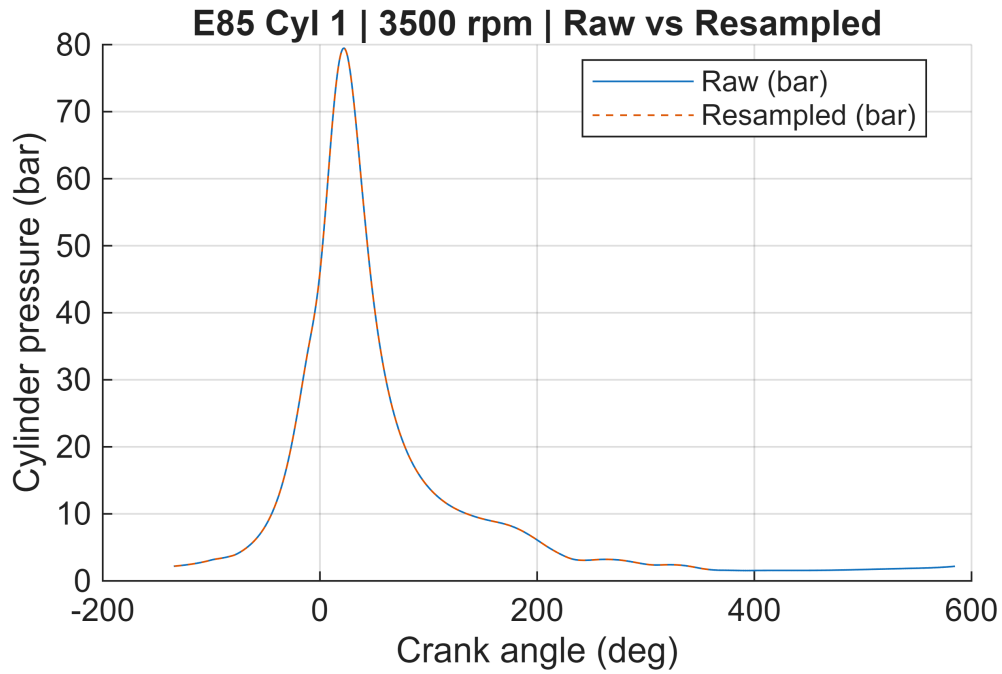
```
fprintf(' (dP/dθ)max = %.2f bar/deg at %.0f deg\n', dPmax_bar_per_deg,
theta_dPmax);
```

```
(dP/dθ)max = 2.11 bar/deg at 7 deg
```

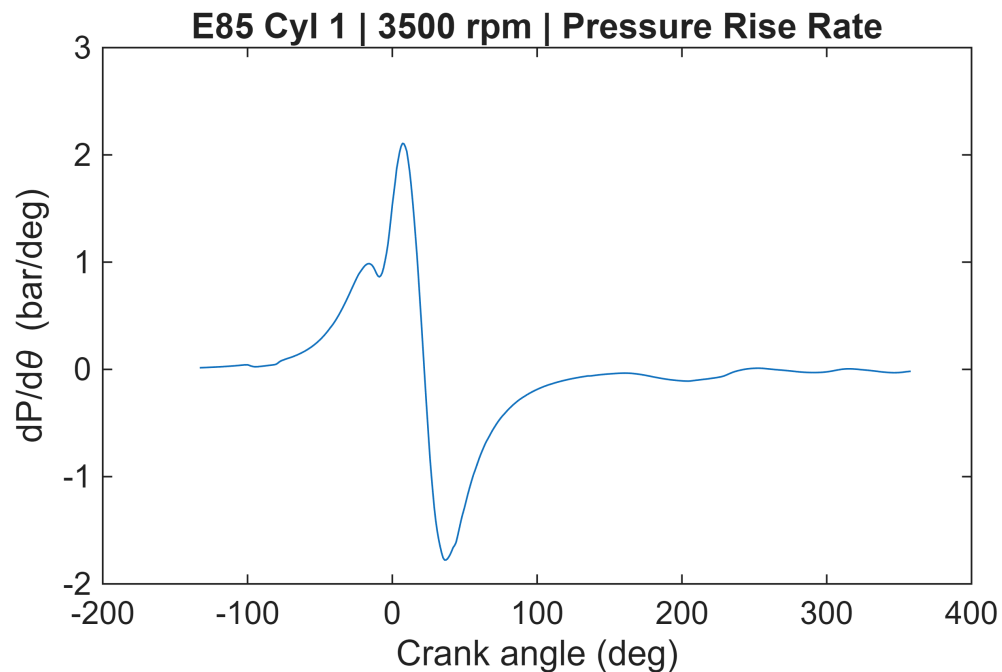
```

figure; hold on; grid on;
plot(theta_raw, P_raw/1e5, '-', 'DisplayName','Raw (bar)');
plot(desired_crank, P_bar, '--', 'DisplayName','Resampled (bar)');
xlabel('Crank angle (deg)');
ylabel('Cylinder pressure (bar)');
title(sprintf('E85 Cyl 1 | %d rpm | Raw vs Resampled', rpm));
legend('Location','best');

```



```
figure; grid on;  
plot(desired_crank, dP, '-');  
xlabel('Crank angle (deg)');  
ylabel('dP/d\theta (bar/deg)');  
title(sprintf('E85 Cyl 1 | %d rpm | Pressure Rise Rate', rpm));
```



An engine speed of 3500 rpm was selected as a representative mid-range operating condition for initial validation of the pressure resampling and combustion metric extraction methodology, as combustion behaviour at this speed is well-developed and avoids low-speed and high-speed artefacts.

```

file = 'E85-Cyl1-Pressures.csv';

T = readtable(file, 'VariableNamingRule','preserve');

T(1,:) = [];

desired_crank = (-360:1:359)';      % deg

nCases = 12;
rpm_vec = zeros(nCases,1);

Pmax_bar      = nan(nCases,1);
theta_Pmax_deg = nan(nCases,1);
dPmax_bar_per_deg = nan(nCases,1);
theta_dPmax_deg = nan(nCases,1);

Pbar_all = nan(numel(desired_crank), nCases);

for caseIdx = 1:nCases

    rpm = 7000 - 500*caseIdx;
    rpm_vec(caseIdx) = rpm;

    crankColIdx = 2*caseIdx - 1;
    pressColIdx = 2*caseIdx;

    theta_raw = double(T{:, crankColIdx});
    P_raw      = double(T{:, pressColIdx});

    mask = ~isnan(theta_raw) & ~isnan(P_raw);
    theta_raw = theta_raw(mask);
    P_raw      = P_raw(mask);

    [theta_raw, sortIdx] = sort(theta_raw);
    P_raw = P_raw(sortIdx);

    [theta_raw, uniqIdx] = unique(theta_raw, 'stable');
    P_raw = P_raw(uniqIdx);

    P_pa = interp1(theta_raw, P_raw, desired_crank, 'linear', NaN);
    P_bar = P_pa / 1e5;

    Pbar_all(:,caseIdx) = P_bar;

    [Pk_bar, idxPk] = max(P_bar, [], 'omitnan');
    Pmax_bar(caseIdx) = Pk_bar;
    theta_Pmax_deg(caseIdx) = desired_crank(idxPk);

    dP = nan(size(P_bar));

```

```

dP(2:end-1) = (P_bar(3:end) - P_bar(1:end-2)) ./ (desired_crank(3:end) -
desired_crank(1:end-2));

```

```

[dPmax, idxdP] = max(dP, [], 'omitnan');
dPmax_bar_per_deg(caseIdx) = dPmax;
theta_dPmax_deg(caseIdx) = desired_crank(idxdP);

```

```

end

```

```

[rpm_vec, sortOrder] = sort(rpm_vec, 'ascend');
Pmax_bar = Pmax_bar(sortOrder);
theta_Pmax_deg = theta_Pmax_deg(sortOrder);
dPmax_bar_per_deg = dPmax_bar_per_deg(sortOrder);
theta_dPmax_deg = theta_dPmax_deg(sortOrder);
Pbar_all = Pbar_all(:, sortOrder);

```

```

fprintf('\nE85 Cyl1 Summary:\n');

```

E85 Cyl1 Summary:

```

fprintf('RPM    Pmax(bar)    theta_Pmax(deg)    dPmax(bar/deg)    theta_dPmax(deg)\n');

```

```

RPM    Pmax(bar)    theta_Pmax(deg)    dPmax(bar/deg)    theta_dPmax(deg)

```

```

for i = 1:nCases
    fprintf('%4d    %8.2f    %8.0f    %8.2f    %8.0f\n', ...
        rpm_vec(i), Pmax_bar(i), theta_Pmax_deg(i), dPmax_bar_per_deg(i),
        theta_dPmax_deg(i));
end

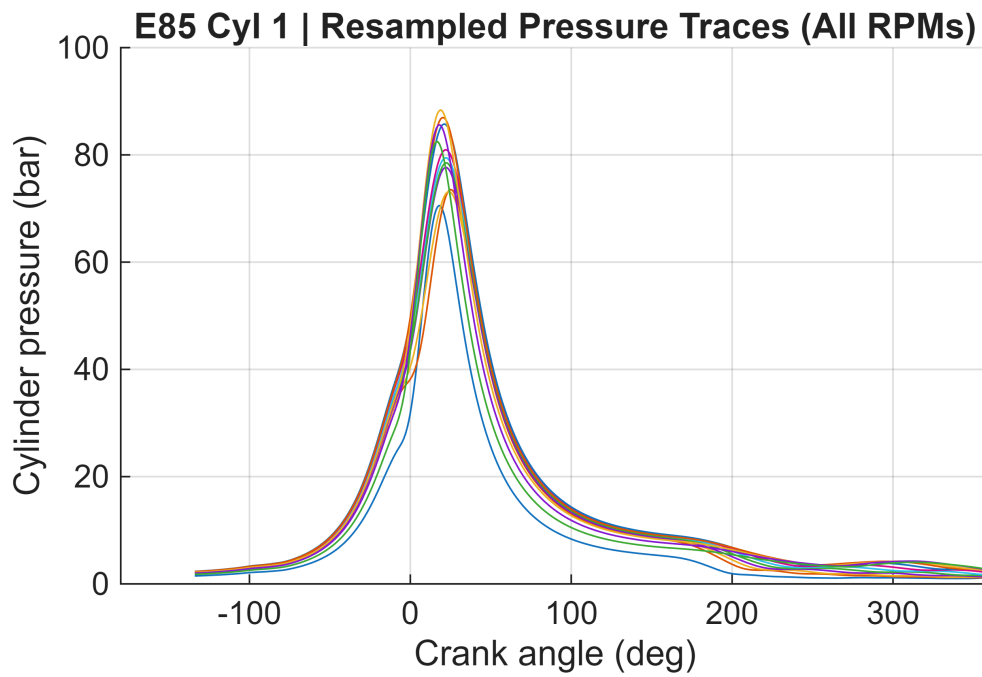
```

|      |       |    |      |    |
|------|-------|----|------|----|
| 1000 | 70.57 | 18 | 3.01 | 7  |
| 1500 | 73.54 | 25 | 2.18 | 13 |
| 2000 | 73.16 | 24 | 1.93 | 10 |
| 2500 | 77.68 | 22 | 2.14 | 8  |
| 3000 | 78.52 | 22 | 2.14 | 8  |
| 3500 | 79.47 | 22 | 2.11 | 7  |
| 4000 | 80.96 | 22 | 2.11 | 8  |
| 4500 | 85.77 | 21 | 2.37 | 7  |
| 5000 | 86.97 | 20 | 2.53 | 6  |
| 5500 | 88.37 | 19 | 2.94 | 6  |
| 6000 | 85.62 | 18 | 2.98 | 6  |
| 6500 | 82.48 | 17 | 3.22 | 5  |

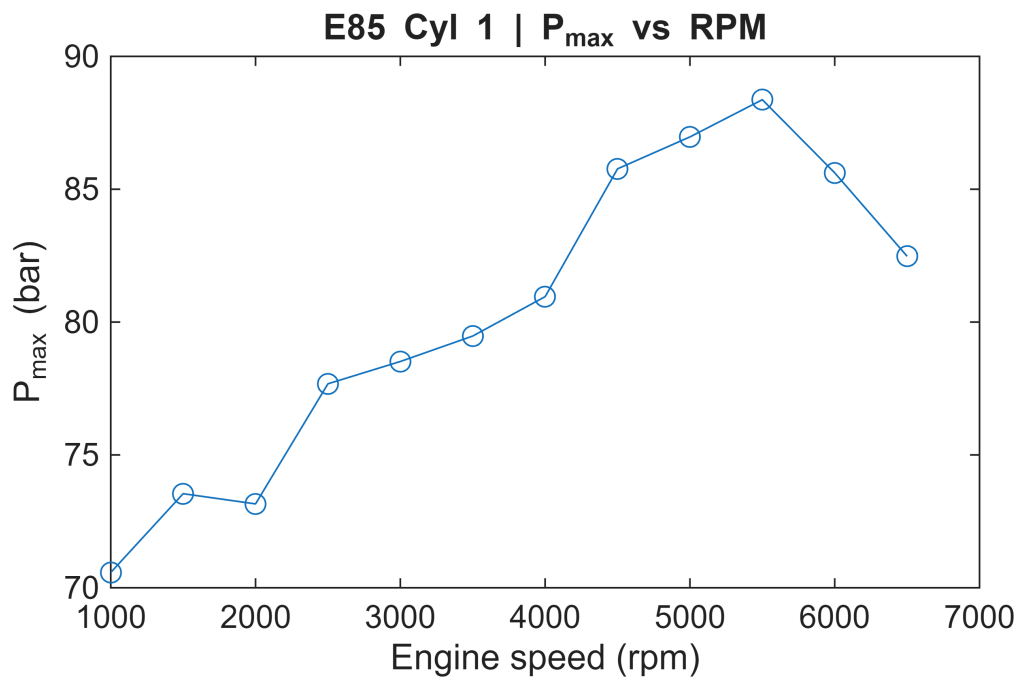
```

% Pressure traces overlay
figure; hold on; grid on;
plot(desired_crank, Pbar_all);
xlabel('Crank angle (deg)');
ylabel('Cylinder pressure (bar)');
title('E85 Cyl 1 | Resampled Pressure Traces (All RPMs)');
xlim([-180 360]);

```



```
% Pmax vs RPM
figure; grid on;
plot(rpm_vec, Pmax_bar, '-o');
xlabel('Engine speed (rpm)');
ylabel('P_{max} (bar)');
title('E85 Cyl 1 | P_{max} vs RPM');
```

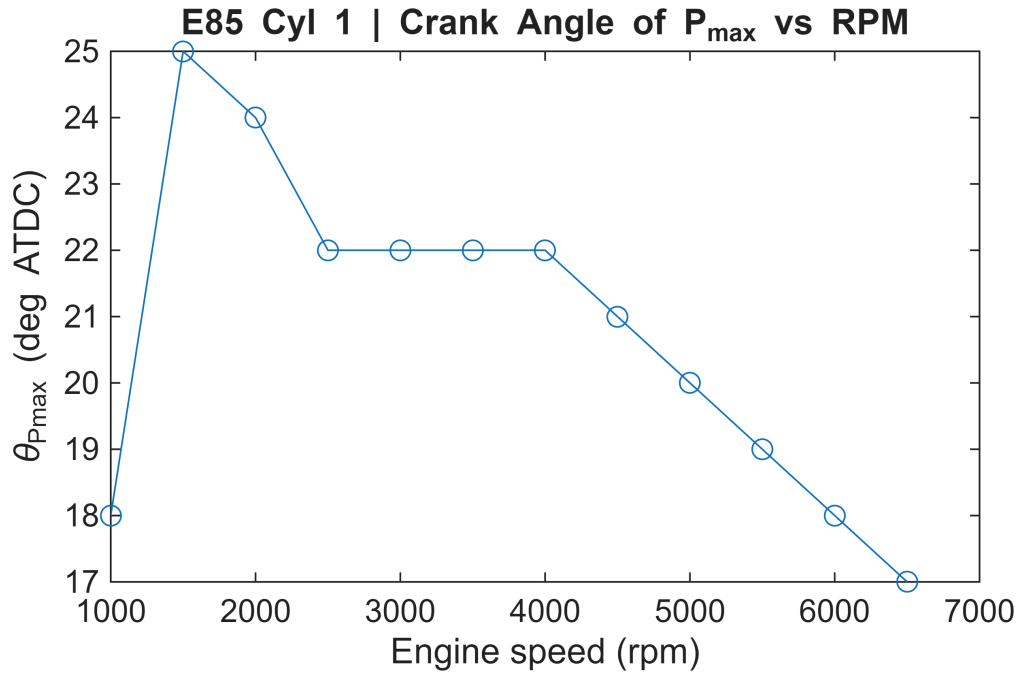


```
% Theta at Pmax vs RPM
figure; grid on;
```

```

plot(rpm_vec, theta_Pmax_deg, '-o');
xlabel('Engine speed (rpm)');
ylabel('\theta_{Pmax} (deg ATDC)');
title('E85 Cyl 1 | Crank Angle of P_{max} vs RPM');

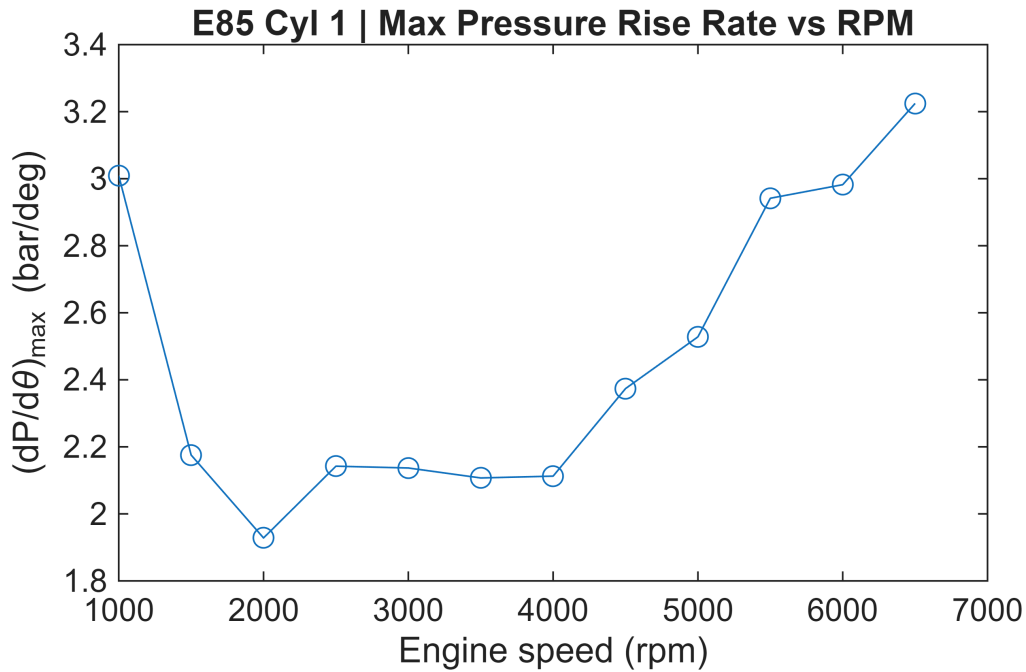
```



```

% Max pressure rise rate vs RPM
figure; grid on;
plot(rpm_vec, dPmax_bar_per_deg, '-o');
xlabel('Engine speed (rpm)');
ylabel('(dP/d\theta)_{max} (bar/deg)');
title('E85 Cyl 1 | Max Pressure Rise Rate vs RPM');

```



## E85 - Metrics

```
% E85 Fuel Metrics
fuelTag = "E85";
files = { ...
    'E85-Cyl1-Pressures.csv', ...
    'E85-Cyl2-Pressures.csv', ...
    'E85-Cyl3-Pressures.csv', ...
    'E85-Cyl4-Pressures.csv'};

nCyl    = numel(files);
nCases = 12;

desired_crank = (-360:1:359)';
nTheta = numel(desired_crank);

rpm_vec = zeros(nCases,1);
for caseIdx = 1:nCases
    rpm_vec(caseIdx) = 7000 - 500*caseIdx;
end
[rpm_vec, rpmSort] = sort(rpm_vec, 'ascend');

Pmax_bar      = nan(nCases, nCyl);
theta_Pmax_deg = nan(nCases, nCyl);
dPmax_bar_per_deg = nan(nCases, nCyl);
theta_dPmax_deg = nan(nCases, nCyl);

Pbar_all = nan(nTheta, nCases, nCyl);
```

```

for c = 1:nCyl
    T = readtable(files{c}, 'VariableNamingRule','preserve');
    T(1,:) = [];

    for caseIdx = 1:nCases

        crankColIdx = 2*caseIdx - 1;
        pressColIdx = 2*caseIdx;

        theta_raw = double(T(:, crankColIdx));
        P_raw      = double(T(:, pressColIdx));

        mask = ~isnan(theta_raw) & ~isnan(P_raw);
        theta_raw = theta_raw(mask);
        P_raw     = P_raw(mask);

        [theta_raw, sortIdx] = sort(theta_raw);
        P_raw = P_raw(sortIdx);

        [theta_raw, uniqIdx] = unique(theta_raw, 'stable');
        P_raw = P_raw(uniqIdx);

        P_pa = interp1(theta_raw, P_raw, desired_crank, 'linear', NaN);
        P_bar = P_pa / 1e5;

        Pbar_all(:, caseIdx, c) = P_bar;

        [Pk_bar, idxPk] = max(P_bar, [], 'omitnan');
        Pmax_bar(caseIdx, c) = Pk_bar;
        theta_Pmax_deg(caseIdx, c) = desired_crank(idxPk);

        dP = nan(size(P_bar));
        dP(2:end-1) = (P_bar(3:end) - P_bar(1:end-2)) ./ (desired_crank(3:end) -
desired_crank(1:end-2));

        [dPmax, idxdP] = max(dP, [], 'omitnan');
        dPmax_bar_per_deg(caseIdx, c) = dPmax;
        theta_dPmax_deg(caseIdx, c) = desired_crank(idxdP);

    end
end

Pmax_bar      = Pmax_bar(rpmSort, :);
theta_Pmax_deg = theta_Pmax_deg(rpmSort, :);
dPmax_bar_per_deg = dPmax_bar_per_deg(rpmSort, :);
theta_dPmax_deg = theta_dPmax_deg(rpmSort, :);
Pbar_all      = Pbar_all(:, rpmSort, :);

Pmax_mean      = mean(Pmax_bar, 2, 'omitnan');

```

```

theta_Pmax_mean    = mean(theta_Pmax_deg, 2, 'omitnan');
dPmax_mean         = mean(dPmax_bar_per_deg, 2, 'omitnan');
theta_dPmax_mean   = mean(theta_dPmax_deg, 2, 'omitnan');

Pmax_std           = std(Pmax_bar, 0, 2, 'omitnan');
theta_Pmax_std     = std(theta_Pmax_deg, 0, 2, 'omitnan');
dPmax_std          = std(dPmax_bar_per_deg, 0, 2, 'omitnan');

fprintf('\n%s Cylinder-Averaged Summary (mean ± std across 4 cylinders)\n',
fuelTag);

```

E85 Cylinder-Averaged Summary (mean ± std across 4 cylinders)

```
fprintf('RPM    Pmax(bar)        theta_Pmax(deg)    dPmax(bar/deg)\n');
```

```
RPM    Pmax(bar)        theta_Pmax(deg)    dPmax(bar/deg)
```

```

for i = 1:nCases
    fprintf('%4d  %6.2f ± %4.2f    %6.2f ± %4.2f    %6.2f ± %4.2f\n', ...
            rpm_vec(i), Pmax_mean(i), Pmax_std(i), theta_Pmax_mean(i),
            theta_Pmax_std(i), dPmax_mean(i), dPmax_std(i));
end

```

|      |              |              |             |
|------|--------------|--------------|-------------|
| 1000 | 68.03 ± 1.72 | 18.75 ± 0.50 | 3.05 ± 0.03 |
| 1500 | 72.94 ± 0.40 | 26.50 ± 1.00 | 2.41 ± 0.16 |
| 2000 | 72.35 ± 0.54 | 25.50 ± 1.00 | 2.14 ± 0.14 |
| 2500 | 76.48 ± 0.81 | 23.50 ± 1.00 | 2.32 ± 0.12 |
| 3000 | 77.66 ± 0.59 | 24.25 ± 1.50 | 2.33 ± 0.13 |
| 3500 | 78.65 ± 0.61 | 23.50 ± 1.00 | 2.29 ± 0.12 |
| 4000 | 80.54 ± 0.36 | 23.50 ± 1.00 | 2.32 ± 0.14 |
| 4500 | 85.57 ± 0.72 | 22.50 ± 1.00 | 2.57 ± 0.14 |
| 5000 | 87.02 ± 1.23 | 21.50 ± 1.00 | 2.72 ± 0.14 |
| 5500 | 88.10 ± 0.99 | 19.75 ± 0.50 | 3.09 ± 0.11 |
| 6000 | 85.50 ± 0.21 | 19.00 ± 0.82 | 3.12 ± 0.09 |
| 6500 | 82.56 ± 0.43 | 17.75 ± 0.50 | 3.33 ± 0.08 |

```
% Pressure traces at one RPM
```

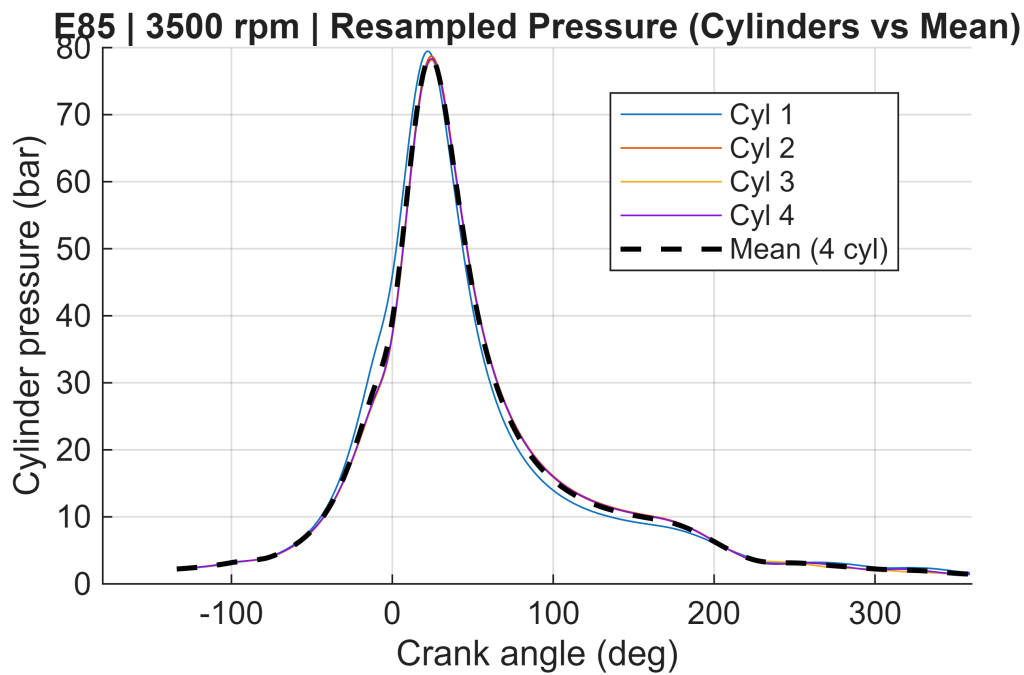
```

targetRPM = 3500;
[~, idxRPM] = min(abs(rpm_vec - targetRPM));

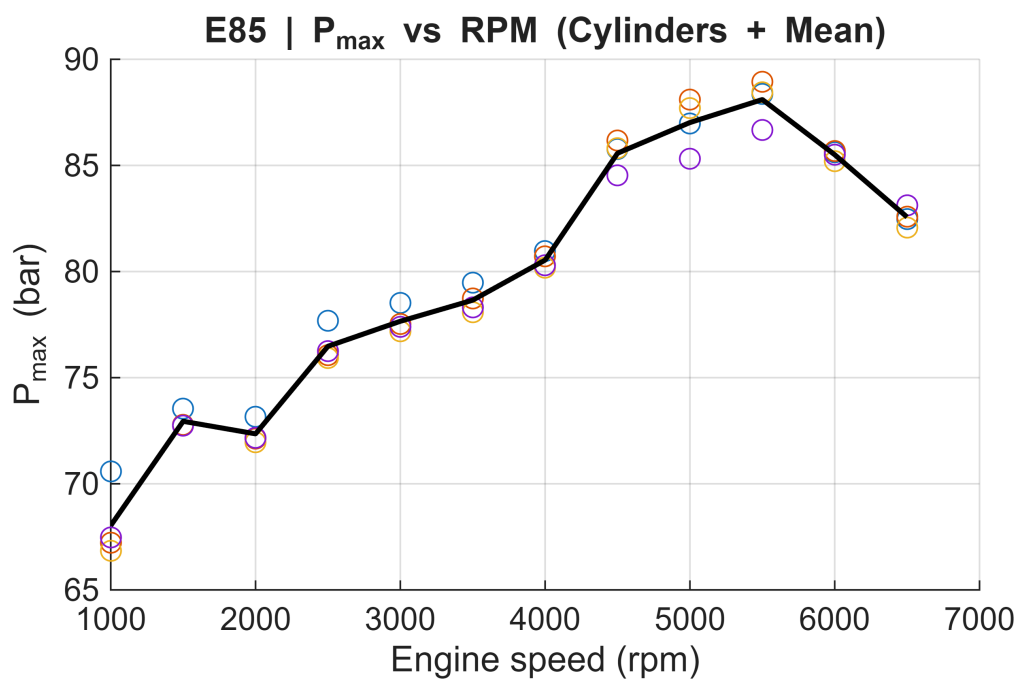
figure; hold on; grid on;
for c = 1:nCyl
    plot(desired_crank, Pbar_all(:, idxRPM, c), 'DisplayName', sprintf('Cyl %d',
c));
end
plot(desired_crank, mean(Pbar_all(:, idxRPM, :)), 3, 'omitnan', 'k--', 'LineWidth',
1.5, ...
'DisplayName', 'Mean (4 cyl)');
xlabel('Crank angle (deg)');
ylabel('Cylinder pressure (bar)');
title(sprintf('%s | %d rpm | Resampled Pressure (Cylinders vs Mean)', fuelTag,
rpm_vec(idxRPM)));
xlim([-180 360]);

```

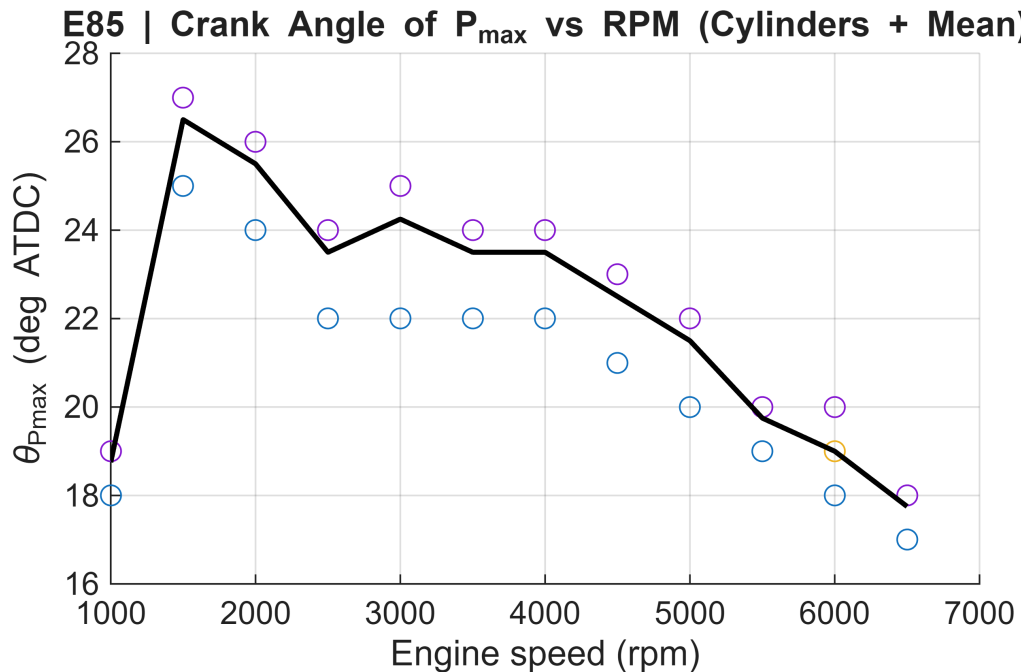
```
legend('Location','best');
```



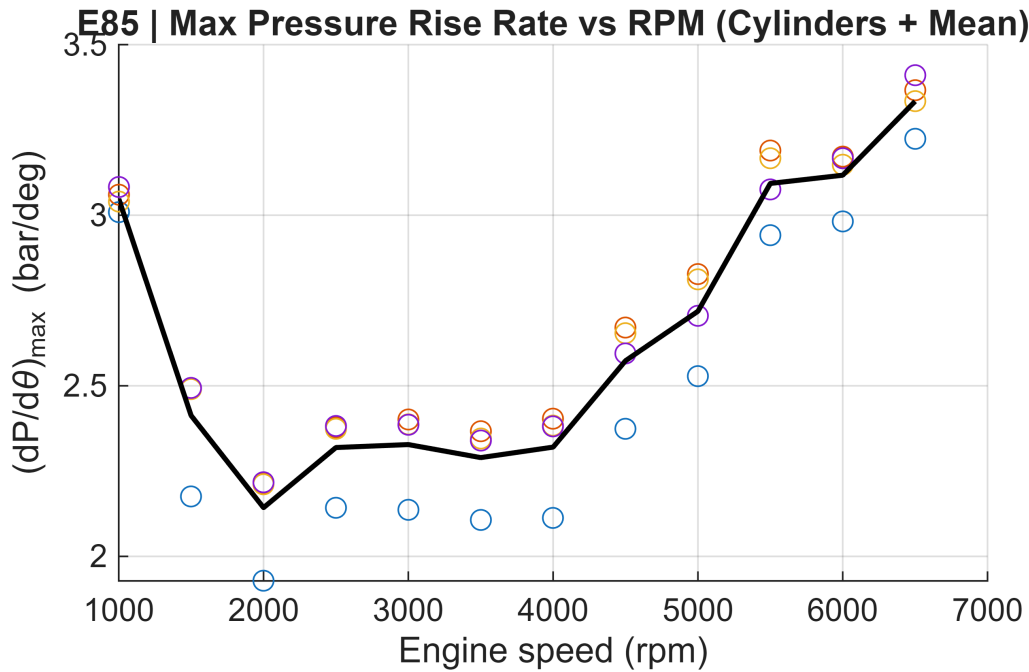
```
% Mean Pmax vs RPM
figure; hold on; grid on;
plot(rpm_vec, Pmax_bar, 'o');
plot(rpm_vec, Pmax_mean, '-k', 'LineWidth', 1.5);
xlabel('Engine speed (rpm)');
ylabel('P_{max} (bar)');
title(sprintf('%s | P_{max} vs RPM (Cylinders + Mean)', fuelTag));
```



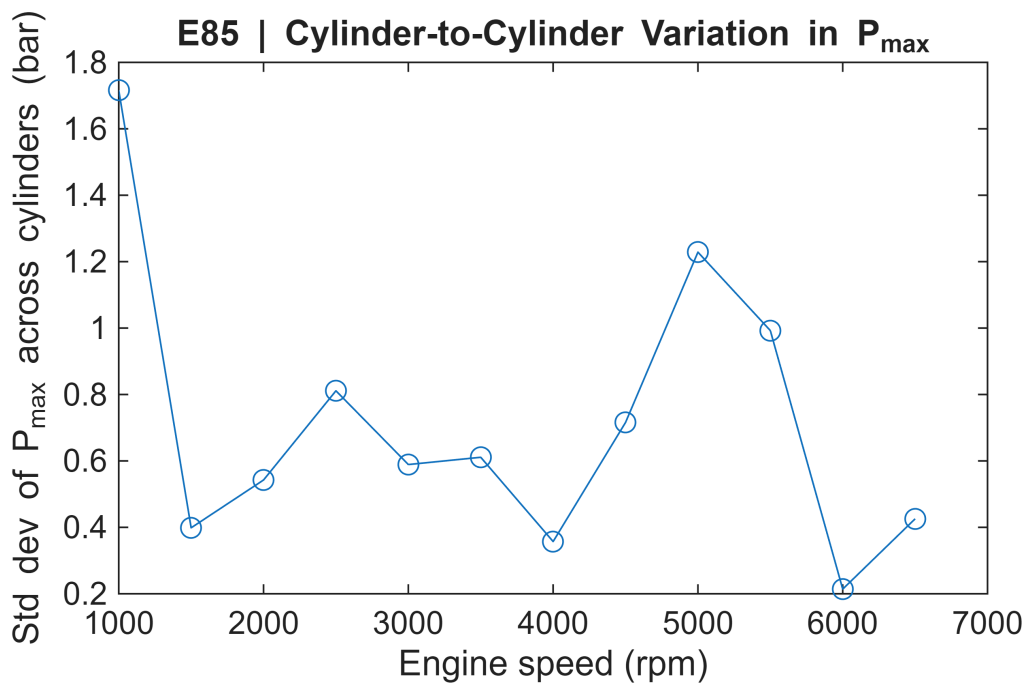
```
% Mean theta_Pmax vs RPM
figure; hold on; grid on;
plot(rpm_vec, theta_Pmax_deg, 'o');
plot(rpm_vec, theta_Pmax_mean, '-k', 'LineWidth', 1.5);
xlabel('Engine speed (rpm)');
ylabel('\theta_{Pmax} (deg ATDC)');
title(sprintf('%s | Crank Angle of P_{max} vs RPM (Cylinders + Mean)', fuelTag));
```



```
% Mean max pressure rise rate vs RPM
figure; hold on; grid on;
plot(rpm_vec, dPmax_bar_per_deg, 'o');
plot(rpm_vec, dPmax_mean, '-k', 'LineWidth', 1.5);
xlabel('Engine speed (rpm)');
ylabel('(dP/d\theta)_{max} (bar/deg)');
title(sprintf('%s | Max Pressure Rise Rate vs RPM (Cylinders + Mean)', fuelTag));
```



```
% Cylinder spread (std dev) vs RPM
figure; grid on;
plot(rpm_vec, Pmax_std, '-o');
xlabel('Engine speed (rpm)');
ylabel('Std dev of P_{max} across cylinders (bar)');
title(sprintf('%s | Cylinder-to-Cylinder Variation in P_{max}', fuelTag));
```



```
save('Results_E85.mat', ...
     'rpm_vec', ...
```

```
'Pmax_mean', 'theta_Pmax_mean', 'dPmax_mean', ...  
'Pmax_std', 'theta_Pmax_std', 'dPmax_std');
```

Across the tested speed range, E85 exhibits a moderate peak cylinder pressure relative to RON95, with Pmax increasing steadily from low speed and peaking in the mid-range before declining at high RPM. The crank angle of peak pressure advances with increasing speed, consistent with reduced residence time and fixed combustion phasing in the WAVE model. Maximum pressure rise rate increases significantly above 4500 rpm, indicating faster combustion once turbulence levels rise. Despite lower energy density, E85 achieves comparable torque and BMEP to RON95, while showing a consistently lower BSFC trend, reflecting improved knock tolerance and combustion efficiency.

## E100 - Metrics:

```
fuelTag = "E100";  
files = { ...  
    'E100-Cyl1-Pressures.csv', ...  
    'E100-Cyl2-Pressures.csv', ...  
    'E100-Cyl3-Pressures.csv', ...  
    'E100-Cyl4-Pressures.csv'};  
  
nCyl = numel(files);  
nCases = 12;  
  
desired_crank = (-360:1:359)';  
nTheta = numel(desired_crank);  
  
rpm_vec = zeros(nCases,1);  
for caseIdx = 1:nCases  
    rpm_vec(caseIdx) = 7000 - 500*caseIdx;  
end  
[rpm_vec, rpmSort] = sort(rpm_vec, 'ascend');  
  
Pmax_bar = nan(nCases, nCyl);  
theta_Pmax_deg = nan(nCases, nCyl);  
dPmax_bar_per_deg = nan(nCases, nCyl);  
theta_dPmax_deg = nan(nCases, nCyl);  
  
Pbar_all = nan(nTheta, nCases, nCyl);  
  
for c = 1:nCyl  
    T = readtable(files{c}, 'VariableNamingRule','preserve');  
    T(1,:) = [];  
  
    for caseIdx = 1:nCases  
  
        crankColIdx = 2*caseIdx - 1;  
        pressColIdx = 2*caseIdx;
```

```

theta_raw = double(T(:, crankColIdx));
P_raw     = double(T(:, pressColIdx));

mask = ~isnan(theta_raw) & ~isnan(P_raw);
theta_raw = theta_raw(mask);
P_raw     = P_raw(mask);

[theta_raw, sortIdx] = sort(theta_raw);
P_raw = P_raw(sortIdx);

[theta_raw, uniqIdx] = unique(theta_raw, 'stable');
P_raw = P_raw(uniqIdx);

P_pa = interp1(theta_raw, P_raw, desired_crank, 'linear', NaN);
P_bar = P_pa / 1e5;

Pbar_all(:, caseIdx, c) = P_bar;

[Pk_bar, idxPk] = max(P_bar, [], 'omitnan');
Pmax_bar(caseIdx, c) = Pk_bar;
theta_Pmax_deg(caseIdx, c) = desired_crank(idxPk);

dP = nan(size(P_bar));
dP(2:end-1) = (P_bar(3:end) - P_bar(1:end-2)) ./ (desired_crank(3:end) -
desired_crank(1:end-2));

[dPmax, idxdP] = max(dP, [], 'omitnan');
dPmax_bar_per_deg(caseIdx, c) = dPmax;
theta_dPmax_deg(caseIdx, c) = desired_crank(idxdP);

end
end

Pmax_bar = Pmax_bar(rpmSort, :);
theta_Pmax_deg = theta_Pmax_deg(rpmSort, :);
dPmax_bar_per_deg = dPmax_bar_per_deg(rpmSort, :);
theta_dPmax_deg = theta_dPmax_deg(rpmSort, :);
Pbar_all = Pbar_all(:, rpmSort, :);

Pmax_mean = mean(Pmax_bar, 2, 'omitnan');
theta_Pmax_mean = mean(theta_Pmax_deg, 2, 'omitnan');
dPmax_mean = mean(dPmax_bar_per_deg, 2, 'omitnan');
theta_dPmax_mean = mean(theta_dPmax_deg, 2, 'omitnan');

Pmax_std = std(Pmax_bar, 0, 2, 'omitnan');
theta_Pmax_std = std(theta_Pmax_deg, 0, 2, 'omitnan');
dPmax_std = std(dPmax_bar_per_deg, 0, 2, 'omitnan');

fprintf('\n%s Cylinder-Averaged Summary (mean ± std across 4 cylinders)\n',
fuelTag);

```

E100 Cylinder-Averaged Summary (mean  $\pm$  std across 4 cylinders)

```
fprintf('RPM    Pmax(bar)        theta_Pmax(deg)    dPmax(bar/deg)\n');
```

```
RPM    Pmax(bar)        theta_Pmax(deg)    dPmax(bar/deg)
```

```
for i = 1:nCases
    fprintf('%4d  %6.2f  $\pm$  %4.2f    %6.2f  $\pm$  %4.2f    %6.2f  $\pm$  %4.2f\n', ...
        rpm_vec(i), Pmax_mean(i), Pmax_std(i), theta_Pmax_mean(i),
        theta_Pmax_std(i), dPmax_mean(i), dPmax_std(i));
end
```

|      |                  |                  |                 |
|------|------------------|------------------|-----------------|
| 1000 | 68.11 $\pm$ 1.63 | 18.75 $\pm$ 0.50 | 3.09 $\pm$ 0.03 |
| 1500 | 72.08 $\pm$ 0.22 | 26.50 $\pm$ 1.00 | 2.45 $\pm$ 0.15 |
| 2000 | 71.48 $\pm$ 0.40 | 25.50 $\pm$ 1.00 | 2.17 $\pm$ 0.14 |
| 2500 | 75.57 $\pm$ 0.74 | 24.00 $\pm$ 1.41 | 2.34 $\pm$ 0.11 |
| 3000 | 76.71 $\pm$ 0.54 | 24.25 $\pm$ 1.50 | 2.35 $\pm$ 0.12 |
| 3500 | 77.66 $\pm$ 0.60 | 23.50 $\pm$ 1.00 | 2.32 $\pm$ 0.12 |
| 4000 | 79.54 $\pm$ 0.35 | 23.50 $\pm$ 1.00 | 2.35 $\pm$ 0.14 |
| 4500 | 84.51 $\pm$ 0.70 | 22.50 $\pm$ 1.00 | 2.60 $\pm$ 0.14 |
| 5000 | 85.94 $\pm$ 1.23 | 21.75 $\pm$ 1.26 | 2.74 $\pm$ 0.14 |
| 5500 | 87.00 $\pm$ 0.93 | 20.00 $\pm$ 0.82 | 3.11 $\pm$ 0.11 |
| 6000 | 84.47 $\pm$ 0.21 | 19.50 $\pm$ 1.00 | 3.13 $\pm$ 0.09 |
| 6500 | 81.56 $\pm$ 0.46 | 17.75 $\pm$ 0.50 | 3.33 $\pm$ 0.07 |

```
% Pressure traces at one RPM
```

```
targetRPM = 3500;
```

```
[~, idxRPM] = min(abs(rpm_vec - targetRPM));
```

```
figure; hold on; grid on;
```

```
for c = 1:nCyl
```

```
    plot(desired_crank, Pbar_all(:, idxRPM, c), 'DisplayName', sprintf('Cyl %d',
    c));
```

```
end
```

```
plot(desired_crank, mean(Pbar_all(:, idxRPM, :)), 3, 'omitnan', 'k--', 'LineWidth',
1.5, ...
```

```
    'DisplayName', 'Mean (4 cyl)');
```

```
xlabel('Crank angle (deg)');
```

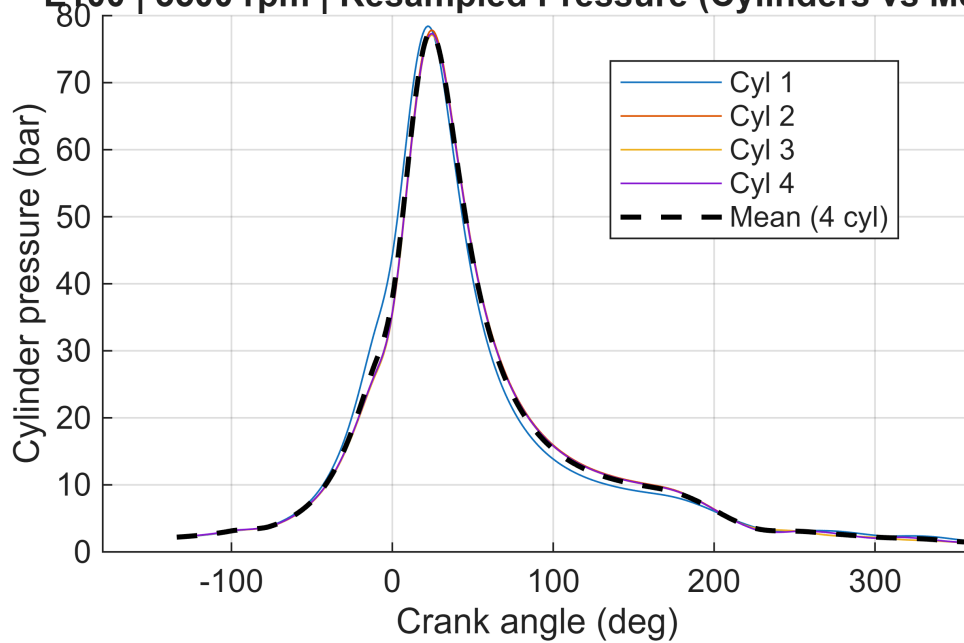
```
ylabel('Cylinder pressure (bar)');
```

```
title(sprintf('%s | %d rpm | Resampled Pressure (Cylinders vs Mean)', fuelTag,
rpm_vec(idxRPM)));
```

```
xlim([-180 360]);
```

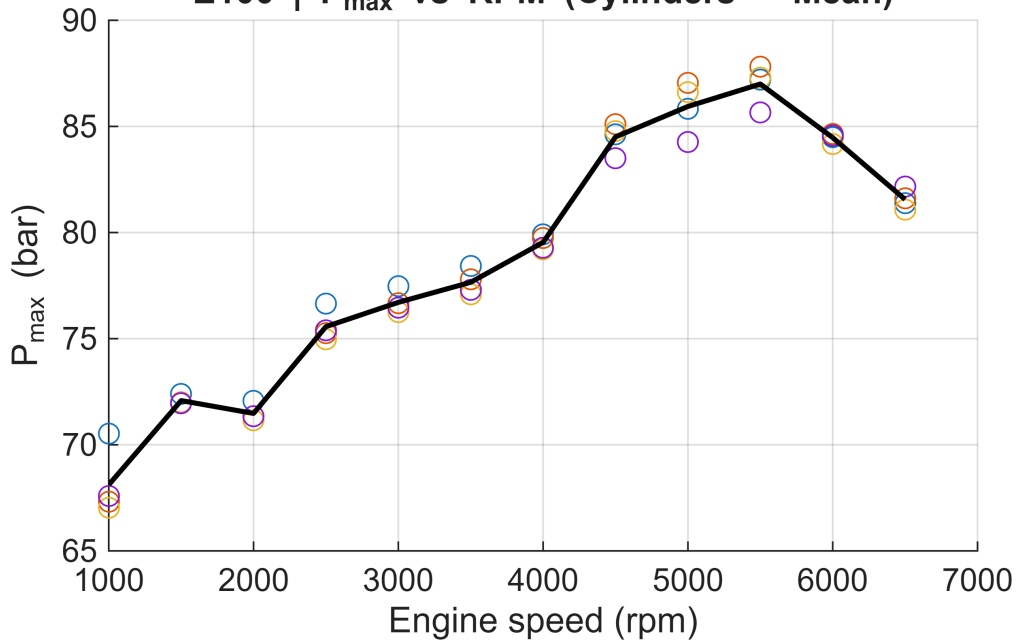
```
legend('Location', 'best');
```

### E100 | 3500 rpm | Resampled Pressure (Cylinders vs Mean)



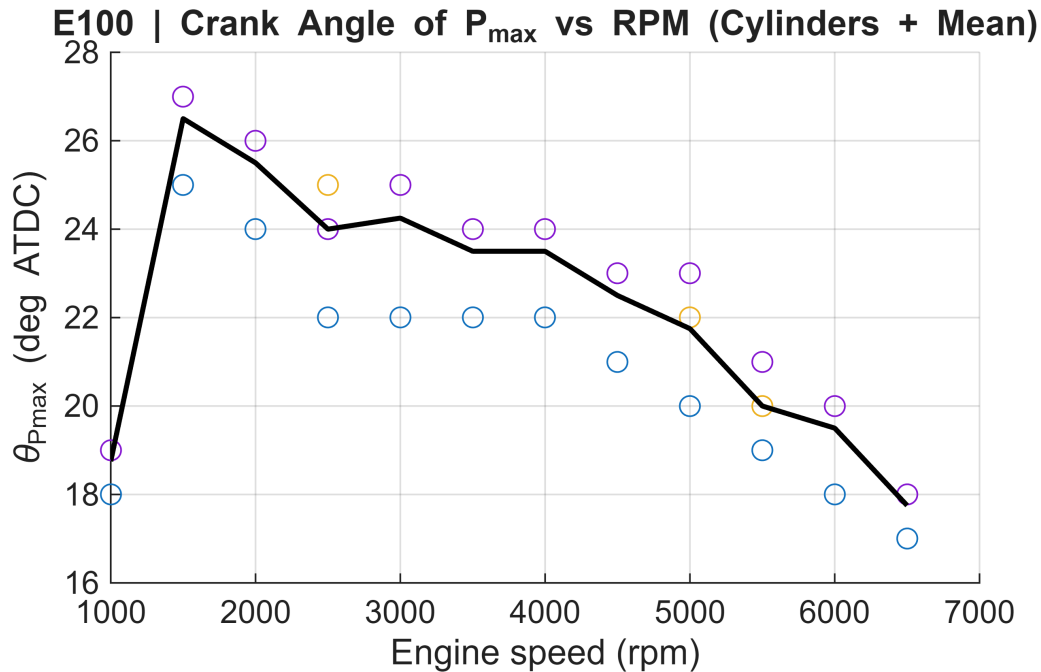
```
% Mean Pmax vs RPM
figure; hold on; grid on;
plot(rpm_vec, Pmax_bar, 'o');
plot(rpm_vec, Pmax_mean, '-k', 'LineWidth', 1.5);
xlabel('Engine speed (rpm)');
ylabel('P_{max} (bar)');
title(sprintf('%s | P_{max} vs RPM (Cylinders + Mean)', fuelTag));
```

### E100 | $P_{\max}$ vs RPM (Cylinders + Mean)

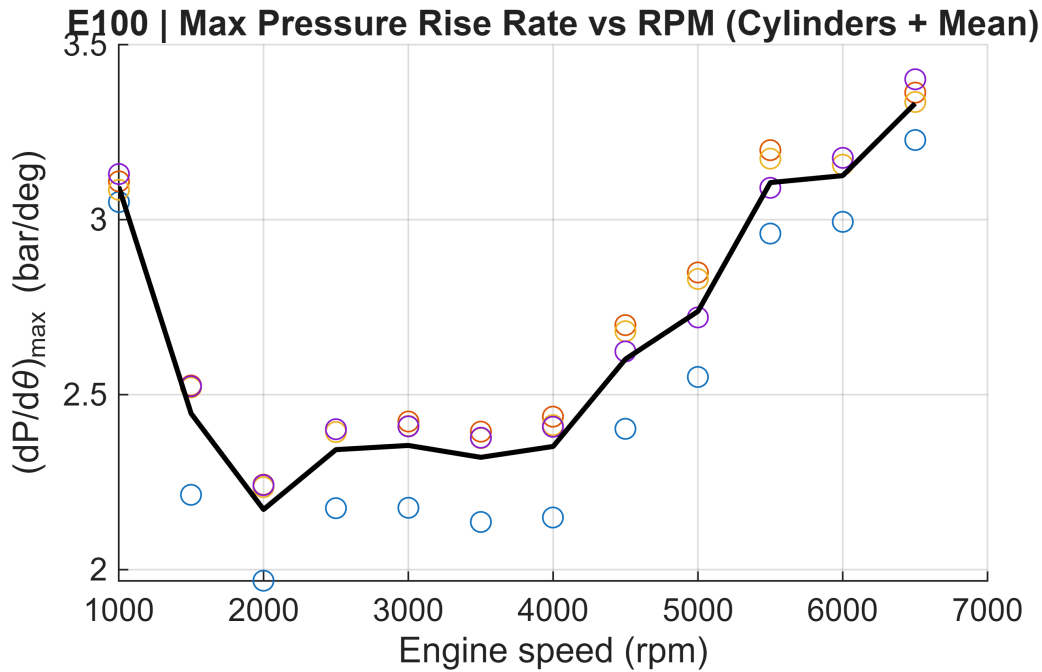


```
% Mean theta_Pmax vs RPM
```

```
figure; hold on; grid on;
plot(rpm_vec, theta_Pmax_deg, 'o');
plot(rpm_vec, theta_Pmax_mean, '-k', 'LineWidth', 1.5);
xlabel('Engine speed (rpm)');
ylabel('\theta_{Pmax} (deg ATDC)');
title(sprintf('%s | Crank Angle of P_{max} vs RPM (Cylinders + Mean)', fuelTag));
```

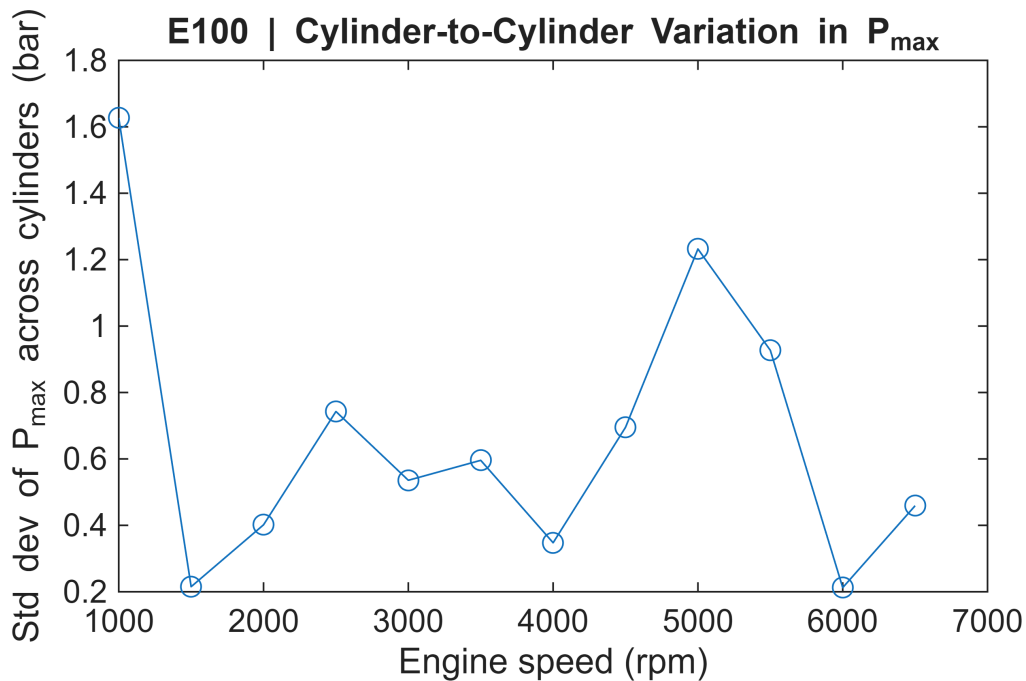


```
% Mean max pressure rise rate vs RPM
figure; hold on; grid on;
plot(rpm_vec, dPmax_bar_per_deg, 'o');
plot(rpm_vec, dPmax_mean, '-k', 'LineWidth', 1.5);
xlabel('Engine speed (rpm)');
ylabel('(dP/d\theta)_{max} (bar/deg)');
title(sprintf('%s | Max Pressure Rise Rate vs RPM (Cylinders + Mean)', fuelTag));
```



% Cylinder spread (std dev) vs RPM

```
figure; grid on;
plot(rpm_vec, Pmax_std, '-o');
xlabel('Engine speed (rpm)');
ylabel('Std dev of P_{max} across cylinders (bar)');
title(sprintf('%s | Cylinder-to-Cylinder Variation in P_{max}', fuelTag));
```



```
save('Results_E100.mat', ...
     'rpm_vec', ...
```

```
'Pmax_mean', 'theta_Pmax_mean', 'dPmax_mean', ...  
'Pmax_std', 'theta_Pmax_std', 'dPmax_std');
```

E100 follows similar pressure and phasing trends to E85 but with slightly reduced Pmax at most speeds. The pressure rise rate increases more sharply at high RPM, indicating a faster burn once combustion is fully developed. The crank angle of peak pressure closely follows E85, confirming that combustion phasing is largely governed by the imposed WAVE settings rather than fuel chemistry. Torque and BMEP remain comparable to the other fuels; however, BSFC is consistently highest due to the significantly lower lower heating value of ethanol.

## RON95 - Metrics:

```
fuelTag = "RON95";  
files = { ...  
    'RON95-Cyl1-Pressures.csv', ...  
    'RON95-Cyl2-Pressures.csv', ...  
    'RON95-Cyl3-Pressures.csv', ...  
    'RON95-Cyl4-Pressures.csv'};  
  
nCyl = numel(files);  
nCases = 12;  
  
desired_crank = (-360:1:359)';  
nTheta = numel(desired_crank);  
  
rpm_vec = zeros(nCases,1);  
for caseIdx = 1:nCases  
    rpm_vec(caseIdx) = 7000 - 500*caseIdx;  
end  
[rpm_vec, rpmSort] = sort(rpm_vec, 'ascend');  
  
Pmax_bar = nan(nCases, nCyl);  
theta_Pmax_deg = nan(nCases, nCyl);  
dPmax_bar_per_deg = nan(nCases, nCyl);  
theta_dPmax_deg = nan(nCases, nCyl);  
  
Pbar_all = nan(nTheta, nCases, nCyl);  
  
for c = 1:nCyl  
    T = readtable(files{c}, 'VariableNamingRule','preserve');  
    T(1,:) = [];  
  
    for caseIdx = 1:nCases  
  
        crankColIdx = 2*caseIdx - 1;  
        pressColIdx = 2*caseIdx;  
  
        theta_raw = double(T{:, crankColIdx});
```

```

P_raw      = double(T{:, pressColIdx});

mask = ~isnan(theta_raw) & ~isnan(P_raw);
theta_raw = theta_raw(mask);
P_raw     = P_raw(mask);

[theta_raw, sortIdx] = sort(theta_raw);
P_raw = P_raw(sortIdx);

[theta_raw, uniqIdx] = unique(theta_raw, 'stable');
P_raw = P_raw(uniqIdx);

P_pa = interp1(theta_raw, P_raw, desired_crank, 'linear', NaN);
P_bar = P_pa / 1e5;

Pbar_all(:, caseIdx, c) = P_bar;

[Pk_bar, idxPk] = max(P_bar, [], 'omitnan');
Pmax_bar(caseIdx, c) = Pk_bar;
theta_Pmax_deg(caseIdx, c) = desired_crank(idxPk);

dP = nan(size(P_bar));
dP(2:end-1) = (P_bar(3:end) - P_bar(1:end-2)) ./ (desired_crank(3:end) -
desired_crank(1:end-2));

[dPmax, idxdP] = max(dP, [], 'omitnan');
dPmax_bar_per_deg(caseIdx, c) = dPmax;
theta_dPmax_deg(caseIdx, c) = desired_crank(idxdP);

end
end

Pmax_bar      = Pmax_bar(rpmSort, :);
theta_Pmax_deg = theta_Pmax_deg(rpmSort, :);
dPmax_bar_per_deg = dPmax_bar_per_deg(rpmSort, :);
theta_dPmax_deg = theta_dPmax_deg(rpmSort, :);
Pbar_all      = Pbar_all(:, rpmSort, :);

Pmax_mean      = mean(Pmax_bar, 2, 'omitnan');
theta_Pmax_mean = mean(theta_Pmax_deg, 2, 'omitnan');
dPmax_mean      = mean(dPmax_bar_per_deg, 2, 'omitnan');
theta_dPmax_mean = mean(theta_dPmax_deg, 2, 'omitnan');

Pmax_std      = std(Pmax_bar, 0, 2, 'omitnan');
theta_Pmax_std = std(theta_Pmax_deg, 0, 2, 'omitnan');
dPmax_std      = std(dPmax_bar_per_deg, 0, 2, 'omitnan');

fprintf('\n%s Cylinder-Averaged Summary (mean ± std across 4 cylinders)\n',
fuelTag);

```

RON95 Cylinder-Averaged Summary (mean  $\pm$  std across 4 cylinders)

```
fprintf('RPM    Pmax(bar)        theta_Pmax(deg)    dPmax(bar/deg)\n');
```

```
RPM    Pmax(bar)        theta_Pmax(deg)    dPmax(bar/deg)
```

```
for i = 1:nCases
    fprintf('%4d  %6.2f  $\pm$  %4.2f    %6.2f  $\pm$  %4.2f    %6.2f  $\pm$  %4.2f\n', ...
        rpm_vec(i), Pmax_mean(i), Pmax_std(i), theta_Pmax_mean(i),
        theta_Pmax_std(i), dPmax_mean(i), dPmax_std(i));
end
```

|      |                  |                  |                 |
|------|------------------|------------------|-----------------|
| 1000 | 68.10 $\pm$ 1.81 | 18.75 $\pm$ 0.50 | 2.92 $\pm$ 0.05 |
| 1500 | 75.93 $\pm$ 0.55 | 25.75 $\pm$ 0.50 | 2.32 $\pm$ 0.18 |
| 2000 | 75.47 $\pm$ 0.74 | 25.00 $\pm$ 1.41 | 2.04 $\pm$ 0.15 |
| 2500 | 79.76 $\pm$ 1.01 | 23.50 $\pm$ 1.00 | 2.23 $\pm$ 0.13 |
| 3000 | 80.99 $\pm$ 0.74 | 23.50 $\pm$ 1.00 | 2.24 $\pm$ 0.14 |
| 3500 | 82.59 $\pm$ 1.35 | 23.25 $\pm$ 1.50 | 2.22 $\pm$ 0.14 |
| 4000 | 84.09 $\pm$ 0.60 | 23.25 $\pm$ 1.50 | 2.24 $\pm$ 0.15 |
| 4500 | 89.27 $\pm$ 0.75 | 22.50 $\pm$ 1.00 | 2.50 $\pm$ 0.15 |
| 5000 | 90.76 $\pm$ 1.23 | 21.50 $\pm$ 1.00 | 2.66 $\pm$ 0.15 |
| 5500 | 91.85 $\pm$ 1.18 | 19.50 $\pm$ 1.00 | 3.07 $\pm$ 0.13 |
| 6000 | 89.12 $\pm$ 0.37 | 18.75 $\pm$ 0.50 | 3.11 $\pm$ 0.10 |
| 6500 | 85.88 $\pm$ 0.45 | 17.00 $\pm$ 0.82 | 3.36 $\pm$ 0.09 |

```
% Pressure traces at one RPM
```

```
targetRPM = 3500;
```

```
[~, idxRPM] = min(abs(rpm_vec - targetRPM));
```

```
figure; hold on; grid on;
```

```
for c = 1:nCyl
```

```
    plot(desired_crank, Pbar_all(:, idxRPM, c), 'DisplayName', sprintf('Cyl %d',
    c));
end
```

```
plot(desired_crank, mean(Pbar_all(:, idxRPM, :)), 3, 'omitnan', 'k--', 'LineWidth',
1.5, ...
```

```
    'DisplayName', 'Mean (4 cyl)');
```

```
xlabel('Crank angle (deg)');
```

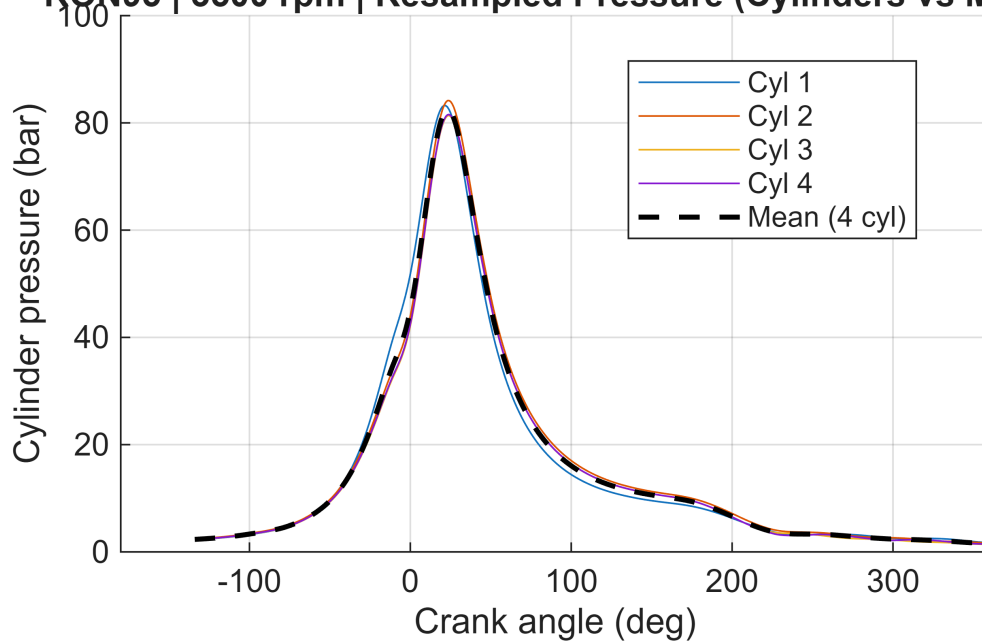
```
ylabel('Cylinder pressure (bar)');
```

```
title(sprintf('%s | %d rpm | Resampled Pressure (Cylinders vs Mean)', fuelTag,
rpm_vec(idxRPM)));
```

```
xlim([-180 360]);
```

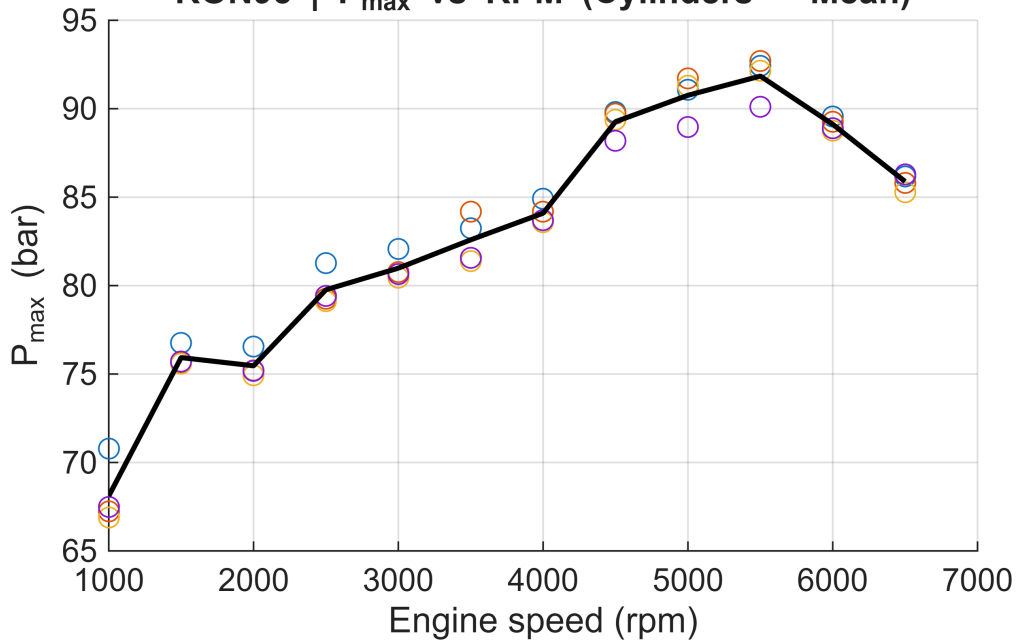
```
legend('Location', 'best');
```

### RON95 | 3500 rpm | Resampled Pressure (Cylinders vs Mean)



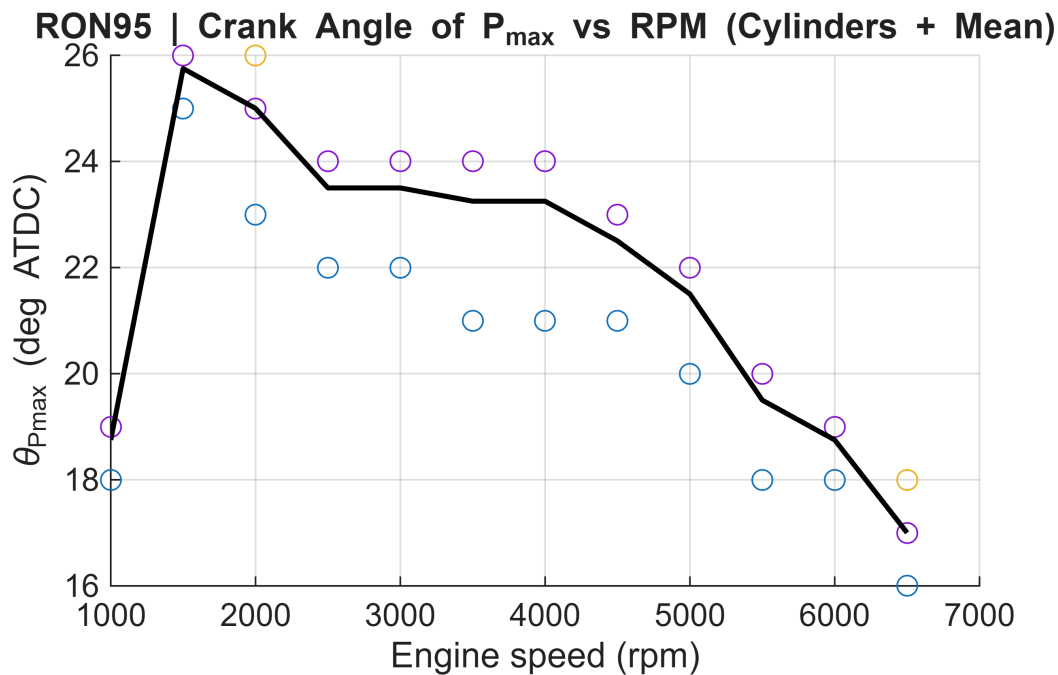
```
% Mean Pmax vs RPM
figure; hold on; grid on;
plot(rpm_vec, Pmax_bar, 'o');
plot(rpm_vec, Pmax_mean, '-k', 'LineWidth', 1.5);
xlabel('Engine speed (rpm)');
ylabel('P_{max} (bar)');
title(sprintf('%s | P_{max} vs RPM (Cylinders + Mean)', fuelTag));
```

### RON95 | $P_{\max}$ vs RPM (Cylinders + Mean)



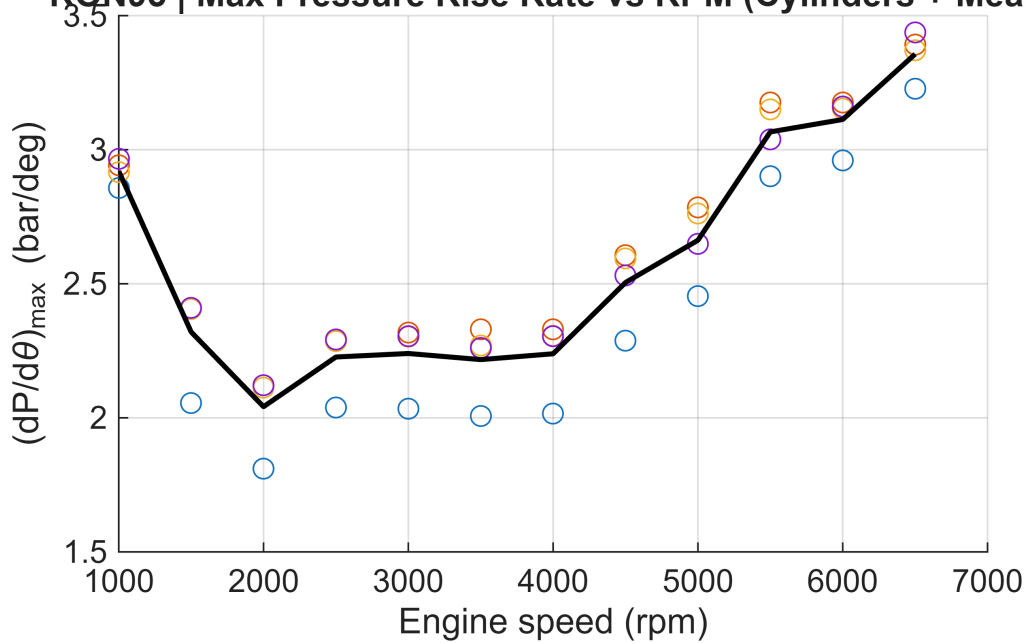
```
% Mean theta_Pmax vs RPM
```

```
figure; hold on; grid on;
plot(rpm_vec, theta_Pmax_deg, 'o');
plot(rpm_vec, theta_Pmax_mean, '-k', 'LineWidth', 1.5);
xlabel('Engine speed (rpm)');
ylabel('\theta_{Pmax} (deg ATDC)');
title(sprintf('%s | Crank Angle of P_{max} vs RPM (Cylinders + Mean)', fuelTag));
```



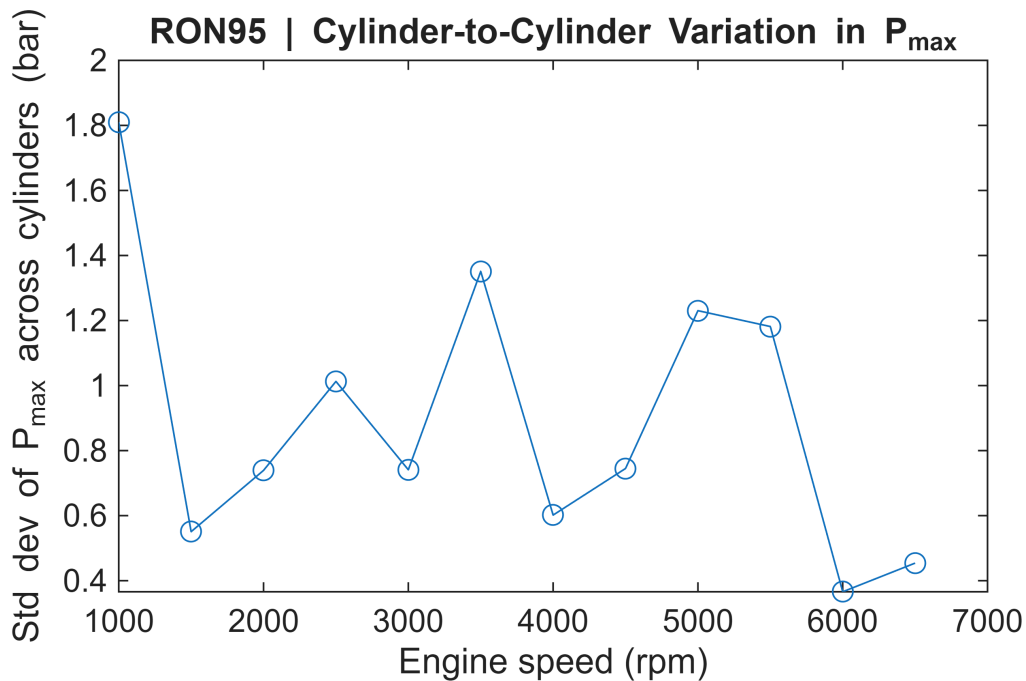
```
% Mean max pressure rise rate vs RPM
figure; hold on; grid on;
plot(rpm_vec, dPmax_bar_per_deg, 'o');
plot(rpm_vec, dPmax_mean, '-k', 'LineWidth', 1.5);
xlabel('Engine speed (rpm)');
ylabel('(dP/d\theta)_{max} (bar/deg)');
title(sprintf('%s | Max Pressure Rise Rate vs RPM (Cylinders + Mean)', fuelTag));
```

**RON95 | Max Pressure Rise Rate vs RPM (Cylinders + Mean)**



% Cylinder spread (std dev) vs RPM

```
figure; grid on;
plot(rpm_vec, Pmax_std, '-o');
xlabel('Engine speed (rpm)');
ylabel('Std dev of P_{max} across cylinders (bar)');
title(sprintf('%s | Cylinder-to-Cylinder Variation in P_{max}', fuelTag));
```



```
save('Results_ROM95.mat', ...
     'rpm_vec', ...
```

```
'Pmax_mean', 'theta_Pmax_mean', 'dPmax_mean', ...
'Pmax_std', 'theta_Pmax_std', 'dPmax_std');
```

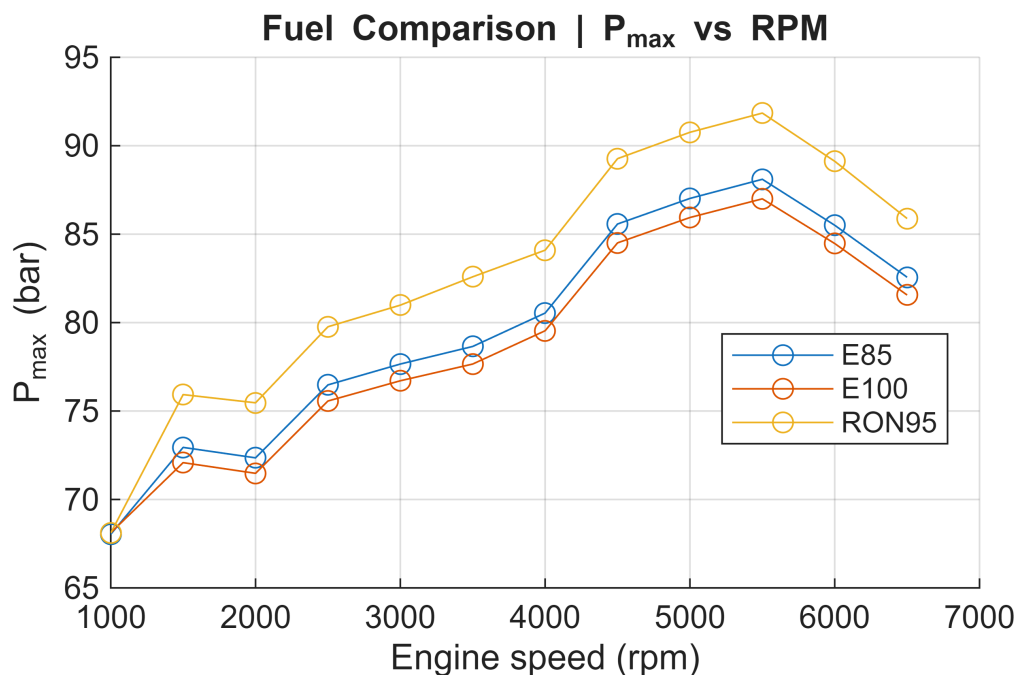
RON95 consistently produces the highest peak cylinder pressure across the speed range, particularly above 4000 rpm. The crank angle of peak pressure is slightly later at low speed but converges with ethanol fuels at higher RPM. Pressure rise rates are lower at mid-range speeds, indicating a less aggressive burn compared to ethanol fuels. RON95 achieves the lowest BSFC across all speeds, reflecting its higher energy density, while delivering similar torque and BMEP under the controlled WAVE operating conditions.

## Fuel comparisons:

```
A = load('Results_E85.mat');
B = load('Results_E100.mat');
C = load('Results_ROM95.mat');

rpm = A.rpm_vec;

% Pmax vs RPM
figure; hold on; grid on;
plot(rpm, A.Pmax_mean, '-o', 'DisplayName', 'E85');
plot(rpm, B.Pmax_mean, '-o', 'DisplayName', 'E100');
plot(rpm, C.Pmax_mean, '-o', 'DisplayName', 'ROM95');
xlabel('Engine speed (rpm)');
ylabel('P_{max} (bar)');
title('Fuel Comparison | P_{max} vs RPM');
legend('Location', 'best');
```

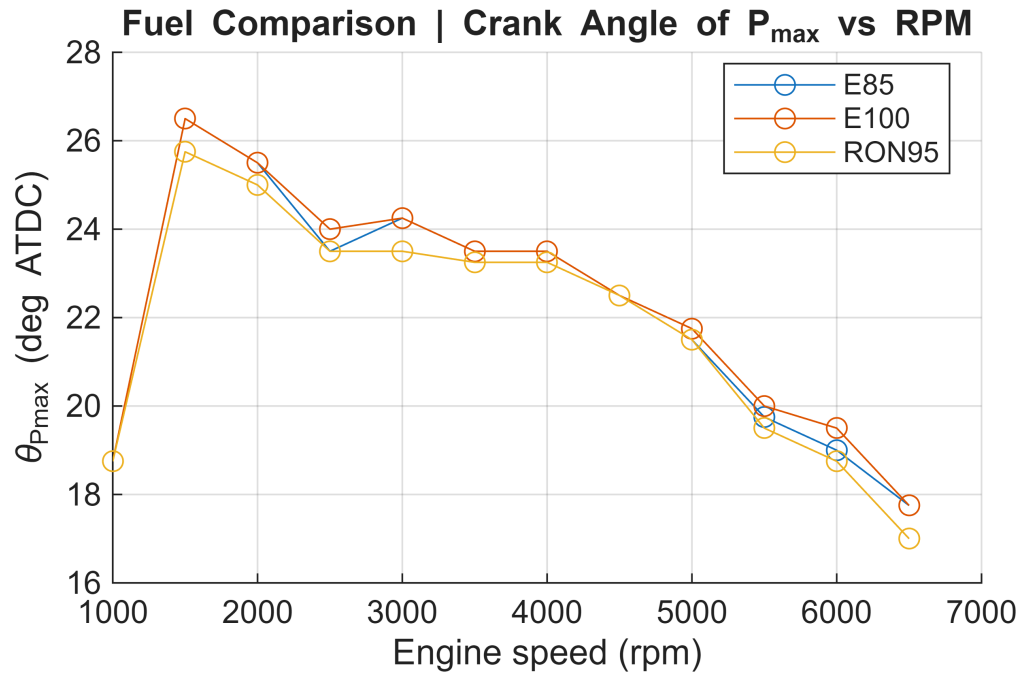


```
% theta_Pmax vs RPM
```

```

figure; hold on; grid on;
plot(rpm, A.theta_Pmax_mean, '-o', 'DisplayName','E85');
plot(rpm, B.theta_Pmax_mean, '-o', 'DisplayName','E100');
plot(rpm, C.theta_Pmax_mean, '-o', 'DisplayName','RON95');
xlabel('Engine speed (rpm)');
ylabel('\theta_{Pmax} (deg ATDC)');
title('Fuel Comparison | Crank Angle of P_{max} vs RPM');
legend('Location','best');

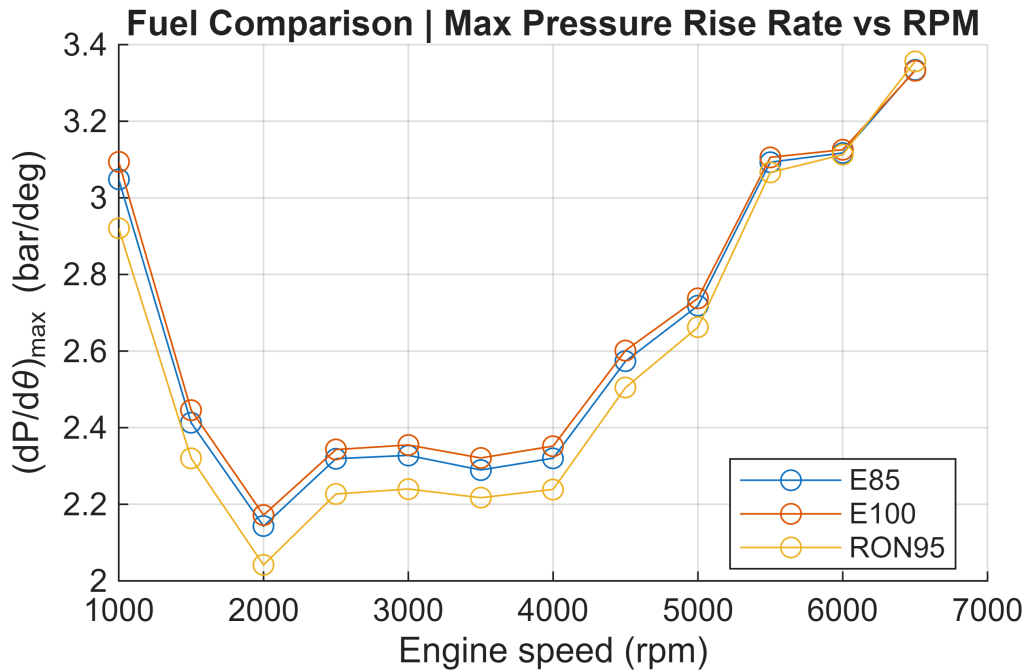
```



```

% max pressure rise rate vs RPM
figure; hold on; grid on;
plot(rpm, A.dPmax_mean, '-o', 'DisplayName','E85');
plot(rpm, B.dPmax_mean, '-o', 'DisplayName','E100');
plot(rpm, C.dPmax_mean, '-o', 'DisplayName','RON95');
xlabel('Engine speed (rpm)');
ylabel('(dP/d\theta)_{max} (bar/deg)');
title('Fuel Comparison | Max Pressure Rise Rate vs RPM');
legend('Location','best');

```



```
% comparison table
fprintf('\nFuel comparison table (means):\n');
```

```
Fuel comparison table (means):
```

```
fprintf('RPM | Pmax E85  E100  RON95 | theta_Pmax E85  E100  RON95 | dPmax E85
E100  RON95\n');
```

```
RPM | Pmax E85  E100  RON95 | theta_Pmax E85  E100  RON95 | dPmax E85  E100  RON95
```

```
for i = 1:numel(rpm)
    fprintf('%4d | %7.2f %7.2f %7.2f | %10.2f %6.2f %6.2f | %8.2f %6.2f %6.2f\n',
    ...
        rpm(i), A.Pmax_mean(i), B.Pmax_mean(i), C.Pmax_mean(i), ...
        A.theta_Pmax_mean(i), B.theta_Pmax_mean(i), C.theta_Pmax_mean(i), ...
        A.dPmax_mean(i), B.dPmax_mean(i), C.dPmax_mean(i));
end
```

|      |       |       |       |       |       |       |      |      |      |
|------|-------|-------|-------|-------|-------|-------|------|------|------|
| 1000 | 68.03 | 68.11 | 68.10 | 18.75 | 18.75 | 18.75 | 3.05 | 3.09 | 2.92 |
| 1500 | 72.94 | 72.08 | 75.93 | 26.50 | 26.50 | 25.75 | 2.41 | 2.45 | 2.32 |
| 2000 | 72.35 | 71.48 | 75.47 | 25.50 | 25.50 | 25.00 | 2.14 | 2.17 | 2.04 |
| 2500 | 76.48 | 75.57 | 79.76 | 23.50 | 24.00 | 23.50 | 2.32 | 2.34 | 2.23 |
| 3000 | 77.66 | 76.71 | 80.99 | 24.25 | 24.25 | 23.50 | 2.33 | 2.35 | 2.24 |
| 3500 | 78.65 | 77.66 | 82.59 | 23.50 | 23.50 | 23.25 | 2.29 | 2.32 | 2.22 |
| 4000 | 80.54 | 79.54 | 84.09 | 23.50 | 23.50 | 23.25 | 2.32 | 2.35 | 2.24 |
| 4500 | 85.57 | 84.51 | 89.27 | 22.50 | 22.50 | 22.50 | 2.57 | 2.60 | 2.50 |
| 5000 | 87.02 | 85.94 | 90.76 | 21.50 | 21.75 | 21.50 | 2.72 | 2.74 | 2.66 |
| 5500 | 88.10 | 87.00 | 91.85 | 19.75 | 20.00 | 19.50 | 3.09 | 3.11 | 3.07 |
| 6000 | 85.50 | 84.47 | 89.12 | 19.00 | 19.50 | 18.75 | 3.12 | 3.13 | 3.11 |
| 6500 | 82.56 | 81.56 | 85.88 | 17.75 | 17.75 | 17.00 | 3.33 | 3.33 | 3.36 |

## Predicted Performance

```
fE85 = 'E85-Engine Data (incl. torque, BMEP, BSFC).csv';
```

```

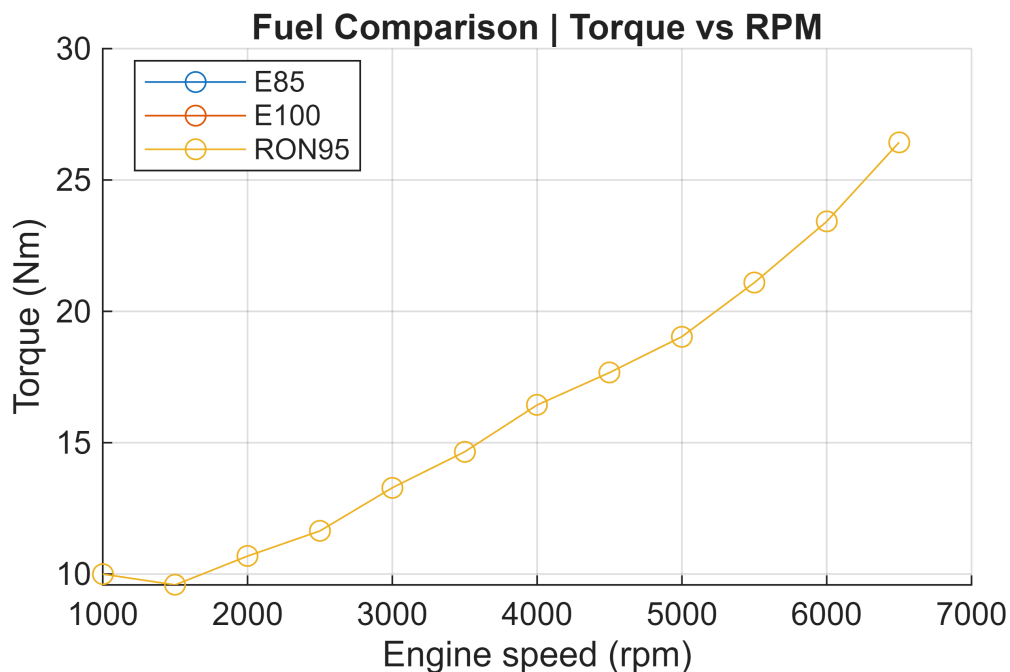
fE100 = 'E100-Engine data (incl. torque, BMEP, BSFC).csv';
fRON   = 'RON95-Engine data (incl. torque, BMEP, BSFC).csv';

E85    = read_perf_table_format(fE85);
E100   = read_perf_table_format(fE100);
RON    = read_perf_table_format(fRON);

rpm_common = intersect(intersect(E85.rpm, E100.rpm), RON.rpm);
E85       = keep_rpm_perf(E85, rpm_common);
E100      = keep_rpm_perf(E100, rpm_common);
RON       = keep_rpm_perf(RON, rpm_common);

% Torque
figure; hold on; grid on;
plot(rpm_common, E85.torque, '-o', 'DisplayName','E85');
plot(rpm_common, E100.torque, '-o', 'DisplayName','E100');
plot(rpm_common, RON.torque, '-o', 'DisplayName','RON95');
xlabel('Engine speed (rpm)');
ylabel('Torque (Nm)');
title('Fuel Comparison | Torque vs RPM');
legend('Location','best');

```

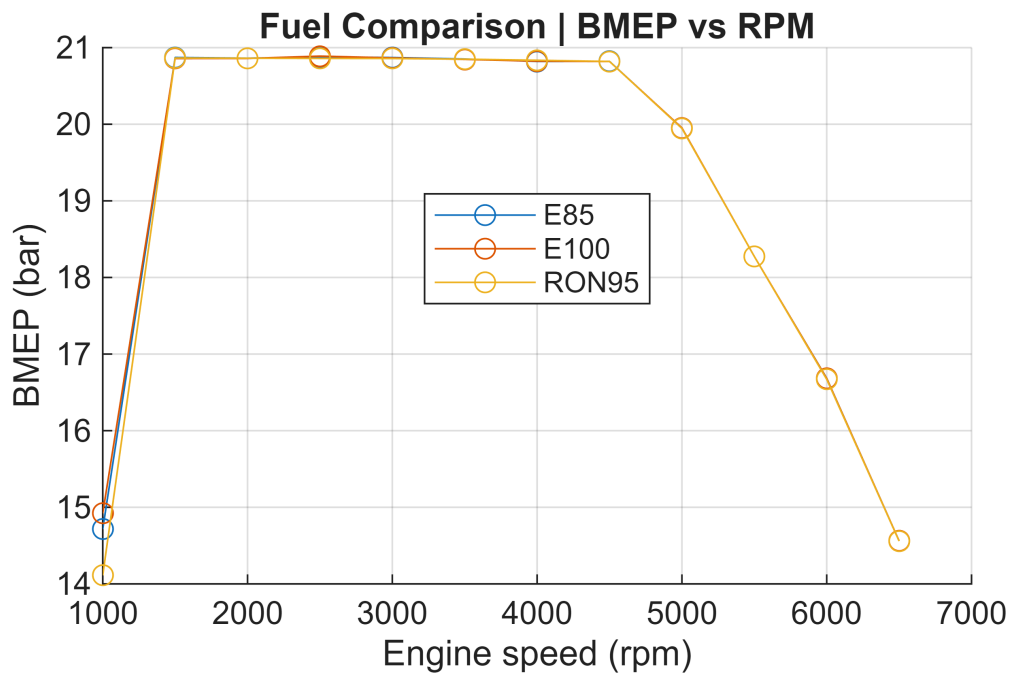


```

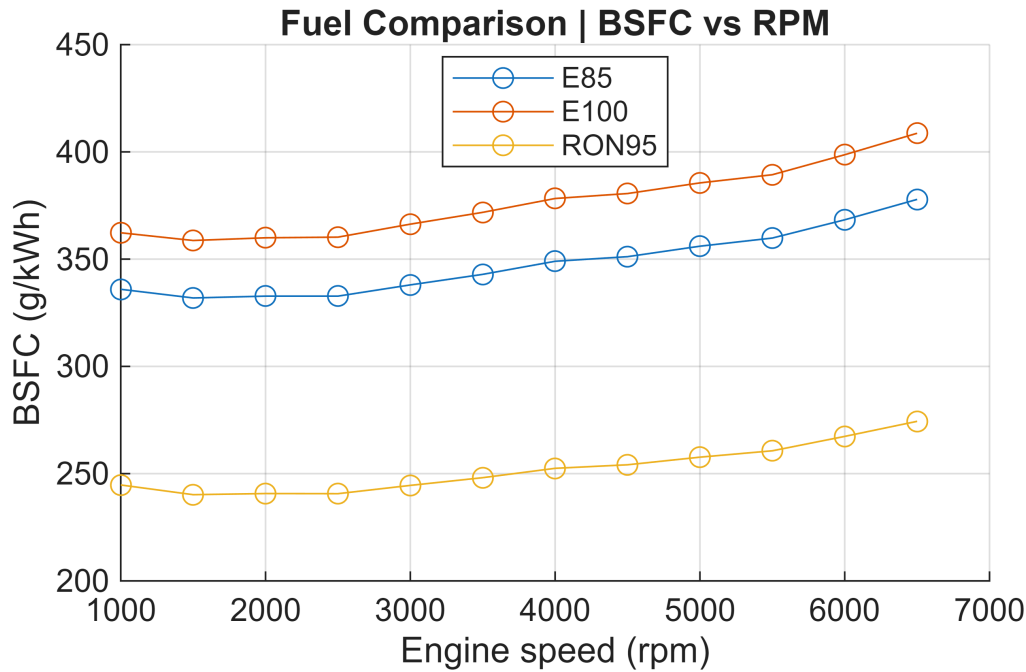
% BMEP
figure; hold on; grid on;
plot(rpm_common, E85.bmeep, '-o', 'DisplayName','E85');
plot(rpm_common, E100.bmeep, '-o', 'DisplayName','E100');
plot(rpm_common, RON.bmeep, '-o', 'DisplayName','RON95');
xlabel('Engine speed (rpm)');
ylabel('BMEP (bar)');
title('Fuel Comparison | BMEP vs RPM');

```

```
legend('Location','best');
```



```
% Plot: BSFC
figure; hold on; grid on;
plot(rpm_common, E85.bsfc, '-o', 'DisplayName','E85');
plot(rpm_common, E100.bsfc, '-o', 'DisplayName','E100');
plot(rpm_common, RON.bsfc, '-o', 'DisplayName','RON95');
xlabel('Engine speed (rpm)');
ylabel('BSFC (g/kWh)');
title('Fuel Comparison | BSFC vs RPM');
legend('Location','best');
```



```
% Print table
fprintf('\nPerformance comparison (common RPM points):\n');
```

Performance comparison (common RPM points):

```
fprintf('RPM | Torque(E85 E100 RON95) | BMEP(E85 E100 RON95) | BSFC(E85 E100
RON95)\n');
```

```
RPM | Torque(E85 E100 RON95) | BMEP(E85 E100 RON95) | BSFC(E85 E100 RON95)
```

```
for i = 1:numel(rpm_common)
    fprintf('%4d | %7.1f %7.1f %7.1f | %6.2f %6.2f %6.2f | %7.1f %7.1f %7.1f\n', ...
        rpm_common(i), ...
        E85.torque(i), E100.torque(i), RON.torque(i), ...
        E85.bmep(i), E100.bmep(i), RON.bmep(i), ...
        E85.bsfc(i), E100.bsfc(i), RON.bsfc(i));
end
```

|      |      |      |      |       |       |       |       |       |       |
|------|------|------|------|-------|-------|-------|-------|-------|-------|
| 1000 | 10.0 | 10.0 | 10.0 | 14.72 | 14.92 | 14.11 | 335.9 | 362.3 | 244.7 |
| 1500 | 9.6  | 9.6  | 9.6  | 20.87 | 20.86 | 20.86 | 331.9 | 358.7 | 240.2 |
| 2000 | 10.7 | 10.7 | 10.7 | 20.86 | 20.86 | 20.86 | 332.8 | 359.9 | 240.7 |
| 2500 | 11.6 | 11.6 | 11.6 | 20.88 | 20.89 | 20.86 | 332.8 | 360.2 | 240.7 |
| 3000 | 13.3 | 13.3 | 13.3 | 20.87 | 20.87 | 20.86 | 338.0 | 366.3 | 244.5 |
| 3500 | 14.7 | 14.7 | 14.7 | 20.85 | 20.85 | 20.85 | 342.9 | 371.8 | 248.1 |
| 4000 | 16.4 | 16.4 | 16.4 | 20.82 | 20.82 | 20.83 | 349.0 | 378.3 | 252.5 |
| 4500 | 17.7 | 17.7 | 17.7 | 20.82 | 20.82 | 20.82 | 351.2 | 380.6 | 254.1 |
| 5000 | 19.0 | 19.0 | 19.0 | 19.95 | 19.95 | 19.95 | 356.0 | 385.5 | 257.7 |
| 5500 | 21.1 | 21.1 | 21.1 | 18.27 | 18.27 | 18.27 | 359.8 | 389.3 | 260.6 |
| 6000 | 23.4 | 23.4 | 23.4 | 16.68 | 16.68 | 16.67 | 368.3 | 398.7 | 267.4 |
| 6500 | 26.4 | 26.4 | 26.4 | 14.56 | 14.56 | 14.56 | 377.9 | 408.7 | 274.4 |

```
function D = read_perf_table_format(filename)
```

```

C = readcell(filename);

rpm = to_num(C(1, 2:end));

names = lower(string(C(2:end, 1)));

idx_torque = find(contains(names, "torqsi") | (contains(names, "torq") &
contains(names, "n*m")), 1);

idx_bmep = find(contains(names, "bmepbar") | (contains(names, "bmep") &
contains(names, "bar")), 1);

idx_bsfc = find(contains(names, "bsfc") & (contains(names, "g/kw") |
contains(names, "kg/kw") | contains(names, "g/kwh") | contains(names, "kg/kwh")), 1);

if isempty(idx_torque) || isempty(idx_bmep) || isempty(idx_bsfc)
    error('Could not locate torque/BMEP/BSFC rows in %s. Check row labels.',
filename);
end

torque = to_num(C(idx_torque+1, 2:end));
bmep = to_num(C(idx_bmep+1, 2:end));
bsfc = to_num(C(idx_bsfc+1, 2:end));

label_bsfc = names(idx_bsfc);
if contains(label_bsfc, "kg/kw") || contains(label_bsfc, "kg/kwh")
    bsfc = bsfc * 1000;
end

valid = ~isnan(rpm) & ~isnan(torque) & ~isnan(bmep) & ~isnan(bsfc);
rpm = rpm(valid);
torque = torque(valid);
bmep = bmep(valid);
bsfc = bsfc(valid);

[rpm, order] = sort(rpm, 'ascend');
torque = torque(order);
bmep = bmep(order);
bsfc = bsfc(order);

D = struct('rpm', rpm, 'torque', torque, 'bmep', bmep, 'bsfc', bsfc,
'bsfc_label', label_bsfc);
end

function v = to_num(cellRow)
v = str2double(string(cellRow));
if all(isnan(v)) && ~isempty(cellRow)
    s = replace(string(cellRow), ",", ".");
    v = str2double(s);
end

```

```

end

function D = keep_rpm_perf(D, rpm_keep)
    [tf, loc] = ismember(rpm_keep, D.rpm);
    if ~all(tf)
        error('RPM alignment failed: some RPM points not found in one dataset.');
```

```

    end
    D.rpm      = D.rpm(loc);
    D.torque   = D.torque(loc);
    D.bmep     = D.bmep(loc);
    D.bsfc     = D.bsfc(loc);
end

```

Across all fuels, torque and BMEP trends are nearly identical, confirming that WAVE operating constraints dominate load output. Differences are therefore primarily combustion-related. Ethanol fuels exhibit slightly earlier combustion phasing and higher pressure rise rates at high RPM, while RON95 achieves higher peak pressures and lower BSFC. These trends align with expected effects of octane rating, laminar flame speed, and fuel heating value.

### **Task3 - Combustion Model Refinement (ALL FUELS @ 3500 rpm)**

```

target_rpm = 3500;
caseIdx = round((7000 - target_rpm)/500);

useSmoothing = true;
smoothWindow = 5;
gamma = 1.34;

% Engine geometry (from WAVE)
geom.bore_mm    = 82.0;
geom.stroke_mm  = 94.6;
geom.rod_mm     = 133.5;
geom.CR         = 11;

opts.AHRR_thr_frac = 0.05;
opts.pad_pts       = 10;

fuels = struct( ...
    'tag',  {"RON95", "E85", "E100"}, ...
    'file', {"RON95-Cyl1-Pressures.csv", "E85-Cyl1-Pressures.csv", "E100-Cyl1-Pressures.csv"} ...
);

SummaryAll = [];

for k = 1:numel(fuels)
    Results(k) = run_task3_one_fuel(fuels(k).tag, fuels(k).file, target_rpm,
    caseIdx, geom, gamma, useSmoothing, smoothWindow, opts);
end

```

```

SummaryAll = [SummaryAll; Results(k).Summary];
end

disp("=== Task 3 Combustion Parameter Estimates (ALL FUELS) ===");

```

```

=== Task 3 Combustion Parameter Estimates (ALL FUELS) ===

```

```

disp(SummaryAll);

```

| Fuel    | RPM  | CA05_deg | CA10_deg | CA50_deg | CA90_deg | BurnDur_CA90_CA10_deg | Wiebe_a | Wiebe_m |
|---------|------|----------|----------|----------|----------|-----------------------|---------|---------|
| "RON95" | 3500 | 12.186   | 19.92    | 42.66    | 62.642   | 42.722                | 1.9259  | 0.47488 |
| "E85"   | 3500 | 12.915   | 20.771   | 42.696   | 62.334   | 41.563                | 1.9215  | 0.45136 |
| "E100"  | 3500 | 13.166   | 21.029   | 42.677   | 62.217   | 41.187                | 1.9125  | 0.43714 |

```

save("Task3_AllFuels_" + target_rpm + "rpm.mat", "Results", "SummaryAll", "geom",
"gamma", "opts");

```

```

% Plots

```

```

wiebeColour = [0.6350 0.0780 0.1840];

```

```

fuelColours = struct( ...
    'E85', [0.0 0.4470 0.7410], ...
    'E100', [0.8500 0.3250 0.0980], ...
    'RON95', [0.9290 0.6940 0.1250] ...
);

```

```

% PER-FUEL PLOTS

```

```

for k = 1:numel(Results)

```

```

    fuel = Results(k).fuelTag;
    col = fuelColours.(fuel);

```

```

% Pressure trace

```

```

figure('Name', fuel + " - Pressure @ " + target_rpm + " rpm");
plot(Results(k).theta_deg, Results(k).P_bar, ...
    'LineWidth', 1.5, 'Color', col);
grid on;
xlabel('Crank angle (deg)');
ylabel('Cylinder pressure (bar)');
title("WAVE Cylinder Pressure - " + fuel + " @ " + target_rpm + " rpm");

```

```

% Apparent Heat Release Rate

```

```

figure('Name', fuel + " - AHRR @ " + target_rpm + " rpm");
plot(Results(k).theta_deg, Results(k).dQ_dth/1000, ...
    'LineWidth', 1.5, 'Color', col);
grid on;
xlabel('Crank angle (deg)');
ylabel('dQ/d\theta (kJ/rad)');
title("Apparent Heat Release Rate - " + fuel + " @ " + target_rpm + " rpm");

```

```

% MFB and Wiebe fit
figure('Name', fuel + " - MFB & Wiebe @ " + target_rpm + " rpm");

% Derived MFB
plot(Results(k).thb, Results(k).xb, ...
     'LineWidth', 1.5, 'Color', col);
hold on;

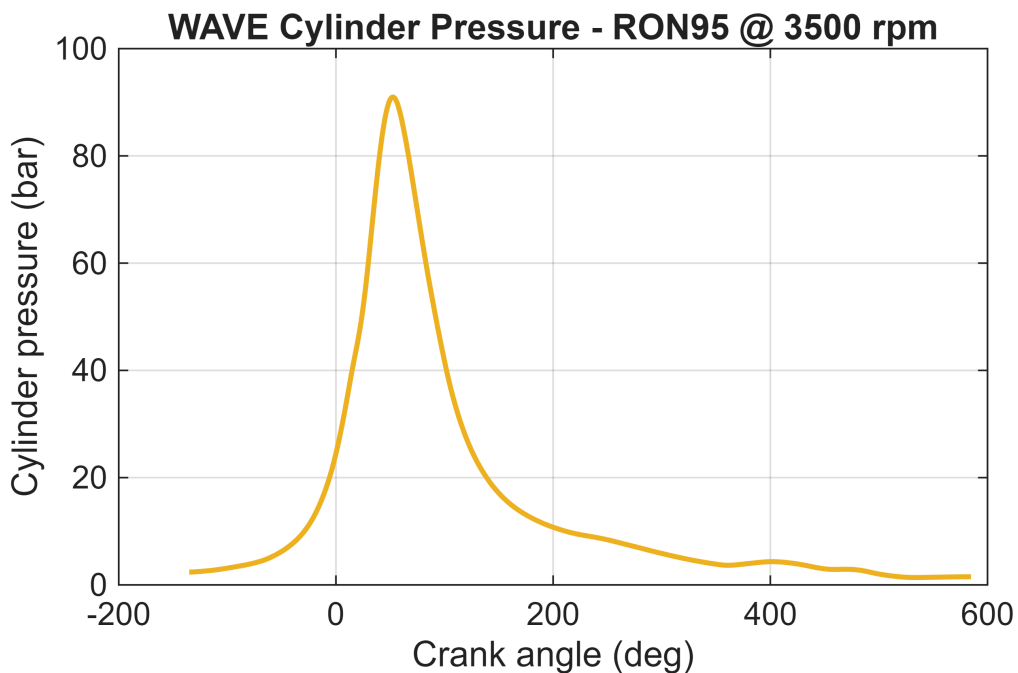
% Wiebe fit
plot(Results(k).thb, Results(k).xb_wiebe, ':', ...
     'LineWidth', 2.0, 'Color', wiebeColour);

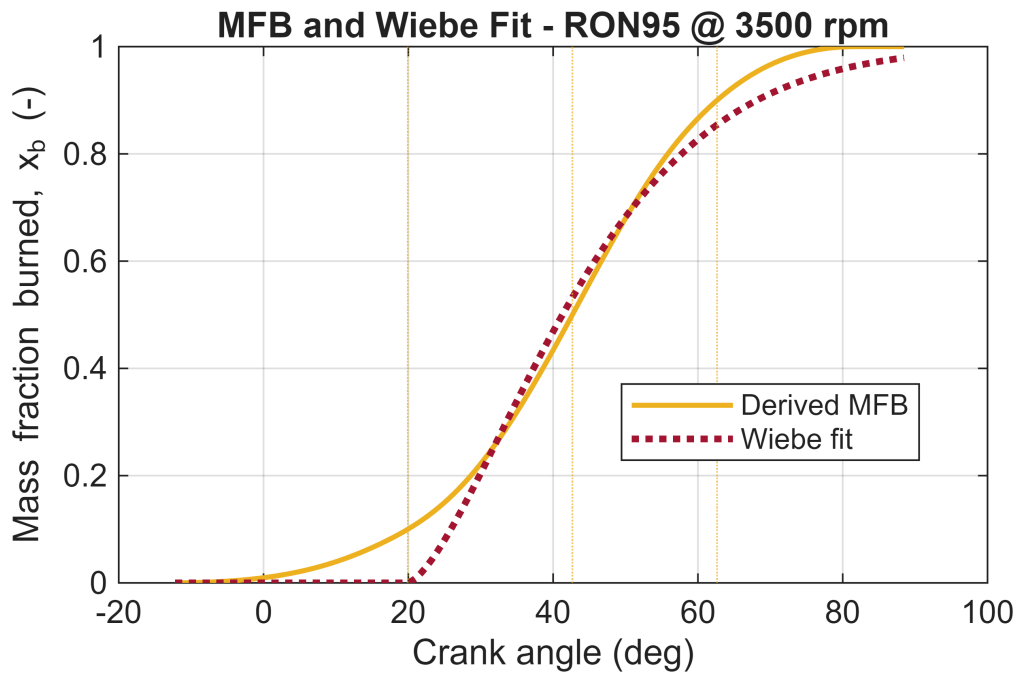
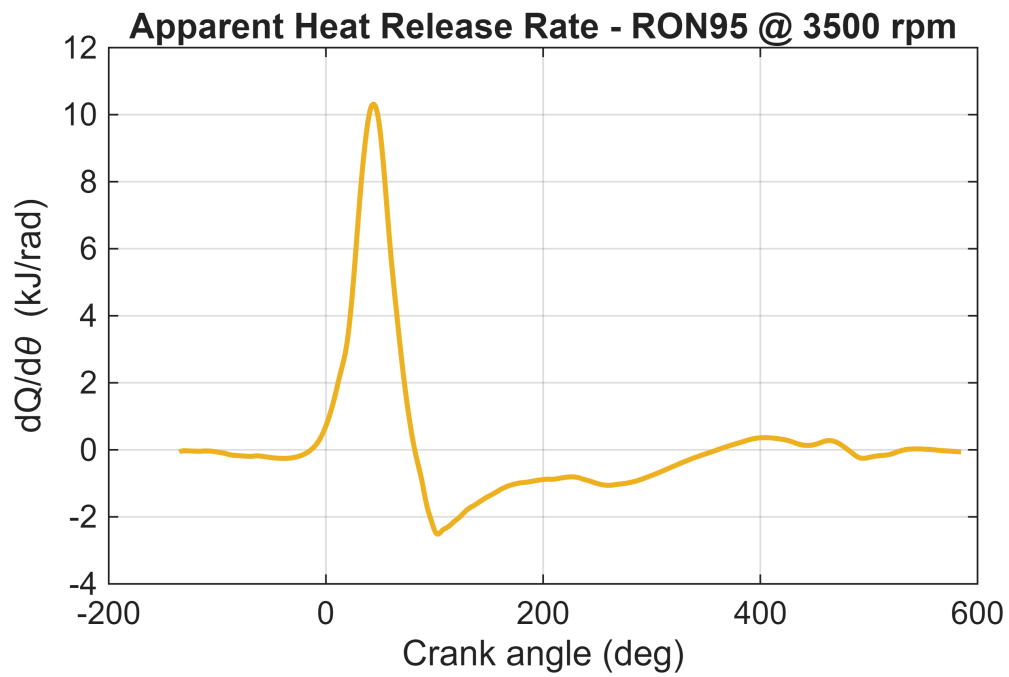
xline(Results(k).CA10, ':', 'Color', col);
xline(Results(k).CA50, ':', 'Color', col);
xline(Results(k).CA90, ':', 'Color', col);

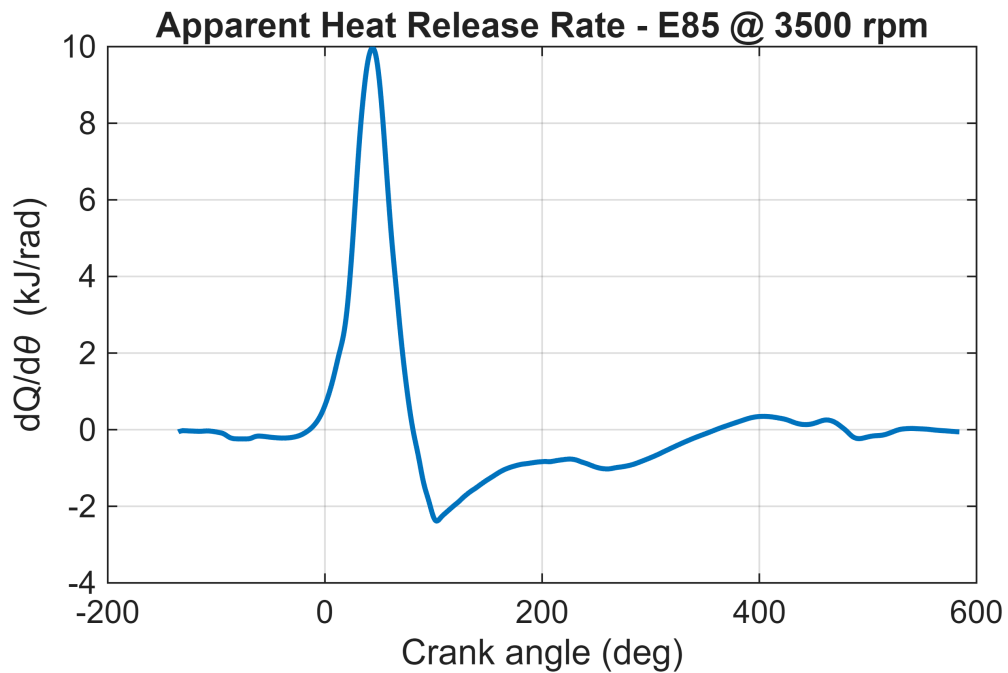
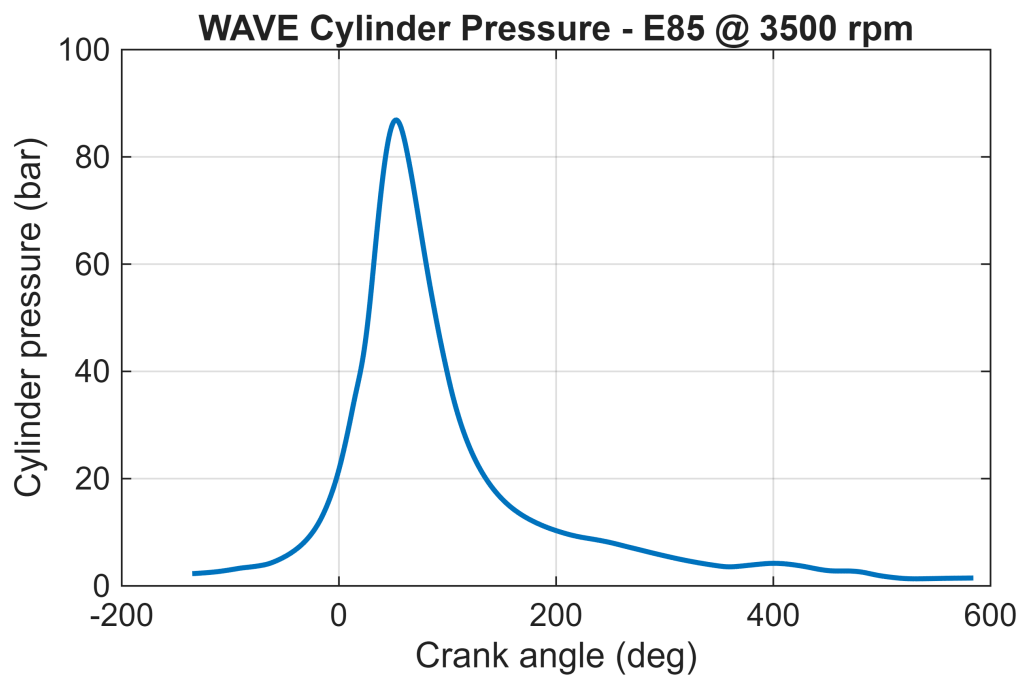
grid on;
xlabel('Crank angle (deg)');
ylabel('Mass fraction burned, x_b (-)');
legend('Derived MFB', 'Wiebe fit', 'Location', 'best');
title("MFB and Wiebe Fit - " + fuel + " @ " + target_rpm + " rpm");

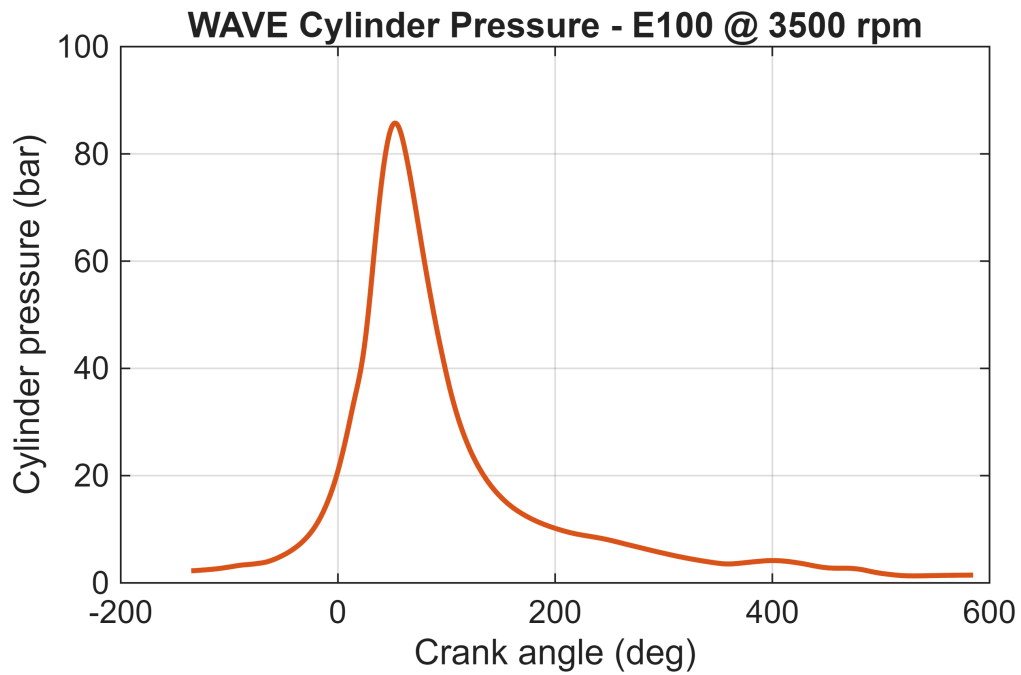
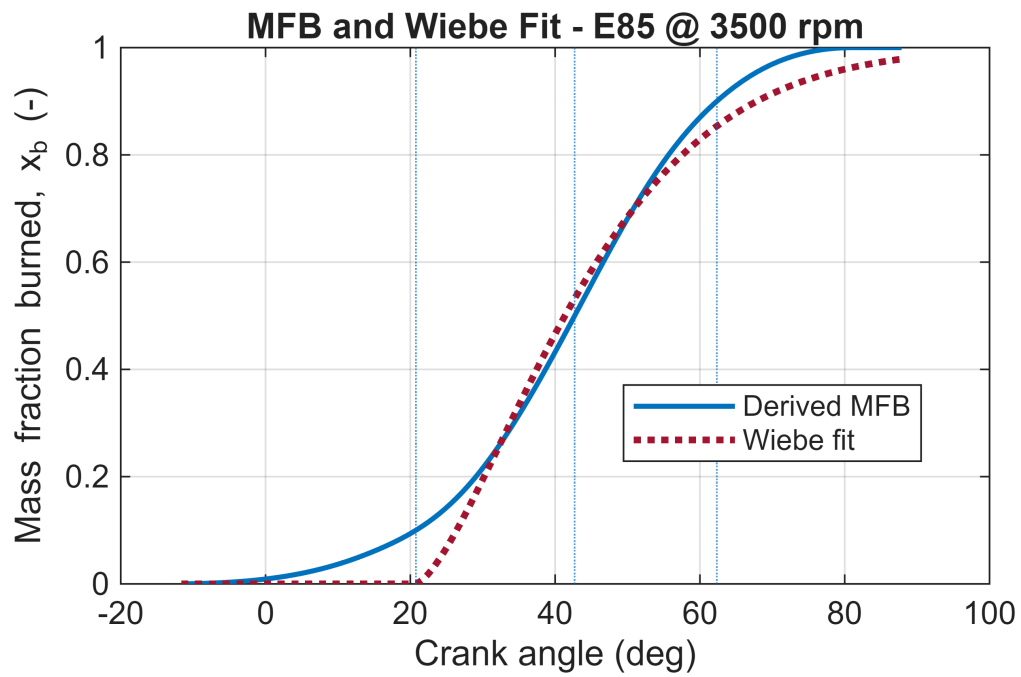
end

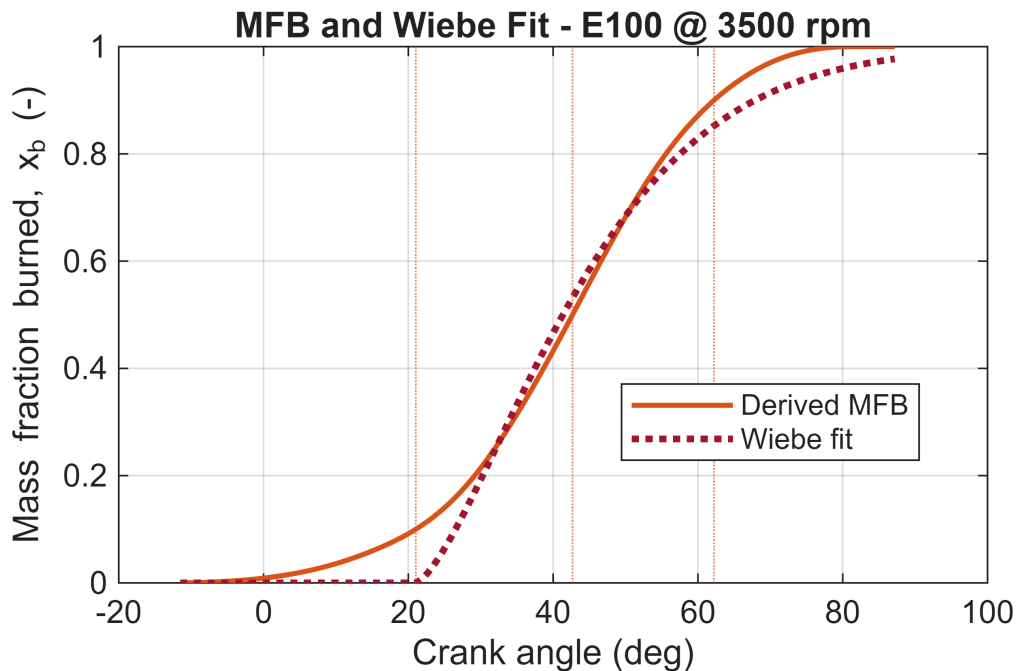
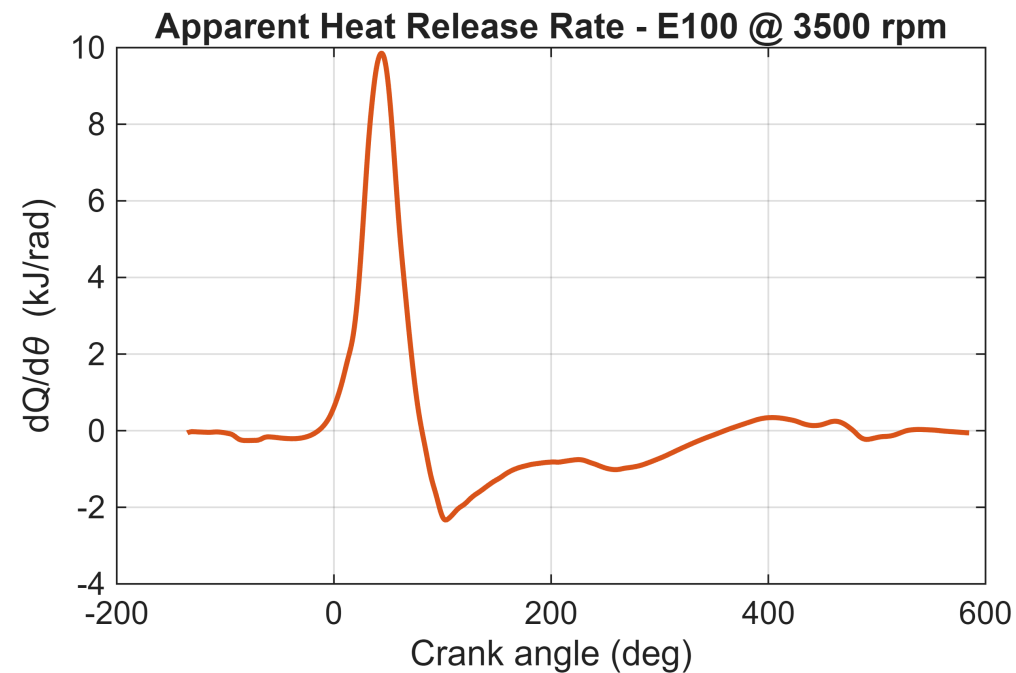
```











```
function out = run_task3_one_fuel(fuelTag, pressFile, target_rpm, caseIdx, geom,
gamma, useSmoothing, smoothWindow, opts)
```

```
    T = readtable(pressFile, 'PreserveVariableNames', true);
    theta_deg = T{:,1};
    if ~isnumeric(theta_deg)
        error("First column is not numeric in %. Expected crank angle in
degrees.", pressFile);
    end
```

```
    Pmat = T{:,2:end};
```

```

    if caseIdx < 1 || caseIdx > size(Pmat,2)
        error("caseIdx=%d out of range for %s (has %d columns).", caseIdx,
pressFile, size(Pmat,2));
    end

    P_raw = Pmat(:,caseIdx);
    P_raw(P_raw < 0.01) = 0.01;

    if useSmoothing
        P_raw = movmean(P_raw, smoothWindow);
    end

    Pmax_val = max(P_raw);
    if Pmax_val > 1e4
        P_Pa = P_raw;
        P_bar = P_raw / 1e5;
    elseif Pmax_val > 500
        P_Pa = P_raw * 1e3;
        P_bar = P_Pa / 1e5;
    else
        P_bar = P_raw;
        P_Pa = P_raw * 1e5;
    end

    bore_m = geom.bore_mm/1000;
    stroke_m = geom.stroke_mm/1000;
    rod_m = geom.rod_mm/1000;
    CR = geom.CR;

    Ap = pi*(bore_m^2)/4;
    Vd = Ap*stroke_m;
    Vc = Vd/(CR - 1);

    theta_rad = deg2rad(theta_deg);
    r = stroke_m/2; l = rod_m;

    x = r*(1 - cos(theta_rad)) + l - sqrt(max(l^2 - (r*sin(theta_rad)).^2, 0));
    V = Vc + Ap*x;

    dp_dth = nan(size(P_Pa));
    dV_dth = nan(size(V));

    for i = 2:numel(theta_rad)-1
        dtheta = theta_rad(i+1) - theta_rad(i-1);
        dp_dth(i) = (P_Pa(i+1) - P_Pa(i-1)) / dtheta;
        dV_dth(i) = (V(i+1) - V(i-1)) / dtheta;
    end

    if useSmoothing
        dp_dth = movmean(dp_dth, smoothWindow, 'omitnan');
    end

```

```

    dV_dth = movmean(dV_dth, smoothWindow, 'omitnan');
end

dQ_dth = (gamma/(gamma-1)) .* P_Pa .* dV_dth + (1/(gamma-1)) .* V .* dp_dth;

valid = ~isnan(dQ_dth) & ~isnan(P_Pa) & ~isnan(V);

Qcum = nan(size(dQ_dth));
Qcum(valid) = cumtrapz(theta_rad(valid), dQ_dth(valid));

dQ = dQ_dth; th = theta_deg;

[pk, iPk] = max(dQ(valid));
idx_valid = find(valid);
iPk_global = idx_valid(iPk);

if pk <= 0
    error("AHRR peak not positive for %. Check units/derivatives.", fuelTag);
end

thr = opts.AHRR_thr_frac * pk;

idx = find(valid & (dQ > thr));
if isempty(idx)
    error("No AHRR points exceed threshold for %. Reduce AHRR_thr_frac.",
fuelTag);
end

blockId = cumsum([true; diff(idx)~=1]);
blocks = splitapply(@(x){x}, idx, blockId);
hasPeak = cellfun(@(b) any(b==iPk_global), blocks);
burnIdx = blocks{find(hasPeak,1,'first')};

burnIdx = (min(burnIdx)-opts.pad_pts):(max(burnIdx)+opts.pad_pts);
burnIdx = burnIdx(burnIdx>=1 & burnIdx<=numel(th));

Qb = Qcum(burnIdx);
thb = th(burnIdx);

Qb_mono = cummax(Qb);
den = Qb_mono(end) - Qb_mono(1);
if den <= 0
    error("Cumulative heat did not increase for %. Check window.", fuelTag);
end

xb = (Qb_mono - Qb_mono(1)) / den;

[xb_u, ia] = unique(xb, 'stable');
thb_u = thb(ia);

```

```

CA_at = @(x) interp1(xb_u, thb_u, x, 'linear', 'extrap');

CA05 = CA_at(0.05);
CA10 = CA_at(0.10);
CA50 = CA_at(0.50);
CA90 = CA_at(0.90);

burnDuration = CA90 - CA10;

% Wiebe
theta0 = CA10;
dtheta = burnDuration;

fitMask = (thb >= CA10) & (thb <= CA90);
th_fit = thb(fitMask);
xb_fit = xb(fitMask);

wiebe = @(a,m,th) 1 - exp( -a .* ((max(th - theta0, 0)./dtheta)).^(m+1) );
obj = @(p) sum( (wiebe(exp(p(1)), exp(p(2)), th_fit) - xb_fit).^2 );

p0 = [log(5), log(2)];
pOpt = fminsearch(obj, p0);

aFit = exp(pOpt(1));
mFit = exp(pOpt(2));

xb_wiebe = wiebe(aFit, mFit, thb);

% Summary table
Summary = table( ...
    string(fuelTag), target_rpm, CA05, CA10, CA50, CA90, burnDuration, aFit,
mFit, ...
    'VariableNames',
{'Fuel', 'RPM', 'CA05_deg', 'CA10_deg', 'CA50_deg', 'CA90_deg', 'BurnDur_CA90_CA10_deg', 'W
iebe_a', 'Wiebe_m'} );

out = struct();
out.fuelTag = fuelTag;
out.pressFile = pressFile;
out.Summary = Summary;

out.theta_deg = theta_deg;
out.P_bar = P_bar;
out.P_Pa = P_Pa;

out.dQ_dth = dQ_dth;
out.Qcum = Qcum;

out.thb = thb;
out.xb = xb;

```

```

    out.xb_wiebe = xb_wiebe;

    out.CA05 = CA05; out.CA10 = CA10; out.CA50 = CA50; out.CA90 = CA90;
    out.burnDuration = burnDuration;
    out.Wiebe_a = aFit; out.Wiebe_m = mFit;
end

fuelColours = struct( ...
    'E85', [0.0 0.4470 0.7410], ...
    'E100', [0.8500 0.3250 0.0980], ...
    'RON95', [0.9290 0.6940 0.1250] ...
);

fuelOrder = ["RON95", "E85", "E100"];

Fuel = strings(numel(Results),1);
CA10 = nan(numel(Results),1);
CA50 = nan(numel(Results),1);
CA90 = nan(numel(Results),1);
BurnDur = nan(numel(Results),1);
Wiebe_a = nan(numel(Results),1);
Wiebe_m = nan(numel(Results),1);

for k = 1:numel(Results)
    Fuel(k) = string(Results(k).fuelTag);
    CA10(k) = Results(k).CA10;
    CA50(k) = Results(k).CA50;
    CA90(k) = Results(k).CA90;
    BurnDur(k) = Results(k).burnDuration;
    Wiebe_a(k) = Results(k).Wiebe_a;
    Wiebe_m(k) = Results(k).Wiebe_m;
end

Task3_Summary = table(Fuel, repmat(target_rpm, numel(Results),1), CA10, CA50, CA90,
    BurnDur, Wiebe_a, Wiebe_m, ...
    'VariableNames',
    {'Fuel', 'RPM', 'CA10_deg', 'CA50_deg', 'CA90_deg', 'BurnDur_CA90_CA10_deg', 'Wiebe_a', 'Wiebe_m'});

[~, idxSort] = ismember(Task3_Summary.Fuel, fuelOrder);
[~, sIdx] = sort(idxSort);
Task3_Summary = Task3_Summary(sIdx,:);

disp("=== Task 3 Summary Table (Fuel Comparison @ " + target_rpm + " rpm) ===");

=== Task 3 Summary Table (Fuel Comparison @ 3500 rpm) ===

disp(Task3_Summary);

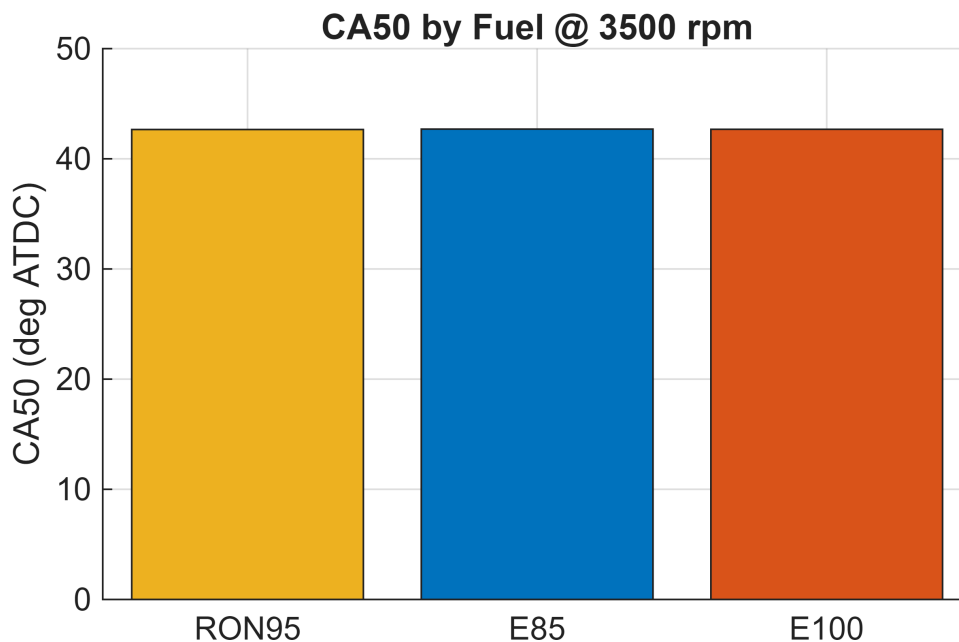
```

| Fuel    | RPM  | CA10_deg | CA50_deg | CA90_deg | BurnDur_CA90_CA10_deg | Wiebe_a | Wiebe_m |
|---------|------|----------|----------|----------|-----------------------|---------|---------|
| "RON95" | 3500 | 19.92    | 42.66    | 62.642   | 42.722                | 1.9259  | 0.47488 |
| "E85"   | 3500 | 20.771   | 42.696   | 62.334   | 41.563                | 1.9215  | 0.45136 |
| "E100"  | 3500 | 21.029   | 42.677   | 62.217   | 41.187                | 1.9125  | 0.43714 |

```
writetable(Task3_Summary, "Task3_Summary_" + target_rpm + "rpm.csv");

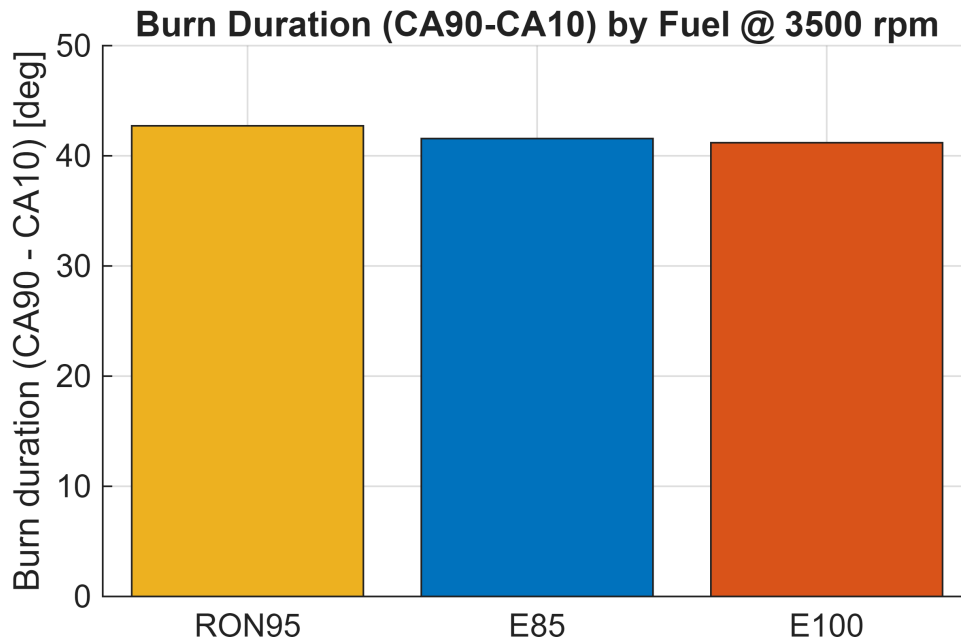
xNames = cellstr(Task3_Summary.Fuel);

% CA50 plot
figure('Name',"Task 3 - CA50 by Fuel @ " + target_rpm + " rpm");
hold on; grid on;
for i = 1:height(Task3_Summary)
    f = char(Task3_Summary.Fuel(i));
    bar(i, Task3_Summary.CA50_deg(i), 'FaceColor', fuelColours.(f));
end
xlim([0.5 height(Task3_Summary)+0.5]);
set(gca,'XTick',1:height(Task3_Summary),'XTickLabel',xNames);
ylabel('CA50 (deg ATDC)');
title("CA50 by Fuel @ " + target_rpm + " rpm");
```



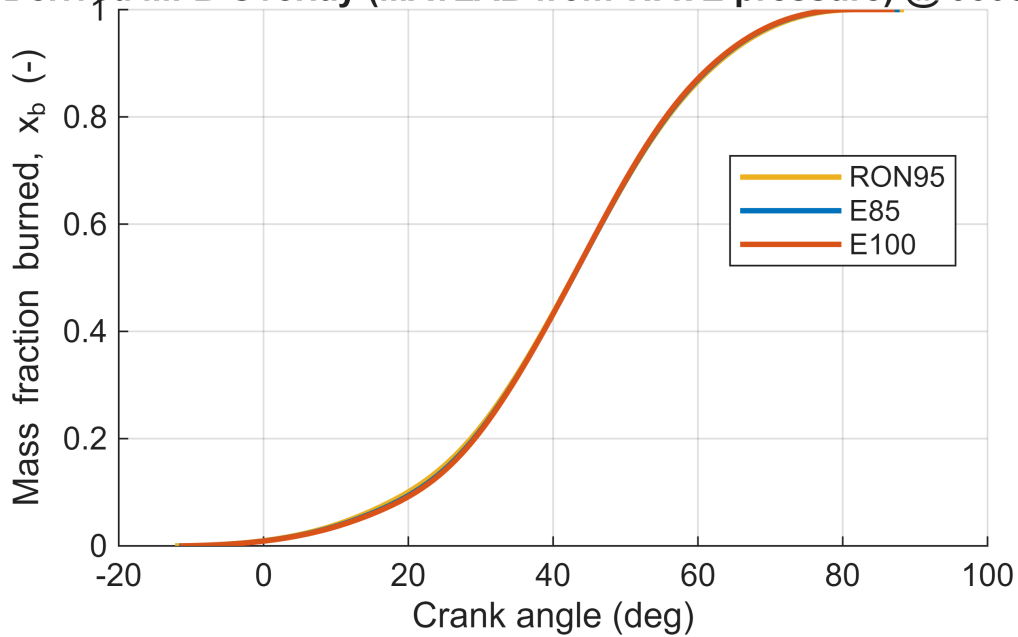
```
% Burn duration plot
figure('Name',"Task 3 - Burn Duration by Fuel @ " + target_rpm + " rpm");
hold on; grid on;
for i = 1:height(Task3_Summary)
    f = char(Task3_Summary.Fuel(i));
    bar(i, Task3_Summary.BurnDur_CA90_CA10_deg(i), 'FaceColor', fuelColours.(f));
end
```

```
xlim([0.5 height(Task3_Summary)+0.5]);
set(gca, 'XTick', 1:height(Task3_Summary), 'XTickLabel', xNames);
ylabel('Burn duration (CA90 - CA10) [deg]');
title('Burn Duration (CA90-CA10) by Fuel @ " + target_rpm + " rpm");
```

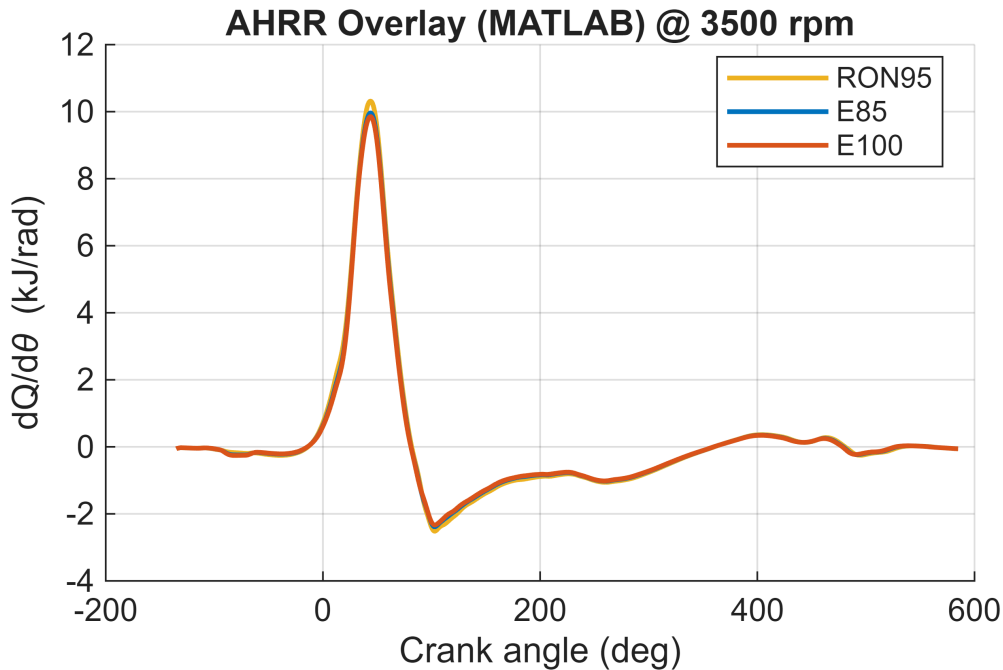


```
% MFB overlay
figure('Name', "Task 3 - Derived MFB Overlay @ " + target_rpm + " rpm");
hold on; grid on;
for k = 1:numel(Results)
    fuel = Results(k).fuelTag;
    col = fuelColours.(fuel);
    plot(Results(k).thb, Results(k).xb, 'LineWidth', 1.6, 'Color', col);
end
xlabel('Crank angle (deg)');
ylabel('Mass fraction burned, x_b (-)');
title("Derived MFB Overlay (MATLAB from WAVE pressure) @ " + target_rpm + " rpm");
legend({Results.fuelTag}, 'Location', 'best');
```

### Derived MFB Overlay (MATLAB from WAVE pressure) @ 3500 rpm



```
% AHRR overlay
figure('Name',"Task 3 - AHRR Overlay @ " + target_rpm + " rpm");
hold on; grid on;
for k = 1:numel(Results)
    fuel = Results(k).fuelTag;
    col = fuelColours.(fuel);
    plot(Results(k).theta_deg, Results(k).dQ_dth/1000, 'LineWidth', 1.4, 'Color',
col); % kJ/rad
end
xlabel('Crank angle (deg)');
ylabel('dQ/d\theta (kJ/rad)');
title("AHRR Overlay (MATLAB) @ " + target_rpm + " rpm");
legend({Results.fuelTag}, 'Location','best');
```



### Single-Fuel Combustion Behaviour

At 3500 rpm, all fuels show a well-defined apparent heat release peak shortly after TDC, followed by a characteristic negative phase associated with expansion. Derived mass fraction burned curves are smooth and monotonic, confirming numerical stability of the AHRR integration. CA50 occurs at approximately 42.7° ATDC for all fuels, indicating that combustion phasing is effectively normalised by the WAVE model at this operating point.

### Burn Duration and Wiebe Parameters

#### Burn Duration and Wiebe Parameters

Burn duration (CA90–CA10) decreases slightly with increasing ethanol content, from approximately 42.7° for RON95 to 41.2° for E100. This indicates a marginally faster overall burn for ethanol fuels. The fitted Wiebe parameter  $m$  decreases from RON95 to E100, reflecting a steeper initial burn rate, while the Wiebe efficiency parameter  $a$  remains similar across fuels. The Wiebe fits closely match the derived MFB curves, validating the single-zone combustion representation.

### Fuel Comparison at 3500 rpm

Despite differences in fuel properties, CA50 is effectively identical across fuels, confirming that combustion timing in WAVE is primarily imposed rather than fuel-driven. Differences therefore manifest mainly in burn shape and duration rather than phasing. Ethanol fuels show slightly shorter burn durations and more progressive heat release, consistent with higher laminar flame speeds and improved knock resistance.

### Task 3 Summary

#### Combustion Model Refinement - Summary

The combustion refinement demonstrates that MATLAB-derived AHRR and MFB analysis can reliably reproduce combustion characteristics from WAVE pressure data. While fuel chemistry influences burn rate and duration, overall combustion phasing remains constrained by the simulation setup. The Wiebe parameters provide a compact and physically meaningful representation of these differences.

## Overall Summary:

This project investigated spark-ignition engine combustion behaviour using a combination of WAVE simulation outputs and MATLAB-based post-processing, with a specific focus on pressure-based combustion analysis and fuel effects. In Task 1, in-cylinder pressure data from three engines with different geometries were analysed to examine how displacement, compression ratio, and operating speed influence pressure development, IMEP, and indicated work. Derived pressure–volume diagrams and polytropic indices highlighted clear differences in compression and expansion behaviour between engines, demonstrating the sensitivity of combustion characteristics to engine architecture.

Task 2 focused on fuel-specific combustion behaviour using WAVE results for RON95, E85, and E100 across a range of engine speeds. Peak cylinder pressure, crank angle of peak pressure, and maximum pressure rise rate were extracted and compared. Ethanol-based fuels showed later combustion phasing and slightly reduced pressure rise rates at lower speeds, consistent with higher knock resistance and slower flame development, while RON95 exhibited earlier peak pressure and higher mid-range pressure levels. Performance metrics including torque, BMEP, and BSFC were used to contextualise the combustion results and assess how fuel properties influence overall engine output.

In Task 3, a single-zone combustion model was implemented in MATLAB to derive apparent heat release rate, mass fraction burned, and Wiebe parameters from WAVE pressure traces at 3500 rpm. The close agreement in CA50 and burn duration across fuels, alongside small variations in Wiebe shape parameters, confirmed that while fuel chemistry influences early burn development, overall combustion duration remained comparable under matched operating conditions. This analysis demonstrates a clear understanding of pressure-based combustion diagnostics and the interpretation of simplified combustion models.