

Combustion and Emission Study: Modelling Fuels and Emissions.

Contemporary Powertrain Design (ACHA6006)

Jay Wexler

2305581

Lecturer: Dr. Owen Williams

University of Wales Trinity Saint David

School of Engineering

Motorcycle Engineering BEng

04/01/2026

Word Count: 2639

Word count note:

Titles, tables of contents, figures, and tables, MATLAB code, outputs, tables, figure captions, and axis labels are excluded.

Contents

1	Task 1 – Pressure-Based Analysis.....	4
1.1	Script Overview	4
1.2	Data Sources and Scope	4
1.3	Aim and Scope of Analysis	4
1.3.1	Aim	4
1.3.2	Scope	4
1.4	Engine Data Description and Assumptions	5
1.5	Import and Pre-Processing of Cylinder Pressure Data	5
1.6	Data quality checks	8
1.7	Identification of Compression and Combustion Events	10
1.7.1	Peak Cylinder Pressure and Combustion Phasing Indicators	10
1.7.2	Interpretation of Compression and Combustion Indicators	13
1.8	IMEP and Cycle Efficiency	14
1.8.1	Cylinder Volume and Pressure–Volume Relationship	14
1.8.2	Indicated Work and IMEP Calculations	17
1.8.3	Combined Results	18
1.9	Comparative Analysis and Indicated Cycle Efficiency	20
1.10	Engineering Discussion and Key Observations	20
1.10.1	Key observations	20
1.10.2	Limitations of the Analysis	20
1.10.3	Relevance to Subsequent Emissions Modelling	20
2	Task 2 – WAVE Simulation Report	21
2.1	Section Introduction	21
2.2	Fuel Properties and Model Setup	21
2.2.1	Fuel Properties	21
2.2.2	Summary of WAVE Model Configuration	22
2.3	Simulation Results	23
2.3.1	Mean Engine Torque vs Engine Speed	23
2.3.2	Brake Mean Effective Pressure (BMEP)	24
2.3.3	Brake Specific Fuel Consumption (BSFC)	25
2.3.4	In-Cylinder Pressure vs Crank Angle	26
2.4	Comparative Analysis of Fuel Effects	27
2.5	Model Assumptions and Limitations	27
2.6	Conclusions	27
3	Task 3 – Wiebe Combustion Modelling	29
3.1	Section Introduction and Aim	29
3.2	Fuels Analysed and Engine Data	29
3.3	Combustion Modelling Methodology	29
3.4	Implementation of Combustion Model	30
3.4.1	Data Import and Pre-processing	30
3.4.2	Apparent Heat Release Analysis	34
3.5	Wiebe Parameter Estimation	41
3.5.1	Initial Parameter Selection	41
3.5.2	Curve Fitting and Calibration	42
3.6	Pressure Reconstruction and Validation vs WAVE	46
3.7	Sensitivity Analysis of Wiebe Parameters	52
3.8	Discussion: Model Accuracy and Limitations	56
3.9	Conclusions and Recommendations	57

4	References	58
5	Appendix	58

Figure 1: Variation of mean engine torque with engine speed for octane, propane, and methanol under constant equivalence ratio conditions.	23
---	----

Figure 2: Variation of brake mean effective pressure (BMEP) with engine speed for octane, propane, and methanol under identical operating conditions.	24
--	----

Figure 3: Variation of brake specific fuel consumption (BSFC) with engine speed for octane, propane, and methanol, calculated from cycle-averaged fuel mass flow and indicated power.	25
--	----

Figure 4: In-cylinder pressure traces for octane, propane, and methanol at [chosen speed, e.g. 7000 rpm] for the front cylinder, shown as a function of crank angle.	26
---	----

Table 1: Key thermochemical properties of the fuels considered	21
--	----

1 Task 1 – Pressure-Based Analysis

1.1 Script Overview

This Live Script contains the MATLAB analysis for Task 1 and Task 3 of the ACHA6006 assessment. Task 1 applies pressure-based thermodynamic analysis to compare combustion behaviour across different engine architectures, while Task 3 uses the derived metrics to inform and validate a Wiebe-based combustion model against Ricardo WAVE outputs. The analysis is pressure-based and employs simplified, single-zone assumptions; Task 2 is addressed separately in the accompanying report.

Plots shown in Tasks 1 and 3 are direct MATLAB outputs included for transparency and illustration of the analysis workflow. Unless otherwise stated, crank angle is expressed in degrees crank angle (degCA).

1.2 Data Sources and Scope

- Experimental cylinder pressure data
- Lecture material and combustion literature.
- Ricardo WAVE engine simulation model
- Engine datasets analysed: **Engine A, B, C** (*CLAAS, Husqvarna 250, McLaren 570S*)

```
clc, clear; close all;  
dataDir = fullfile(pwd, 'data');
```

1.3 Aim and Scope of Analysis

1.3.1 Aim

- Analyse in-cylinder pressure data to characterise combustion behaviour.
- Extract pressure-based indicators such as **peak pressure**, **pressure rise rate**, and **IMEP**.
- Compare combustion behaviour across different engine architectures.

1.3.2 Scope

- Zero-dimensional thermodynamic interpretation of pressure data.
- Analysis based on **mean-cycle behaviour** under steady conditions.
- No explicit modelling of flame propagation, chemical kinetics, or emissions formation.

1.4 Engine Data Description and Assumptions

Engine configurations analysed

- **CLAAS**: four-stroke turbocharged diesel
- **McLaren 570S**: four-stroke twin-turbo spark-ignition
- **Husqvarna 250**: naturally aspirated two-stroke.

Key assumptions

- Pressure data are representative and physically plausible.
- Engine geometry and operating conditions remain constant for each dataset.
- Gas behaviour approximated as ideal where required.
- Heat transfer, blow-by, scavenging losses, and cycle-to-cycle variability neglected.

1.5 Import and Pre-Processing of Cylinder Pressure Data

- Cylinder pressure data imported from CSV files.
- Crank-angle conventions differ due to engine architecture:
- Four-stroke engines span 720° CA.
- Two-stroke engine spans 360° CA.
- Data cleaned by:
- removing non-finite values,
- sorting by crank angle,
- removing duplicate crank-angle points.
- Pressure traces aligned to a common crank-angle reference window for comparison.

```
% Read CSV files (numeric only)
A_CLAAS = readmatrix(fullfile(dataDir, 'CLAAS_pressure.csv'));
A_Husq  = readmatrix(fullfile(dataDir, 'Husqvarna250_pressure.csv'));
A_McL   = readmatrix(fullfile(dataDir, 'McLaren570S_pressure.csv'));

% Extract first two columns: [CrankAngle_deg, Pressure_bar]
ca_CLAAS = A_CLAAS(:,1);    p_CLAAS = A_CLAAS(:,2);
ca_Husq  = A_Husq(:,1);    p_Husq  = A_Husq(:,2);
ca_McL   = A_McL(:,1);     p_McL   = A_McL(:,2);

bad = (p_McL == 0);
ca_McL(bad) = [];
p_McL(bad)  = [];
```

```

% Remove NaNs / Infs, sort by crank angle, remove duplicates
[ca_CLAAS, p_CLAAS] = cleanTrace(ca_CLAAS, p_CLAAS);
[ca_Husq, p_Husq ] = cleanTrace(ca_Husq, p_Husq );
[ca_McL, p_McL ] = cleanTrace(ca_McL, p_McL );

% Align McLaren crank-angle convention
% Convert 0-720 -> -360-360
ca_McL = ca_McL - 360;

% Recentre so peak pressure occurs near 0 deg (TDC reference)
[~, iPk] = max(p_McL);
ca_McL = ca_McL - ca_McL(iPk);

% Wrap back into -360..360 and clean again (wrap can create duplicates)
ca_McL = mod(ca_McL + 360, 720) - 360;
[ca_McL, p_McL] = cleanTrace(ca_McL, p_McL);

% Trim all engines to a common crank-angle window for comparison
win = [-400, 400];

[ca_CLAAS, p_CLAAS] = trimWindow(ca_CLAAS, p_CLAAS, win);
[ca_Husq, p_Husq ] = trimWindow(ca_Husq, p_Husq, win);
[ca_McL, p_McL ] = trimWindow(ca_McL, p_McL, win);

% Parameter bounds: number of points retained
fprintf('Points after trimming: CLAAS=%d | Husq=%d | McLaren=%d\n', ...
    numel(ca_CLAAS), numel(ca_Husq), numel(ca_McL));

```

```

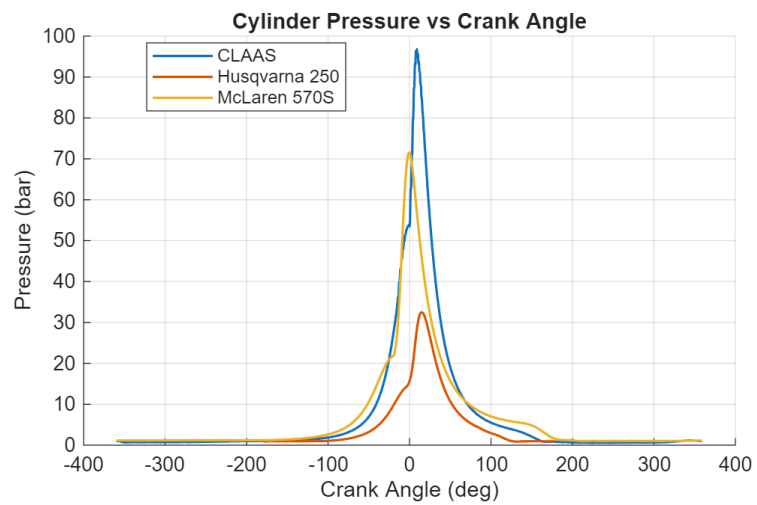
Points after trimming: CLAAS=880 | Husq=360 | McLaren=719

```

```

% Plot: pressure vs crank angle (all engines)
figure('Name','Cylinder Pressure vs Crank Angle'); hold on;
plot(ca_CLAAS, p_CLAAS, 'LineWidth', 1.2);
plot(ca_Husq, p_Husq, 'LineWidth', 1.2);
plot(ca_McL, p_McL, 'LineWidth', 1.2);
xlabel('Crank Angle (deg)');
ylabel('Pressure (bar)');
title('Cylinder Pressure vs Crank Angle');
legend('CLAAS','Husqvarna 250','McLaren 570S','Location','best');
grid on;

```



1.6 Data quality checks

- Crank-angle coverage checked to confirm a continuous engine cycle.
- Mean crank-angle resolution $\sim 1^\circ$, sufficient for numerical differentiation.
- Pressure levels verified for physical plausibility:
 - near-atmospheric values around BDC,
 - peak pressures consistent with engine type.
- Cleaning steps ensure stable calculation of pressure rise rate.

```
% Crank-angle range and resolution checks
fprintf('Crank-angle data quality summary:\n');
fprintf('CLAAS:\n');
fprintf('  CA range = %.1f to %.1f deg\n', min(ca_CLAAS), max(ca_CLAAS));
fprintf('  Mean CA step = %.3f deg\n\n', mean(diff(ca_CLAAS)));

fprintf('Husqvarna 250:\n');
fprintf('  CA range = %.1f to %.1f deg\n', min(ca_Husq), max(ca_Husq));
fprintf('  Mean CA step = %.3f deg\n\n', mean(diff(ca_Husq)));

fprintf('McLaren 570S:\n');
fprintf('  CA range = %.1f to %.1f deg\n', min(ca_McL), max(ca_McL));
fprintf('  Mean CA step = %.3f deg\n\n', mean(diff(ca_McL)));
```

Crank-angle data quality summary:

CLAAS:

CA range = -360.0 to 359.0 deg

Mean CA step = 0.818 deg

Husqvarna 250:

CA range = -180.0 to 179.0 deg

Mean CA step = 1.000 deg

McLaren 570S:

CA range = -360.0 to 359.0 deg

Mean CA step = 1.001 deg

```
% Basic pressure plausibility checks (visual + numeric)
fprintf('Pressure plausibility checks (bar):\n');
fprintf('  CLAAS      : min = %.4f, max = %.2f\n', min(p_CLAAS), max(p_CLAAS));
fprintf('  Husqvarna   : min = %.4f, max = %.2f\n', min(p_Husq), max(p_Husq));
fprintf('  McLaren     : min = %.4f, max = %.2f\n\n', min(p_McL), max(p_McL));
```

```
Pressure plausibility checks (bar):
CLAAS      : min = 0.5900, max = 96.75
Husqvarna  : min = 0.8760, max = 32.51
McLaren    : min = 1.0045, max = 71.52
```

A single zero-pressure sample, identified as a measurement error, was removed prior to thermodynamic analysis to prevent contamination of differentiation and p–V integration.

1.7 Identification of Compression and Combustion Events

1.7.1 Peak Cylinder Pressure and Combustion Phasing Indicators

```
[PkP_CLAAS, idx] = max(p_CLAAS);
PkCA_CLAAS = ca_CLAAS(idx);

[PkP_Husq, idx] = max(p_Husq);
PkCA_Husq = ca_Husq(idx);

[PkP_McL, idx] = max(p_McL);
PkCA_McL = ca_McL(idx);

% Pressure rise rate (dP/dCA)

% Light smoothing prior to differentiation to reduce numerical noise
smoothSpan = 21;

p_CLAAS_f = movmean(p_CLAAS, smoothSpan);
p_Husq_f  = movmean(p_Husq,  smoothSpan);
p_McL_f   = movmean(p_McL,   smoothSpan);

% Numerical differentiation (bar/deg)
dPdCA_CLAAS = gradient(p_CLAAS_f) ./ gradient(ca_CLAAS);
dPdCA_Husq  = gradient(p_Husq_f)  ./ gradient(ca_Husq);
dPdCA_McL   = gradient(p_McL_f)   ./ gradient(ca_McL);

% Maximum pressure rise rate and crank-angle location
[MaxdPdCA_CLAAS, idx] = max(dPdCA_CLAAS);
MaxdPdCA_CA_CLAAS = ca_CLAAS(idx);

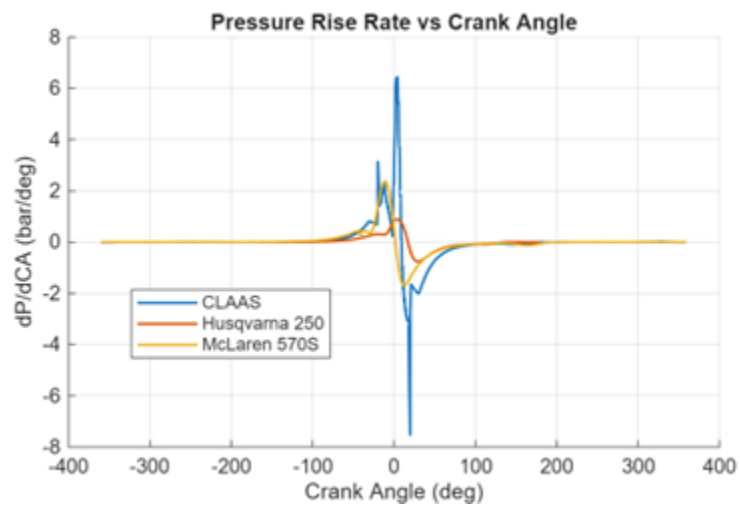
[MaxdPdCA_Husq, idx] = max(dPdCA_Husq);
MaxdPdCA_CA_Husq = ca_Husq(idx);

[MaxdPdCA_McL, idx] = max(dPdCA_McL);
MaxdPdCA_CA_McL = ca_McL(idx);
```

```

% Plot: pressure rise rate vs crank angle
figure('Name','Pressure Rise Rate'); hold on;
plot(ca_CLAAS, dPdCA_CLAAS, 'LineWidth', 1.2);
plot(ca_Husq, dPdCA_Husq, 'LineWidth', 1.2);
plot(ca_McL, dPdCA_McL, 'LineWidth', 1.2);
xlabel('Crank Angle (deg)');
ylabel('dP/dCA (bar/deg)');
title('Pressure Rise Rate vs Crank Angle');
legend('CLAAS','Husqvarna 250','McLaren 570S','Location','best');
grid on;

```



```
% Summary table of combustion indicators
```

```
Engine = ["CLAAS"; "Husqvarna 250"; "McLaren 570S"];
```

```
PeakPressure_bar = [PkP_CLAAS; PkP_Husq; PkP_McL];
```

```
PeakPressureCA_deg = [PkCA_CLAAS; PkCA_Husq; PkCA_McL];
```

```
MaxdPdCA_barPerDeg = [MaxdPdCA_CLAAS; MaxdPdCA_Husq; MaxdPdCA_McL];
```

```
MaxdPdCA_CA_deg = [MaxdPdCA_CA_CLAAS; MaxdPdCA_CA_Husq; MaxdPdCA_CA_McL];
```

```
Section4_Results = table( ...  
    Engine, ...  
    PeakPressure_bar, ...  
    PeakPressureCA_deg, ...  
    MaxdPdCA_barPerDeg, ...  
    MaxdPdCA_CA_deg);
```

```
disp(Section4_Results);
```

Engine	PeakPressure_bar	PeakPressureCA_deg	MaxdPdCA_barPerDeg	MaxdPdCA_CA_deg
"CLAAS"	96.75	9	6.4393	4.4
"Husqvarna 250"	32.511	15	0.89079	3
"McLaren 570S"	71.519	0	2.3761	-11

1.7.2 Interpretation of Compression and Combustion Indicators

- Crank angle zero treated as **TDC reference** after alignment.
- Compression identified from gradual pressure increase prior to combustion.
- Combustion onset inferred from sustained increase in $\frac{dP}{dCA}$:
- used as a **relative indicator**, not absolute SOC.
- Peak cylinder pressure and its crank-angle location extracted directly.

Pressure rise rate analysis

- Pressure data lightly smoothed prior to differentiation to reduce numerical noise.
- Maximum $\frac{dP}{dCA}$ used to characterise **combustion aggressiveness**.
- Crank-angle location of maximum $\frac{dP}{dCA}$ used as a comparative phasing indicator.

1.8 IMEP and Cycle Efficiency

1.8.1 Cylinder Volume and Pressure–Volume Relationship

```
% Convert pressure from bar to Pa for thermodynamic calculations
p_CLAAS_Pa = p_CLAAS * 1e5;
p_Husq_Pa  = p_Husq  * 1e5;
p_McL_Pa   = p_McL   * 1e5;

%% Engine geometry (SI units)
gCLAAS.Bore   = 0.104;    % m
gCLAAS.Stroke = 0.132;    % m
gCLAAS.Conrod = 0.195;    % m
gCLAAS.CR     = 17.5;     % -

gHusq.Bore    = 0.06634;  % m
gHusq.Stroke  = 0.072;    % m
gHusq.Conrod  = 0.125;    % m
gHusq.CR      = 11.6;     % -

gMcL.Bore     = 0.093;    % m
gMcL.Stroke   = 0.069;    % m
gMcL.Conrod   = 0.110;    % m
gMcL.CR       = 10.5;     % -

%% Crank angle to radians
th_CLAAS = deg2rad(ca_CLAAS);
th_Husq  = deg2rad(ca_Husq);
th_McL   = deg2rad(ca_McL);

%% Cylinder volume traces
V_CLAAS = cylVolume(th_CLAAS, gCLAAS);
V_Husq  = cylVolume(th_Husq, gHusq);
V_McL   = cylVolume(th_McL, gMcL);

%% Swept and clearance volumes
[Vd_CLAAS, Vc_CLAAS] = sweptClearance(gCLAAS);
[Vd_Husq,  Vc_Husq ] = sweptClearance(gHusq);
[Vd_McL,   Vc_McL  ] = sweptClearance(gMcL);

%% Volume checks
fprintf('Volume checks:\n');
fprintf('CLAAS: Vmin=%.3e, Vmax=%.3e, Vd=%.3e\n', ...
    min(V_CLAAS), max(V_CLAAS), Vd_CLAAS);
fprintf('Husq : Vmin=%.3e, Vmax=%.3e, Vd=%.3e\n', ...
    min(V_Husq), max(V_Husq), Vd_Husq);
```

```
fprintf('McL : Vmin=%.3e, Vmax=%.3e, Vd=%.3e\n\n', ...  
       min(V_McL), max(V_McL), Vd_McL);
```

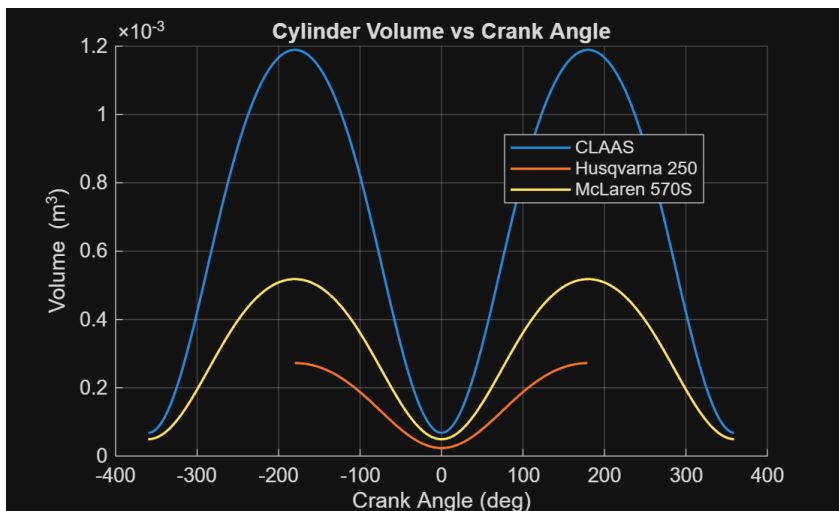
Volume checks:

CLAAS: Vmin=6.796e-05, Vmax=1.189e-03, Vd=1.121e-03

Husq : Vmin=2.348e-05, Vmax=2.723e-04, Vd=2.489e-04

McL : Vmin=4.934e-05, Vmax=5.180e-04, Vd=4.687e-04

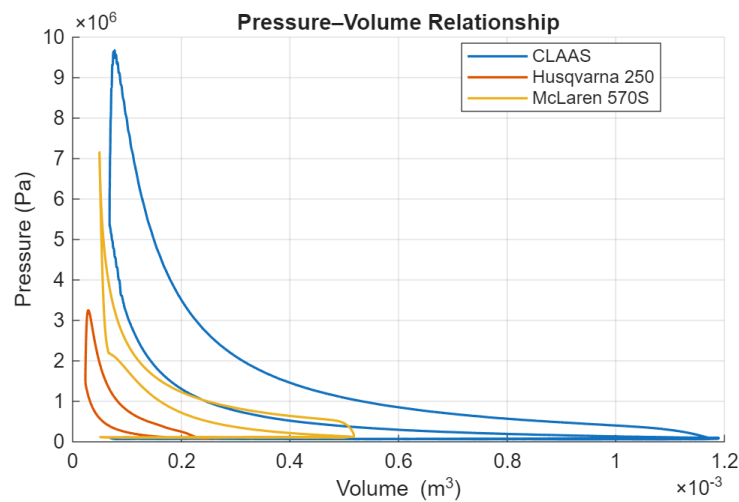
```
% Plot: cylinder volume vs crank angle  
figure('Name','Cylinder Volume vs Crank Angle'); hold on;  
plot(ca_CLAAS, V_CLAAS, 'LineWidth',1.2);  
plot(ca_Husq, V_Husq, 'LineWidth',1.2);  
plot(ca_McL, V_McL, 'LineWidth',1.2);  
xlabel('Crank Angle (deg)');  
ylabel('Volume (m^3)');  
title('Cylinder Volume vs Crank Angle');  
legend('CLAAS','Husqvarna 250','McLaren 570S','Location','best');  
grid on;
```



```

%% Plot: p-V relationship
figure('Name','p-V Relationship'); hold on;
plot(V_CLAAS, p_CLAAS_Pa, 'LineWidth',1.2);
plot(V_Husq, p_Husq_Pa, 'LineWidth',1.2);
plot(V_McL, p_McL_Pa, 'LineWidth',1.2);
xlabel('Volume (m^3)');
ylabel('Pressure (Pa)');
title('Pressure-Volume Relationship');
legend('CLAAS','Husqvarna 250','McLaren 570S','Location','best');
grid on;

```



- Instantaneous cylinder volume calculated using a **slider-crank kinematic model**.
- Geometry defined using bore, stroke, connecting rod length, and compression ratio.
- Volume traces reflect differences in engine size and cycle type:
 - two-stroke cycle completed over 360° ,
 - four-stroke cycles span 720° .
- Pressure-volume data used as the basis for indicated work estimation.

1.8.2 Indicated Work and IMEP Calculations

1.8.2.1 Indicated work and IMEP

```
Wi_CLAAS = trapz(V_CLAAS, p_CLAAS_Pa);  
Wi_Husq = trapz(V_Husq, p_Husq_Pa);  
Wi_McL = trapz(V_McL, p_McL_Pa);  
  
fprintf('Indicated work Wi (J):\n');  
fprintf('  CLAAS           : %.2f\n', Wi_CLAAS);  
fprintf('  Husqvarna 250: %.2f\n', Wi_Husq);  
fprintf('  McLaren 570S : %.2f\n\n', Wi_McL);
```

```
Indicated work Wi (J):  
CLAAS           : 1092.27  
Husqvarna 250: 123.23  
McLaren 570S : 254.48
```

```
% IMEP calculation  
IMEP_CLAAS = Wi_CLAAS / Vd_CLAAS;  
IMEP_Husq = Wi_Husq / Vd_Husq;  
IMEP_McL = Wi_McL / Vd_McL;  
  
% Convert to bar  
IMEP_CLAAS_bar = IMEP_CLAAS / 1e5;  
IMEP_Husq_bar = IMEP_Husq / 1e5;  
IMEP_McL_bar = IMEP_McL / 1e5;  
  
%% Summary table  
Engine = ["CLAAS"; "Husqvarna 250"; "McLaren 570S"];  
IMEP_bar = [IMEP_CLAAS_bar; IMEP_Husq_bar; IMEP_McL_bar];  
  
Section5_Results = table(Engine, IMEP_bar);  
disp(Section5_Results);
```

Engine	IMEP_bar
"CLAAS"	9.7409
"Husqvarna 250"	4.9517
"McLaren 570S"	5.4294

1.8.2.2 Indicated work and IMEP

- Indicated work estimated by numerical integration of the pressure–volume relationship:

$$W_i = \oint p dV$$

- Indicated mean effective pressure (IMEP) calculated as:

$$IMEP = \frac{W_i}{V_d}$$

- IMEP provides a **size-independent metric** for comparing indicated loading.
- Values are indicative and neglect heat transfer and gas exchange losses.

1.8.3 Combined Results

1.8.3.1 Combined Results Summary

```
Engine = ["CLAAS"; "Husqvarna 250"; "McLaren 570S"];

PeakPressure_bar = [PkP_CLAAS; PkP_Husq; PkP_McL];
Max_dP_dCA_barDeg = [MaxdPdCA_CLAAS; MaxdPdCA_Husq; MaxdPdCA_McL];
IMEP_bar = [IMEP_CLAAS_bar; IMEP_Husq_bar; IMEP_McL_bar];

Combined_Results = table( ...
    Engine, ...
    PeakPressure_bar, ...
    Max_dP_dCA_barDeg, ...
    IMEP_bar);

disp('Combined Combustion Pressure Metrics and IMEP Summary');
disp(Combined_Results);
```

Combined Combustion Pressure Metrics and IMEP Summary			
Engine	PeakPressure_bar	Max_dP_dCA_barDeg	IMEP_bar
"CLAAS"	96.75	6.4393	9.7409
"Husqvarna 250"	32.511	0.89079	4.9517
"McLaren 570S"	71.519	2.3761	5.4294

- Key combustion pressure indicators and IMEP presented together:
- peak cylinder pressure,
- maximum pressure rise rate,
- IMEP.
- Summary links combustion behaviour (Section 4) with indicated performance (Section 5).
- Values are drawn from previous sections.

1.8.3.2 Interpretation of Combined Results

- Higher peak pressure and pressure rise rate generally correspond to higher IMEP.
- The CLAAS dataset exhibits the most aggressive pressure behaviour and highest IMEP, consistent with high compression ratio diesel operation.
- The McLaren dataset shows moderated pressure rise despite boost, reflecting knock-limited spark-ignition combustion.
- The Husqvarna dataset exhibits lower absolute pressures but comparable IMEP to the McLaren under the analysed conditions, illustrating that engines of different architecture can exhibit similar indicated loading.

1.9 Comparative Analysis and Indicated Cycle Efficiency

Combustion behaviour

- Diesel combustion produces higher peak pressures and faster pressure rise.
- Spark-ignition combustion under boost requires moderated phasing to avoid knock.
- Two-stroke operation limits pressure development due to scavenging and reduced effective compression.

Indicated performance

- IMEP enables meaningful comparison across engines of differing size and cycle type.
- Absolute power output depends on engine speed and firing frequency, not IMEP alone.

1.10 Engineering Discussion and Key Observations

1.10.1 Key observations

- Combustion strategy and engine architecture strongly influence pressure development.
- Pressure-based metrics provide effective comparative insight using minimal data.
- Similar IMEP values can occur in engines with very different configurations.

1.10.2 Limitations of the Analysis

- Zero-dimensional analysis only.
- No explicit modelling of heat transfer, residuals, or emissions chemistry.
- Pressure-based indicators provide relative rather than absolute combustion timing.

1.10.3 Relevance to Subsequent Emissions Modelling

- Pressure rise behaviour and combustion phasing influence in-cylinder temperature history.
- Results provide a foundation for Wiebe-based combustion modelling

2 Task 2 – WAVE Simulation Report

2.1 Section Introduction

This report investigates the influence of alternative fuels on spark-ignition engine performance using a one-dimensional Ricardo WAVE model of a Ducati Panigale 1199 engine. A gasoline surrogate is used as the baseline fuel and compared against propane and methanol, selected to represent gaseous and oxygenated alternatives with markedly different stoichiometric and energy characteristics.

All fuels are evaluated under identical operating conditions to enable a controlled comparison of mean engine torque, brake mean effective pressure (BMEP), brake specific fuel consumption (BSFC), and in-cylinder pressure development. The objective is to assess whether predicted performance trends align with expectations from fundamental fuel properties within the limitations of simplified combustion modelling.

2.2 Fuel Properties and Model Setup

Fuel	Chemical Formula	Stoichiometric AFR (–)	Lower Heating Value (LHV) (MJ/kg)	Liquid Density (kg/m³)	Data Source
Gasoline (Octane)	C ₈ H ₁₈	15.03	44.43	702.7	(McAllister et al. 2011; Realis WAVE Software 2025)
Propane	C ₃ H ₈	15.58	46.55	507.7	
Methanol	CH ₃ OH	6.43	20.11	791.29	

Table 1: Key thermochemical properties of the fuels considered

2.2.1 Fuel Properties

The key thermochemical properties of the fuels considered in this study are summarised in Table 1. A gasoline surrogate (octane) is used as the baseline fuel and compared against propane and methanol, representing gaseous hydrocarbon and oxygenated alcohol fuels, respectively. These fuels were selected to provide a wide range of stoichiometric air–fuel ratios and energy contents while remaining suitable for spark-ignition operation.

Stoichiometric air–fuel ratio and lower heating value (LHV) are particularly important for interpreting trends in fuel mass flow rate and brake specific fuel consumption (BSFC), as they directly influence the quantity of fuel required to achieve a given equivalence ratio and indicated work output. Fuel density affects injector mass flow behaviour within the WAVE model and therefore influences predicted fuelling requirements.

All fuel properties were implemented using the Realis WAVE fuel database to ensure consistent reference data across simulations.

2.2.2 Summary of WAVE Model Configuration

All simulations were performed using a one-dimensional Realis (formerly Ricardo) WAVE model of a Ducati Panigale 1199 S spark-ignition V-twin engine with a total displacement of 1.198 L. Engine speed was swept from 5000 to 12,000 rpm across eight operating points.

To ensure a controlled comparison, all geometric parameters, valve timings, boundary conditions, and combustion model settings were held constant across fuels. Each fuel was defined using its respective thermochemical properties, while equivalence ratio and ignition timing were fixed for all cases. Combustion was modelled using a prescribed burn approach, appropriate for comparative studies where relative fuel effects are of primary interest.

2.3 Simulation Results

2.3.1 Mean Engine Torque vs Engine Speed

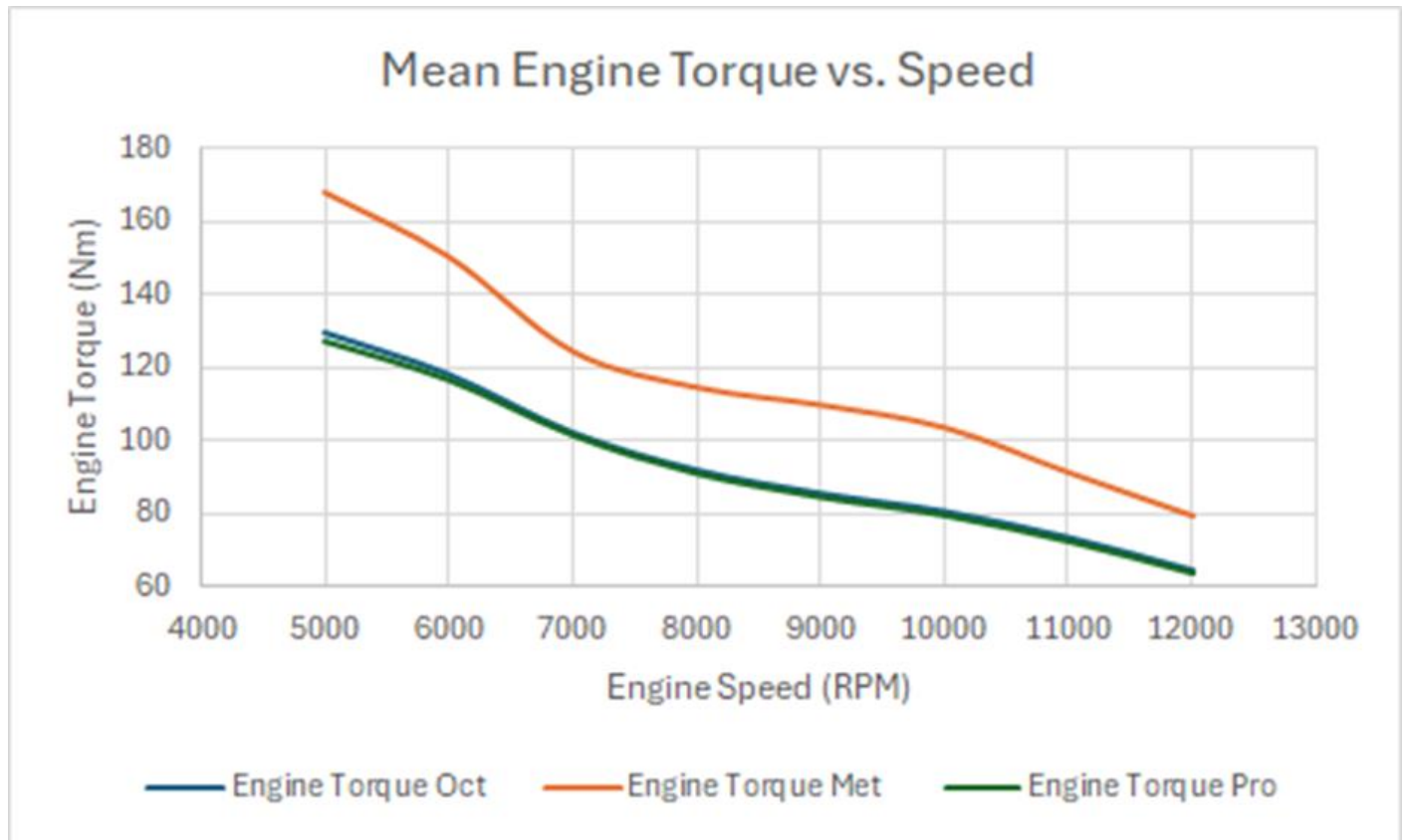


Figure 1: Variation of mean engine torque with engine speed for octane, propane, and methanol under constant equivalence ratio conditions.

Figure 1 shows the variation of mean engine torque with engine speed for octane, propane, and methanol under identical operating conditions. Torque decreases with increasing engine speed for all fuels, reflecting reduced load capability at higher rotational speeds. Octane and propane exhibit closely matched torque levels across the speed range, indicating that fuel type has a limited influence on load capability when equivalence ratio and combustion phasing are held constant. Methanol produces marginally higher torque at most operating points.

2.3.2 Brake Mean Effective Pressure (BMEP)

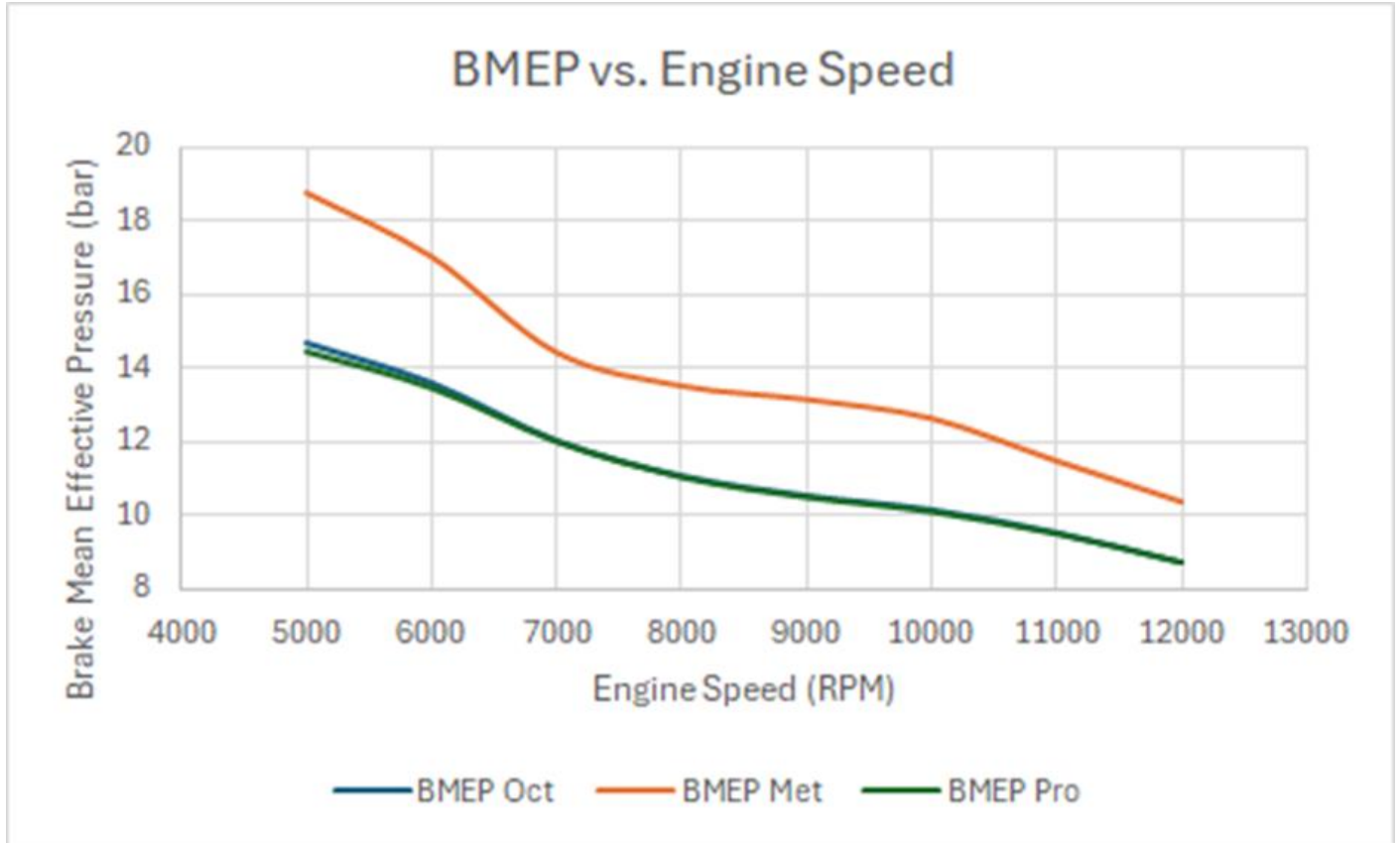


Figure 2: Variation of brake mean effective pressure (BMEP) with engine speed for octane, propane, and methanol under identical operating conditions.

The corresponding BMEP trends, shown in Figure 2, closely mirror the torque results. Octane and propane display near-identical BMEP values, while methanol consistently produces slightly higher BMEP, indicating a modest increase in effective work per cycle under the fixed operating strategy.

2.3.3 Brake Specific Fuel Consumption (BSFC)

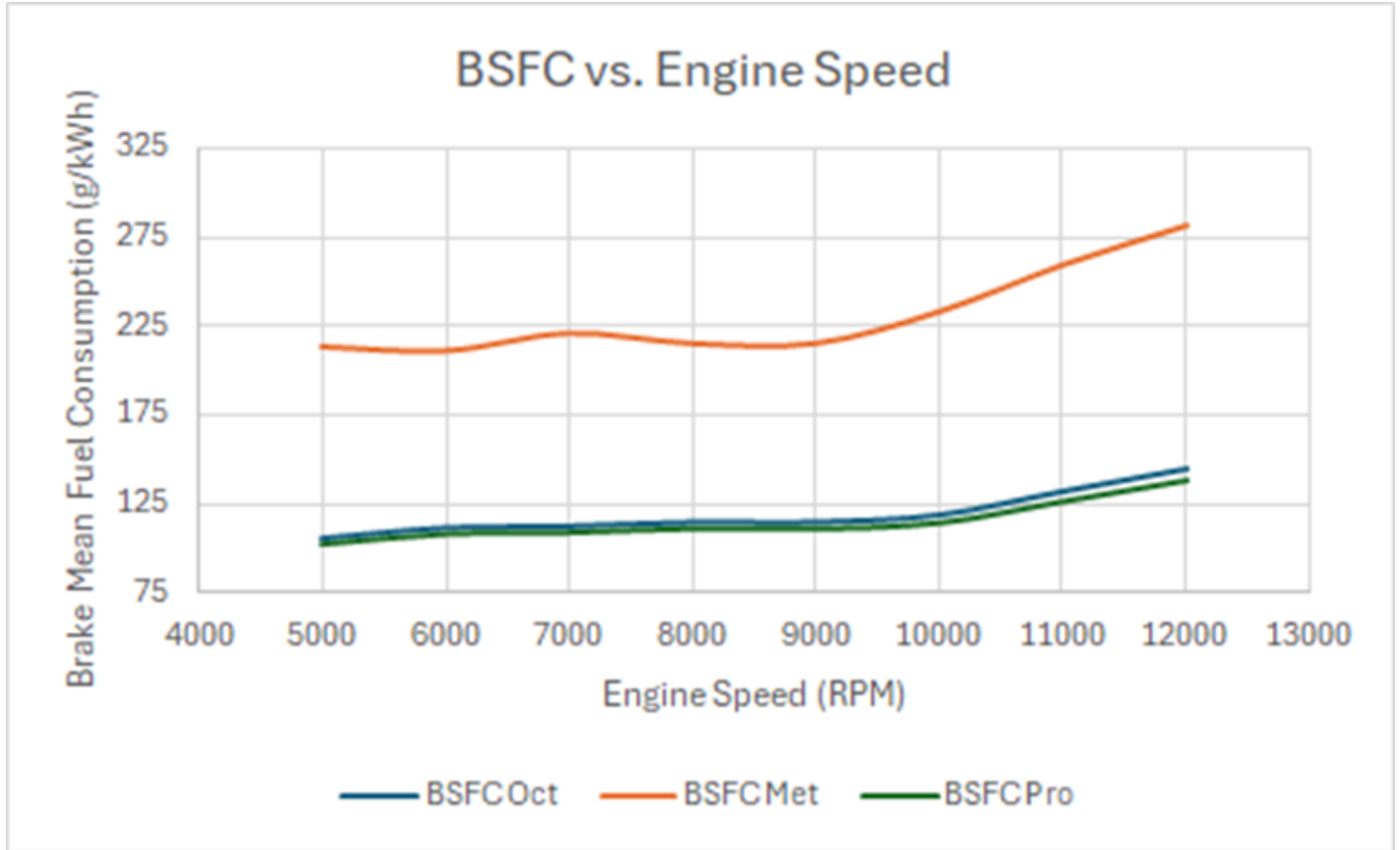


Figure 3: Variation of brake specific fuel consumption (BSFC) with engine speed for octane, propane, and methanol, calculated from cycle-averaged fuel mass flow and indicated power.

Figure 3 presents brake specific fuel consumption (BSFC) as a function of engine speed. In contrast to torque and BMEP, significant differences are observed between fuels. Methanol exhibits substantially higher BSFC across all operating points, indicating a greater fuel mass requirement to achieve comparable indicated work. This behaviour is primarily attributed to methanol's low lower heating value and low stoichiometric air–fuel ratio. Octane and propane show similar BSFC levels, reflecting their comparable gravimetric energy content.

2.3.4 In-Cylinder Pressure vs Crank Angle

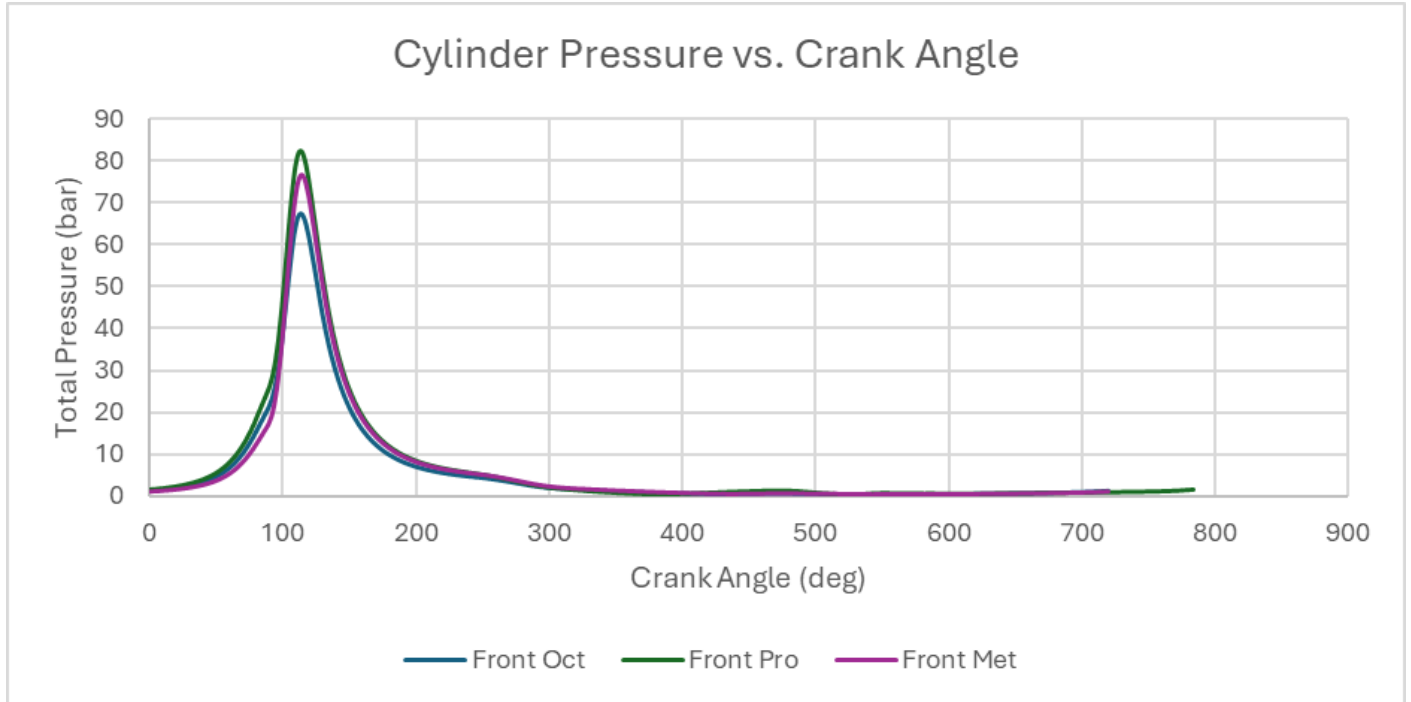


Figure 4: In-cylinder pressure traces for octane, propane, and methanol at [chosen speed, e.g. 7000 rpm] for the front cylinder, shown as a function of crank angle.

Figure 4 shows in-cylinder pressure traces at a representative mid-range engine speed. All fuels exhibit similar pressure development and peak pressure phasing, with peak pressure occurring shortly after top dead centre. The similarity of pressure traces confirms that combustion phasing was effectively constrained by the modelling approach and that observed performance differences arise primarily from fuel properties rather than changes in combustion timing.

2.4 Comparative Analysis of Fuel Effects

All fuels were evaluated under identical operating conditions, with equivalence ratio, engine geometry, and combustion model settings held constant. This controlled approach ensures that observed performance differences arise primarily from variations in fuel properties rather than changes in operating strategy or combustion phasing.

Torque and BMEP results indicate broadly similar load capability for all fuels. Octane and propane exhibit closely matched performance, reflecting their comparable stoichiometric requirements and gravimetric energy content. Methanol produces marginally higher torque and BMEP across much of the speed range, indicating a slight increase in effective work per cycle despite its lower energy content on a mass basis. Given the fixed ignition timing and prescribed burn approach, this behaviour is attributed to thermodynamic effects rather than fuel-specific optimisation.

Clear fuel-dependent trends are observed in brake specific fuel consumption. Methanol consistently exhibits higher BSFC, reflecting its low lower heating value and low stoichiometric air–fuel ratio, which require increased fuelling under fixed equivalence ratio operation. Octane and propane display similar BSFC levels, consistent with their similar energy densities. In-cylinder pressure traces confirm similar combustion phasing across all fuels.

2.5 Model Assumptions and Limitations

The results presented are subject to several modelling assumptions inherent to the one-dimensional Ricardo WAVE framework and the configuration adopted in this study. Combustion was modelled using a prescribed burn approach, which represents heat release through simplified correlations rather than detailed chemical kinetics. As a result, fuel-specific effects related to flame speed, ignition delay, and detailed reaction pathways are not explicitly captured (Realis WAVE Software 2025).

A fixed equivalence ratio and ignition timing strategy was applied across all fuels to enable a controlled comparison. While this approach isolates the influence of fuel properties, it prevents optimisation of combustion phasing or mixture strength for individual fuels and limits the exploitation of fuel-specific advantages.

Heat transfer and in-cylinder losses are represented using empirical correlations, and fuel properties were assumed constant. Consequently, the results should be interpreted as comparative trends rather than predictions of fully optimised engine performance (Realis WAVE Software 2025).

2.6 Conclusions

This study compared the performance of gasoline, propane, and methanol using a one-dimensional Ricardo WAVE model under identical operating conditions. Torque and BMEP trends were broadly similar for all fuels, indicating comparable load capability when equivalence ratio and combustion phasing were fixed. Methanol exhibited substantially higher BSFC, reflecting its low lower heating value and stoichiometric air–fuel ratio. Overall, the predicted trends align with expectations from fundamental

fuel properties and demonstrate the suitability of simplified engine modelling for comparative fuel assessment.

3 Task 3 – Wiebe Combustion Modelling

3.1 Section Introduction and Aim

One-dimensional engine simulation tools such as WAVE rely on simplified combustion sub-models, often using default Wiebe function parameters. While computationally efficient, these parameters may not accurately represent combustion phasing or burn duration for different fuels.

The aim of this task is to refine combustion modelling by estimating Wiebe function parameters from pressure-derived data using MATLAB, and to evaluate the accuracy and limitations of WAVE combustion predictions for alternative fuels.

3.2 Fuels Analysed and Engine Data

Three fuels were analysed: octane, propane, and methanol. octane was used as a reference fuel, while propane and methanol represent alternative fuels with differing combustion characteristics. In-cylinder pressure traces were obtained from WAVE simulations at a fixed engine speed of **9000 rpm**, with identical operating conditions applied for all fuels. These pressure histories form the primary input for the MATLAB combustion modelling and Wiebe parameter calibration.

3.3 Combustion Modelling Methodology

Combustion was modelled using a single-zone Wiebe function to represent the cumulative mass fraction burned as a function of crank angle. The model is defined by the start of combustion, total burn duration, and two shape parameters controlling burn rate and profile. In-cylinder pressure traces obtained from WAVE at a fixed engine speed of 9000 rpm were used to derive apparent heat release, from which initial estimates of combustion phasing and duration were obtained. A homogeneous charge and prescribed pressure–volume history were assumed, and the model is intended to capture relative fuel-to-fuel trends rather than absolute combustion behaviour.

3.4 Implementation of Combustion Model

MATLAB is used in this task to process the WAVE-exported in-cylinder pressure data and implement the combustion model. Pressure traces were imported and converted to consistent units before being differentiated with respect to crank angle. Apparent heat release rate and cumulative heat release were then calculated and used to estimate combustion phasing and burn duration. These quantities provided the basis for fitting Wiebe function parameters and reconstructing pressure traces for comparison with WAVE results.

3.4.1 Data Import and Pre-processing

```
clc; clear; close all;

rootDir = fileparts(mfilename('fullpath'));
dataDir = fullfile(pwd, 'data');

%% Read WAVE pressure CSV files (numeric format)
% Files contain:
%   Column 1: Crank angle [deg]
%   Column 2: Cylinder pressure [bar]
A_oct_f = readmatrix(fullfile(dataDir, 'octane_cylf.csv'));
A_oct_r = readmatrix(fullfile(dataDir, 'octane_cylr.csv'));

A_pro_f = readmatrix(fullfile(dataDir, 'propane_cylf.csv'));
A_pro_r = readmatrix(fullfile(dataDir, 'propane_cylr.csv'));

A_met_f = readmatrix(fullfile(dataDir, 'methanol_cylf.csv'));
A_met_r = readmatrix(fullfile(dataDir, 'methanol_cylr.csv'));

%% Extract crank angle and pressure vectors
ca_oct_f = A_oct_f(:,1);  p_oct_f = A_oct_f(:,2);
ca_oct_r = A_oct_r(:,1);  p_oct_r = A_oct_r(:,2);

ca_pro_f = A_pro_f(:,1);  p_pro_f = A_pro_f(:,2);
ca_pro_r = A_pro_r(:,1);  p_pro_r = A_pro_r(:,2);

ca_met_f = A_met_f(:,1);  p_met_f = A_met_f(:,2);
ca_met_r = A_met_r(:,1);  p_met_r = A_met_r(:,2);

%% Clean pressure traces
% Removes non-finite values, sorts by crank angle, and removes duplicates.
[ca_oct_f, p_oct_f] = cleanTrace(ca_oct_f, p_oct_f);
[ca_oct_r, p_oct_r] = cleanTrace(ca_oct_r, p_oct_r);

[ca_pro_f, p_pro_f] = cleanTrace(ca_pro_f, p_pro_f);
[ca_pro_r, p_pro_r] = cleanTrace(ca_pro_r, p_pro_r);
```

```

[ca_met_f, p_met_f] = cleanTrace(ca_met_f, p_met_f);
[ca_met_r, p_met_r] = cleanTrace(ca_met_r, p_met_r);

%% Apply common crank-angle window
% A consistent window ensures fair comparison of combustion phasing
% and burn duration across fuels and cylinders.
win = [-360, 360];

[ca_oct_f, p_oct_f] = trimWindow(ca_oct_f, p_oct_f, win);
[ca_oct_r, p_oct_r] = trimWindow(ca_oct_r, p_oct_r, win);

[ca_pro_f, p_pro_f] = trimWindow(ca_pro_f, p_pro_f, win);
[ca_pro_r, p_pro_r] = trimWindow(ca_pro_r, p_pro_r, win);

[ca_met_f, p_met_f] = trimWindow(ca_met_f, p_met_f, win);
[ca_met_r, p_met_r] = trimWindow(ca_met_r, p_met_r, win);

%% Quality check: retained data points
fprintf('Points after trimming:\n');
fprintf('Octane    cylF = %d | cylR = %d\n', numel(ca_oct_f), numel(ca_oct_r));
fprintf('Propane   cylF = %d | cylR = %d\n', numel(ca_pro_f), numel(ca_pro_r));
fprintf('Methanol  cylF = %d | cylR = %d\n', numel(ca_met_f), numel(ca_met_r));

```

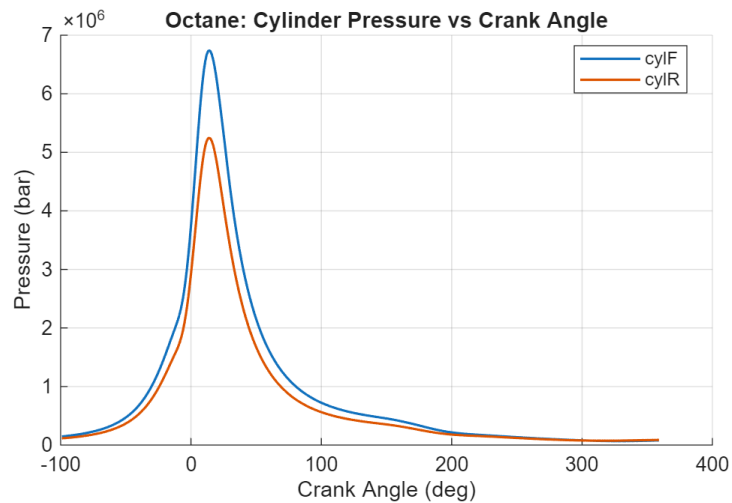
```

Points after trimming:
Octane    cylF = 459 | cylR = 459
Propane   cylF = 522 | cylR = 522
Methanol  cylF = 459 | cylR = 459

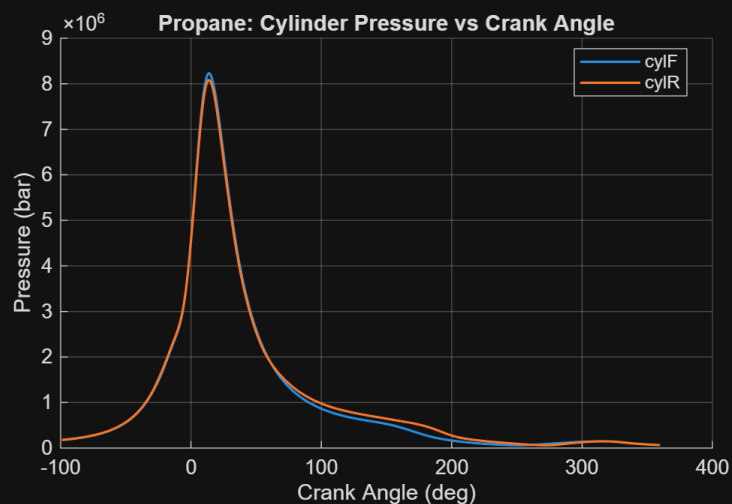
```

```
%% Visual check: pressure vs crank angle
```

```
figure('Name','Octane: Pcyl vs CA'); hold on;  
plot(ca_oct_f, p_oct_f, 'LineWidth', 1.2);  
plot(ca_oct_r, p_oct_r, 'LineWidth', 1.2);  
xlabel('Crank Angle (deg)'); ylabel('Pressure (bar)');  
title('Octane: Cylinder Pressure vs Crank Angle');  
legend('cylF','cylR','Location','best'); grid on;
```



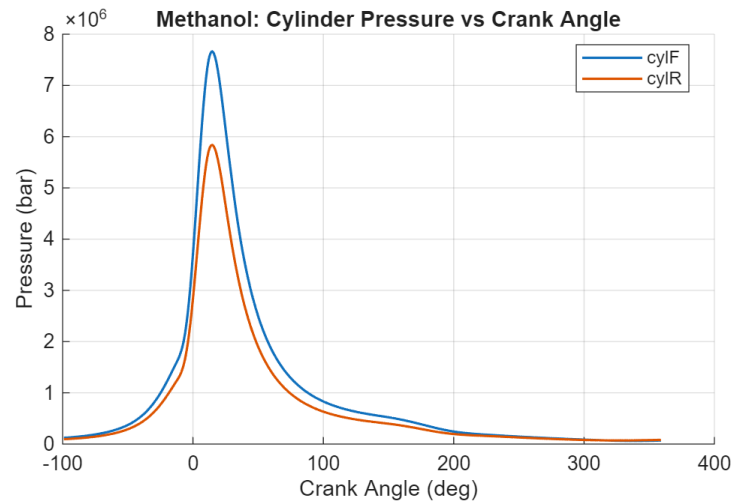
```
figure('Name','Propane: Pcyl vs CA'); hold on;  
plot(ca_pro_f, p_pro_f, 'LineWidth', 1.2);  
plot(ca_pro_r, p_pro_r, 'LineWidth', 1.2);  
xlabel('Crank Angle (deg)'); ylabel('Pressure (bar)');  
title('Propane: Cylinder Pressure vs Crank Angle');  
legend('cylF','cylR','Location','best'); grid on;
```



```

figure('Name','Methanol: Pcyl vs CA'); hold on;
plot(ca_met_f, p_met_f, 'LineWidth', 1.2);
plot(ca_met_r, p_met_r, 'LineWidth', 1.2);
xlabel('Crank Angle (deg)'); ylabel('Pressure (bar)');
title('Methanol: Cylinder Pressure vs Crank Angle');
legend('cylF','cylR','Location','best'); grid on;

```



3.4.2 Apparent Heat Release Analysis

```
% Derive SOC and burn duration from pressure data (for Wiebe fitting)

% --- Settings (single operating point, 9000 rpm) ---
gamma = 1.35;           % effective gamma for single-zone aHRR
useVolumeFromCSV = false; % set true only if your CSV includes V(θ) [m^3]
volColumnIndex = 3;

% Geometry (used only if useVolumeFromCSV = false)
B = 0.112;      % [m]
S = 0.0608;     % [m]
L = 0.1102;     % [m]
CR = 12.5;      % [-]

% Select cylinder for calibration
useCylinder = "cylF"; % Calibration cylinder

% Derivative stabilisation (toolbox-free)
doSmooth = true;
movAvgWindow = 5;      % number of CA points (keep small to avoid shifting SOC)

fuels = ["octane", "propane", "methanol"];
mapName = struct("octane", "oct", "propane", "pro", "methanol", "met");

results = struct();

for k = 1:numel(fuels)
    fuel = fuels(k);
    tag = mapName.(fuel);

    % --- Select dataset (pressure already in Pa) ---
    if useCylinder == "cylF"
        CA = eval("ca_" + tag + "_f");
        P = eval("p_" + tag + "_f");
    else
        CA = eval("ca_" + tag + "_r");
        P = eval("p_" + tag + "_r");
    end

    CA = CA(:); P = P(:);

    % --- Smooth pressure only for numerical differentiation ---
    if doSmooth
        P_s = movmean(P, movAvgWindow);
    else
```

```

    P_s = P;
end

% --- Volume trace V(θ) ---
if useVolumeFromCSV
    if useCylinder == "cylF"
        A = eval("A_" + tag + "_f");
    else
        A = eval("A_" + tag + "_r");
    end
    V = A(:,volColumnIndex);
    V = V(1:numel(CA));
else
    r = S/2;
    A_p = pi*(B^2)/4;

    Vd = A_p * S;
    Vc = Vd / (CR - 1);

    theta = deg2rad(CA);
    x = r*(1 - cos(theta)) + (L - sqrt(L^2 - (r*sin(theta)).^2));
V = Vc + A_p .* x;
end

% --- Gradients vs crank angle (deg) ---
dP_dCA = gradient(P_s, CA);
dV_dCA = gradient(V, CA);

% --- Single-zone apparent heat release rate (aHRR) ---
dQdCA = (gamma/(gamma-1))*P_s.*dV_dCA + (1/(gamma-1))*V.*dP_dCA;

% --- Full-cycle cumulative HR (only to get a provisional SOC for windowing) ---
Q = cumtrapz(CA, dQdCA);
Qn = Q - min(Q);

if max(Qn) > 0
    xb_tmp = Qn ./ max(Qn);
else
    xb_tmp = zeros(size(Qn));
end

SOC_tmp = interp1(xb_tmp, CA, 0.01, 'linear', 'extrap');

% --- Windowed cumulative HR to avoid negative tail dominating xb ---
idx_comb = CA >= (SOC_tmp - 5) & CA <= (SOC_tmp + 120);
CAw = CA(idx_comb);

```

```

dQw = dQdCA(idx_comb);

Q_comb = cumtrapz(CAw, dQw);
Q_comb = Q_comb - min(Q_comb);

% Stop at maximum cumulative HR (end of net heat release)
[Qmax, iMax] = max(Q_comb);
CAw = CAw(1:iMax);
Q_comb = Q_comb(1:iMax);

if Qmax > 0
    xb = Q_comb ./ Qmax; % final burn fraction proxy (monotonic)
else
    xb = zeros(size(Q_comb));
end

% --- Combustion phasing metrics (from windowed xb) ---
SOC = interp1(xb, CAw, 0.01, 'linear', 'extrap');
CA10 = interp1(xb, CAw, 0.10, 'linear', 'extrap');
CA50 = interp1(xb, CAw, 0.50, 'linear', 'extrap');
CA90 = interp1(xb, CAw, 0.90, 'linear', 'extrap');

fuelKey = char(fuel); % ensures field is a proper struct field name

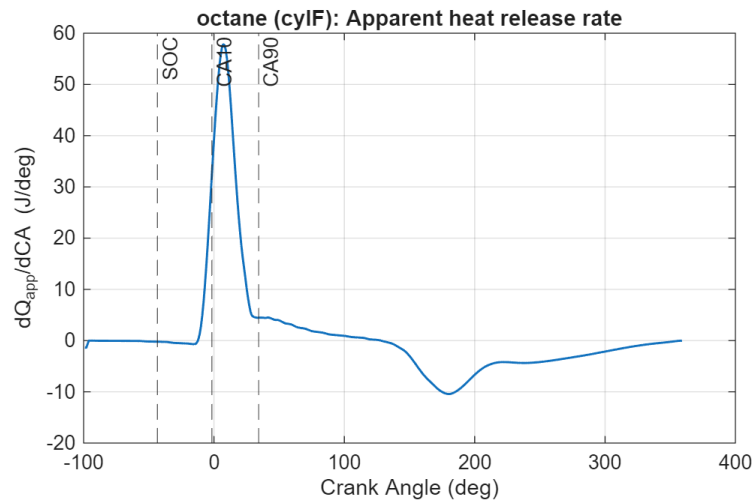
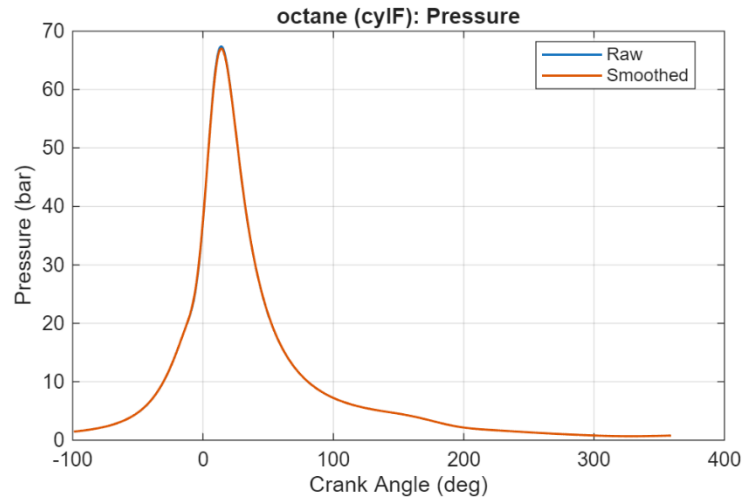
results.(fuel).CA = CA;
results.(fuel).CAw = CAw;
results.(fuel).P = P;
results.(fuel).P_s = P_s;
results.(fuel).V = V;
results.(fuel).dP_dCA = dP_dCA;
results.(fuel).dQdCA = dQdCA;
results.(fuel).Q = Q;
results.(fuel).xb = xb;
results.(fuel).SOC = SOC;
results.(fuel).CA10 = CA10;
results.(fuel).CA50 = CA50;
results.(fuel).CA90 = CA90;
results.(fuel).dur_10_90 = CA90 - CA10;

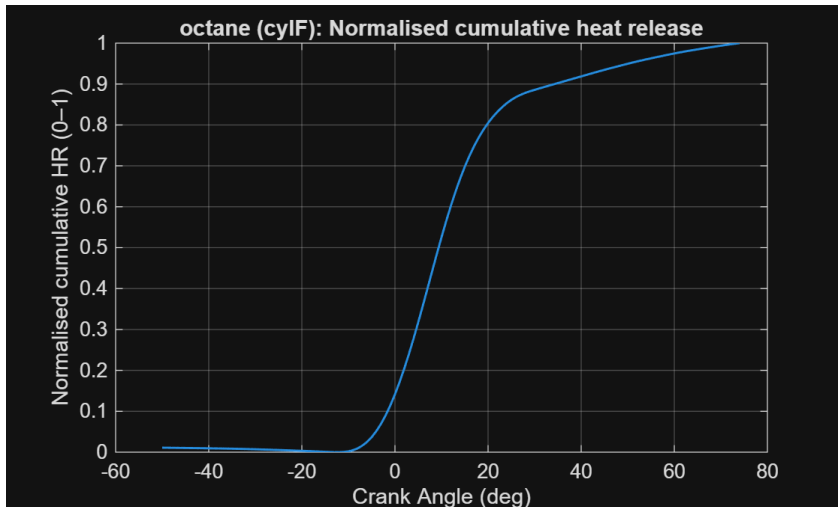
fprintf("%s (%s): SOC~%.1f | CA10=%.1f | CA50=%.1f | CA90=%.1f | Dur10-90=%.1f\n", ...
    fuel, useCylinder, SOC, CA10, CA50, CA90, CA90-CA10);

```

```
% --- Plots (kept consistent for cross-fuel comparison) ---
figure('Name', fuel + " - Pressure (raw vs smoothed)");
plot(CA, P/1e5, 'LineWidth', 1.1); hold on;
plot(CA, P_s/1e5, 'LineWidth', 1.1);
xlabel('Crank Angle (deg)'); ylabel('Pressure (bar)');
title(fuel + " (" + useCylinder + "): Pressure");
legend('Raw', 'Smoothed', 'Location', 'best'); grid on;
```

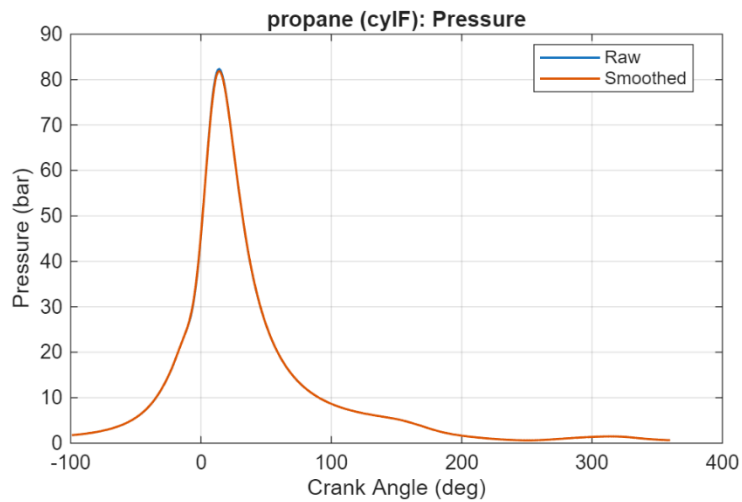
octane (cylF): SOC~-43.4 | CA10=-1.6 | CA50=9.4 | CA90=34.3 | Dur10-90=35.9 deg

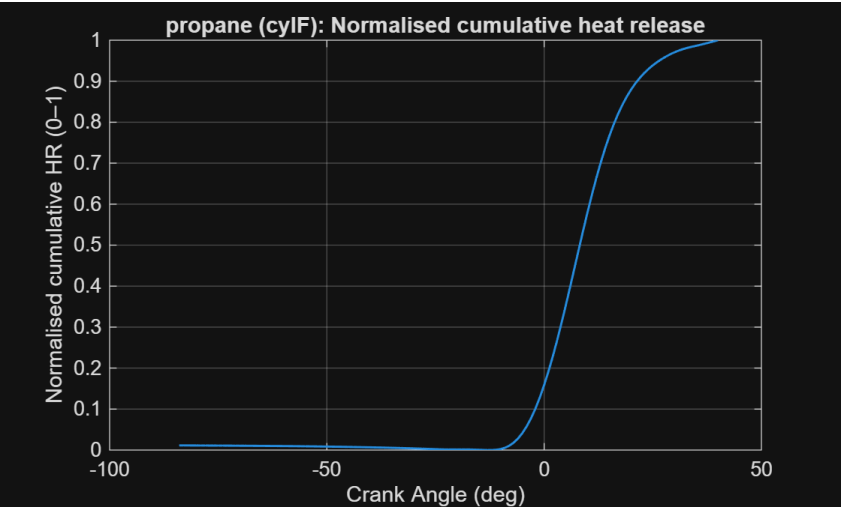
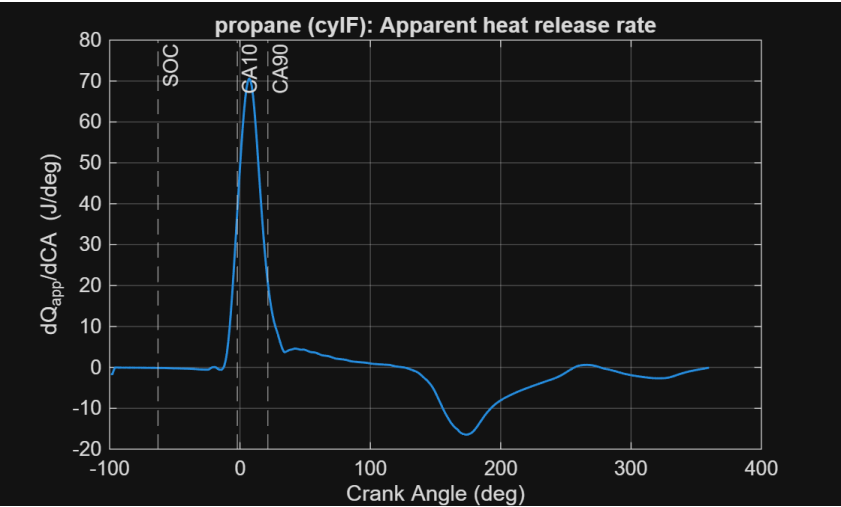




```
figure('Name', fuel + " - aHRR");
plot(CA, dQdCA, 'LineWidth', 1.1);
xlabel('Crank Angle (deg)'); ylabel('dQ_{app}/dCA (J/deg)');
title(fuel + " (" + useCylinder + "): Apparent heat release rate");
grid on; xline(SOC, '--', 'SOC'); xline(CA10, '--', 'CA10'); xline(CA90, '--', 'CA90');
```

propane (cylF): SOC=-62.9 | CA10=-2.1 | CA50=8.2 | CA90=21.4 | Dur10-90=23.5 deg



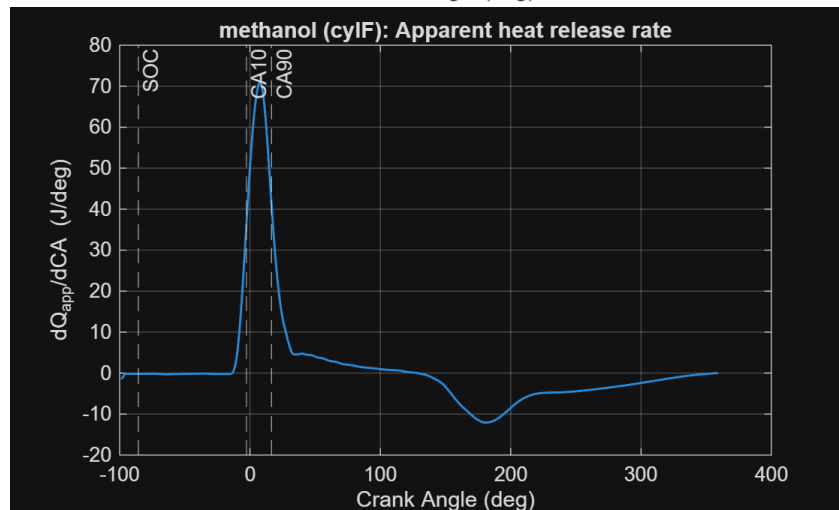
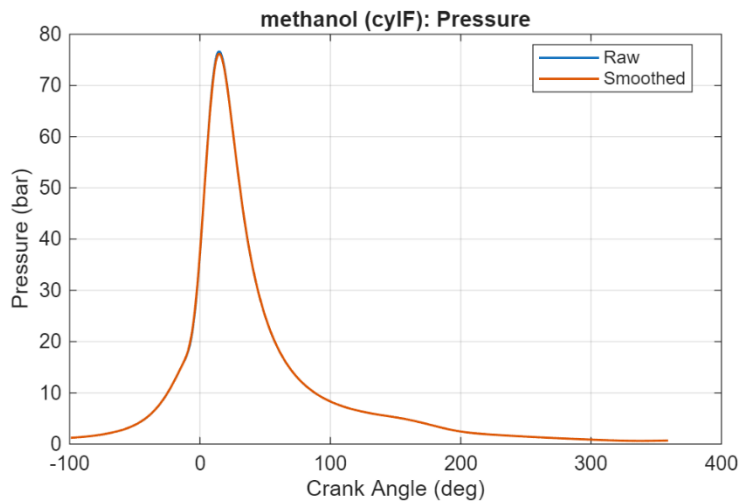


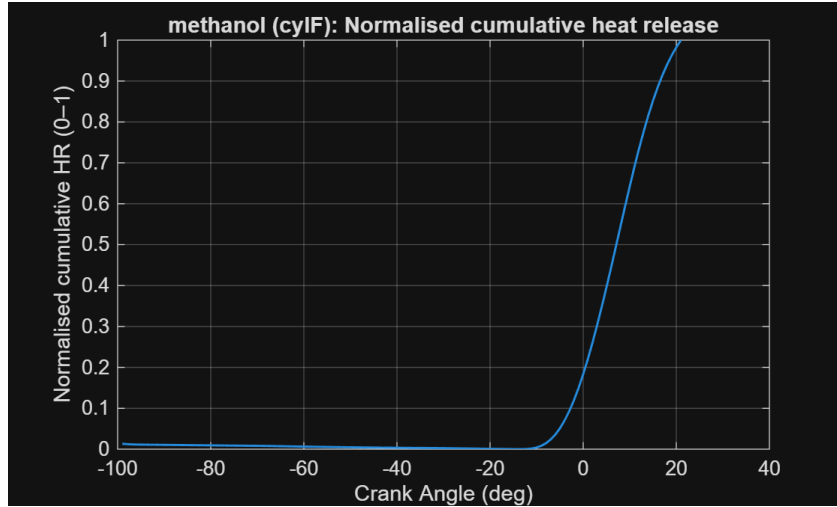
```

figure('Name', fuel + " - Burn fraction proxy (xb)");
plot(CAw, xb, 'LineWidth', 1.1);
xlabel('Crank Angle (deg)'); ylabel('Normalised cumulative HR (0-1)');
title(fuel + " (" + useCylinder + "): Normalised cumulative heat release");
grid on;
end

```

methanol (cylF): SOC=-85.7 | CA10=-2.7 | CA50=7.1 | CA90=16.5 | Dur10-90=19.3 deg





The burn fraction proxy derived from apparent heat release represents one of several valid post-processing choices. In this study, the cumulative heat release was defined to increase monotonically over the main combustion window and reach unity at the point of maximum net heat release, providing a stable basis for CA10–CA90 extraction and Wiebe fitting. Alternative definitions are possible and highlight the sensitivity of simplified combustion models to post-processing assumptions rather than unique physical truth.

3.5 Wiebe Parameter Estimation

$$x_b(\theta) = 1 - \exp \left[-a \left(\frac{\theta - \theta_0}{\Delta\theta} \right)^{m+1} \right]$$

(Stone and Macmillan, 2012)

where x_b is the cumulative burn fraction, θ_0 the start of combustion, $\Delta\theta$ the burn duration, and a and m are Wiebe shape parameters controlling burn completeness and rate.

3.5.1 Initial Parameter Selection

SOC and burn duration were derived from pressure-based cumulative heat release, with SOC defined at 1% burn fraction and burn duration from CA10–CA90. These values were used to initialise the Wiebe function, with initial guesses of $a=4.6$ and $m=2$, consistent with typical spark-ignition engine ranges.

3.5.2 Curve Fitting and Calibration

```
fuels = {'octane','propane','methanol'};
xb_end = 0.99;
a_fix = -log(1 - xb_end); % completeness parameter

fit = struct();

for k = 1:numel(fuels)
    fuel = fuels{k};

    CAw = results.(fuel).CAw(:);
    xb = results.(fuel).xb(:);

    SOC = results.(fuel).SOC;

    % Use your CA10-CA90-based duration as initial guess
    CA10 = results.(fuel).CA10;
    CA90 = results.(fuel).CA90;
    dth0 = max(10, (CA90 - CA10) / 0.8); % rough map from 10-90 to 0-100%
    m0 = 2;

    theta = CAw - SOC;

    % Wiebe function (only active after SOC)
    wiebe = @(th, dth, m) (th<=0).*0 + (th>=dth).*1 + ...
        ((th>0 & th<dth).*(1 - exp(-a_fix.*(th./dth).^(m+1))));

    % Objective: SSE to pressure-derived xb in combustion window
    obj = @(x) sum( (wiebe(theta, max(1,x(1)), max(0.2,x(2))) - xb).^2 );

    % Optimise [dtheta, m]
    m_lo = 0.8; m_hi = 4.0;
    d_lo = 10; d_hi = 200;

    pen = @(dth,m) 1e4*(max(0,m_lo-m)^2 + max(0,m-m_hi)^2 + ...
        max(0,d_lo-dth)^2 + max(0,dth-d_hi)^2);

    obj = @(x) sum( (wiebe(theta, max(1,x(1)), max(0.2,x(2))) - xb).^2 ) ...
        + pen(max(1,x(1)), max(0.2,x(2))));

    xopt = fminsearch(obj, [dth0, m0], optimset('Display','off'));
    dth_fit = max(1, xopt(1));
    m_fit = max(0.2, xopt(2));

    xb_fit = wiebe(theta, dth_fit, m_fit);
```

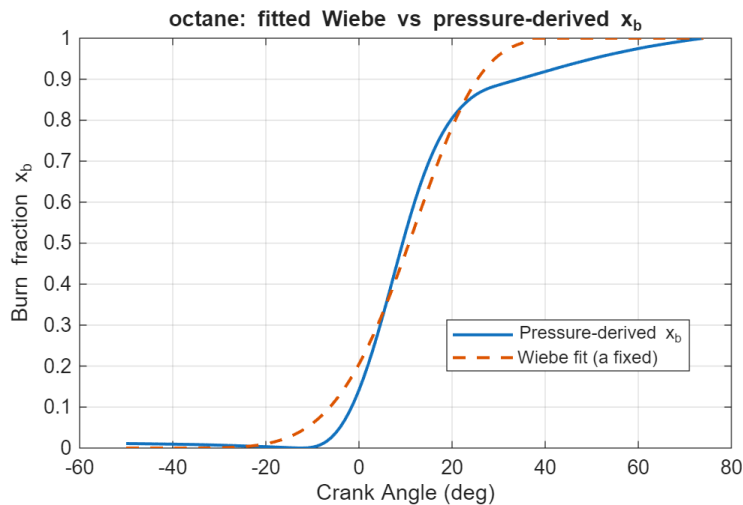
```

% Store
fit.(fuel).a = a_fix;
fit.(fuel).m = m_fit;
fit.(fuel).dtheta = dth_fit;

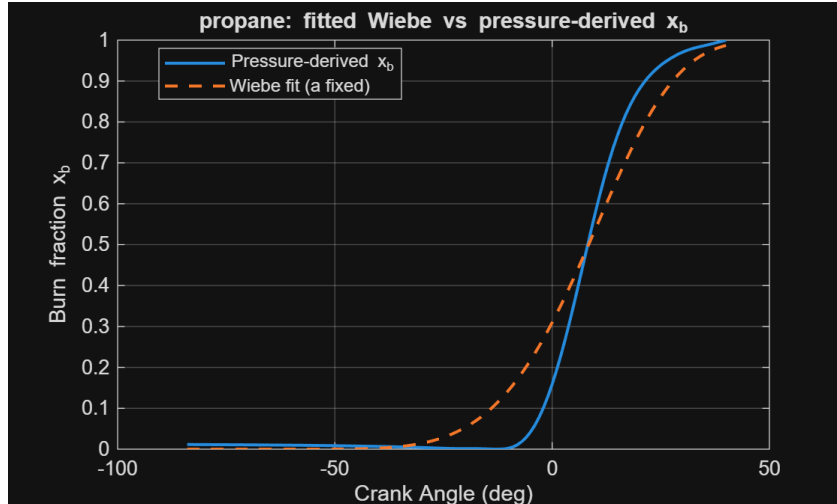
% Plot comparison
figure('Name', [fuel ' - Wiebe fit to pressure-derived x_b']);
plot(CAw, xb, 'LineWidth', 1.5); hold on;
plot(CAw, xb_fit, '--', 'LineWidth', 1.5);
grid on; ylim([0 1]);
xlabel('Crank Angle (deg)'); ylabel('Burn fraction x_b');
title([fuel ': fitted Wiebe vs pressure-derived x_b']);
legend('Pressure-derived x_b', 'Wiebe fit (a fixed)', 'Location', 'best');

fprintf('%s fit: a=%.3f (fixed), m=%.2f, dtheta=%.1f deg\n', fuel, a_fix, m_fit,
dth_fit);
end

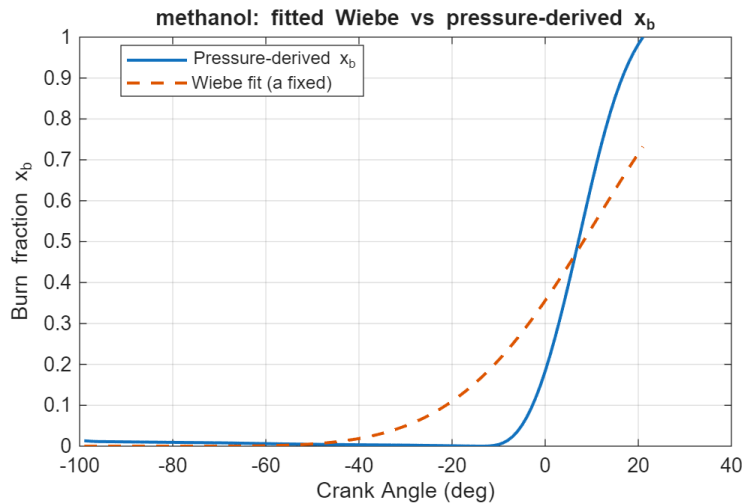
```



octane fit: a=4.605 (fixed), m=4.00, dtheta=79.1 deg



propane fit: $a=4.605$ (fixed), $m=4.00$, $d\theta=104.1$ deg



methanol fit: $a=4.605$ (fixed), $m=4.00$, $d\theta=137.0$ deg

Methanol and propane are modelled as idealised single-component fuels within the WAVE framework. Differences in combustion behaviour are therefore inferred from pressure-derived metrics rather than resolved chemical kinetics or mixture preparation effects. Observed trends should be interpreted as effective fuel behaviour under the prescribed modelling assumptions.

Methanol exhibits a highly delayed but rapid increase in pressure-derived burn fraction. When fitted without constraints, this behaviour drives the Wiebe shape parameter towards extreme values, reflecting limitations in parameter identifiability under pressure-only fitting. Applying bounds limits this sensitivity, resulting in slightly reduced numerical fit quality but a more physically plausible combustion representation.

- Similar Wiebe shapes result from comparable imposed burn durations across all fuels.
- Fuel effects appear mainly in combustion phasing, with earlier inferred start of combustion; however, the underlying physical mechanisms cannot be isolated with the present modelling approach.

- All cases reach complete combustion ($x_b = 1$) within the defined crank-angle window.

3.6 Pressure Reconstruction and Validation vs WAVE

```
% --- Reconstruction settings ---
gamma_rec = 1.35;           % keep consistent with aHRR section
P_units   = "bar";         % your imported WAVE pressure is in bar (per comments)
assume_bar_input = true;    % set false if your p_* already converted to Pa
elsewhere

% Choose which cylinder pressure to validate against (same as calibration)
useCylinder = "cylF";      % must match earlier choice

% Wiebe parameters used (currently fixed)
m_use = fit.(fuel).m;
xb_end = 0.99;

% Reconstruction window extension (degCA)
pad_before = 20;
pad_after  = 40;

fuels = {'octane','propane','methanol'};

val = struct(); % store validation metrics

for k = 1:numel(fuels)
    fuel = fuels{k};

    % Pull data from results
    CA    = results.(fuel).CA(:);
    Praw  = results.(fuel).P(:);
    V     = results.(fuel).V(:);
    P_wave = Praw;

    % Need dV/dCA on same grid
    dVdCA = gradient(V, CA);

    % Use your phasing from Section 4.2 / 5
    SOC    = results.(fuel).SOC;
    CA10   = results.(fuel).CA10;
    CA90   = results.(fuel).CA90;

    % Burn duration estimate (same approach you used in section 5.2)
    th10 = CA10 - SOC;
    th90 = CA90 - SOC;
    dtheta_from = @(th, x) th ./ ((-log(1-x))./a_fix).^(1/(m_use+1));
    dth_use = fit.(fuel).dtheta;
```

```

% Build Wiebe xb on the full CA grid (so we can reconstruct pressure on same
grid)
theta = CA - SOC;
xb = zeros(size(theta));
idx = (theta >= 0) & (theta <= dth_use);
xb(idx) = 1 - exp( -a_fix .* (theta(idx) ./ dth_use).^(m_use + 1) );
xb(theta > dth_use) = 1;

% Heat release rate: dQ/dCA = Q_total * dxb/dCA
% Use total apparent heat release from your windowed aHRR result as Q_total
proxy:
% (This ties reconstruction magnitude to your pressure-derived heat release.)
Q_total = max(results.(fuel).Q) - min(results.(fuel).Q); % J (proxy)
dxb_dCA = gradient(xb, CA);
dQdCA = Q_total .* dxb_dCA;

% Select reconstruction window around combustion to avoid integrating full 720deg
CA_start = SOC - pad_before;
CA_end = SOC + dth_use + pad_after;
idx_rec = CA >= CA_start & CA <= CA_end;

CAr = CA(idx_rec);
Vr = V(idx_rec);
dV = dVdCA(idx_rec);
dQ = dQdCA(idx_rec);
Pw = P_wave(idx_rec);

% Initial condition: start from WAVE at window start
p_rec = zeros(size(CAr));
p_rec(1) = Pw(1);

% Integrate dp/dCA
for i = 1:numel(CAr)-1
    dCA = CAr(i+1) - CAr(i);

    dpdCA = ((gamma_rec-1)/Vr(i))*dQ(i) - gamma_rec*(p_rec(i)/Vr(i))*dV(i);

    p_rec(i+1) = p_rec(i) + dpdCA*dCA;
end

% ---- Validation metrics ----
[pmax_w, iw] = max(Pw);
[pmax_r, ir] = max(p_rec);

th_pmax_w = CAr(iw);

```

```

th_pmax_r = CAr(ir);

rmse = sqrt(mean((p_rec - Pw).^2));
mae = mean(abs(p_rec - Pw));

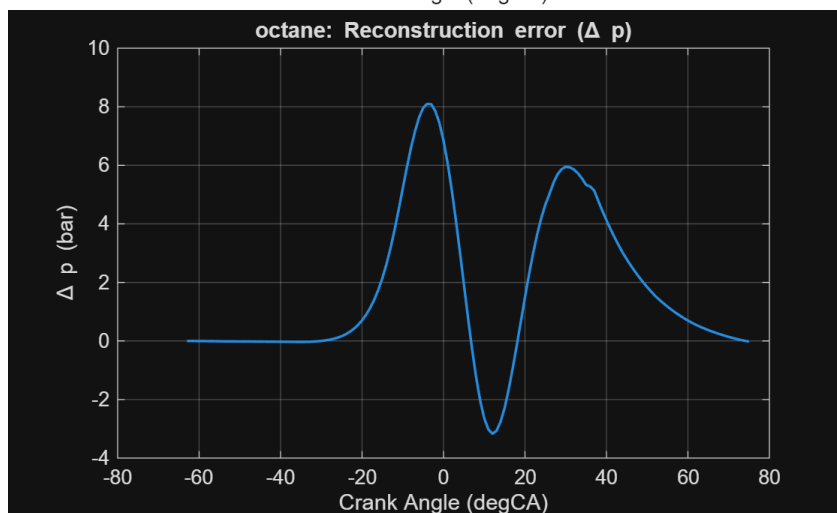
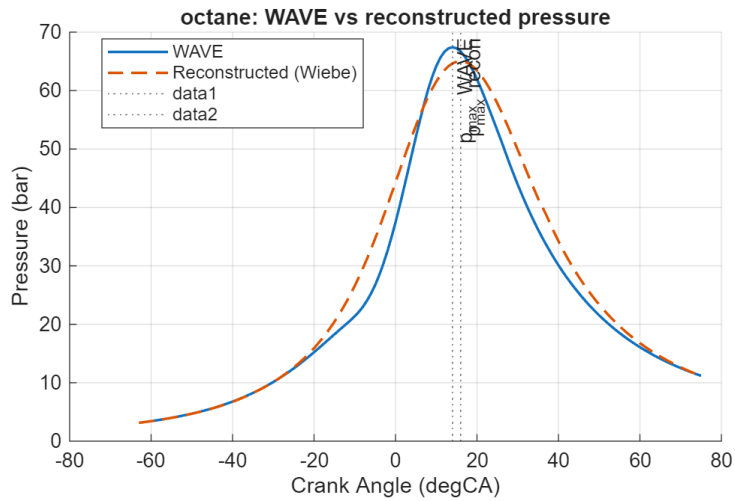
% Store
val.(fuel).CAr = CAr;
val.(fuel).Pw = Pw;
val.(fuel).Pr = p_rec;
val.(fuel).pmax_w = pmax_w;
val.(fuel).pmax_r = pmax_r;
val.(fuel).th_pmax_w = th_pmax_w;
val.(fuel).th_pmax_r = th_pmax_r;
val.(fuel).rmse = rmse;
val.(fuel).mae = mae;

% ---- Plots ----
figure('Name', [fuel ' - Pressure overlay (WAVE vs reconstructed)']); hold on;
plot(CAr, Pw/1e5, 'LineWidth', 1.3);
plot(CAr, p_rec/1e5, '--', 'LineWidth', 1.3);
xlabel('Crank Angle (degCA)'); ylabel('Pressure (bar)');
title([fuel ': WAVE vs reconstructed pressure']);
legend('WAVE','Reconstructed (Wiebe)','Location','best'); grid on;
xline(th_pmax_w, ':', 'p_{max} WAVE'); xline(th_pmax_r, ':', 'p_{max} recon');

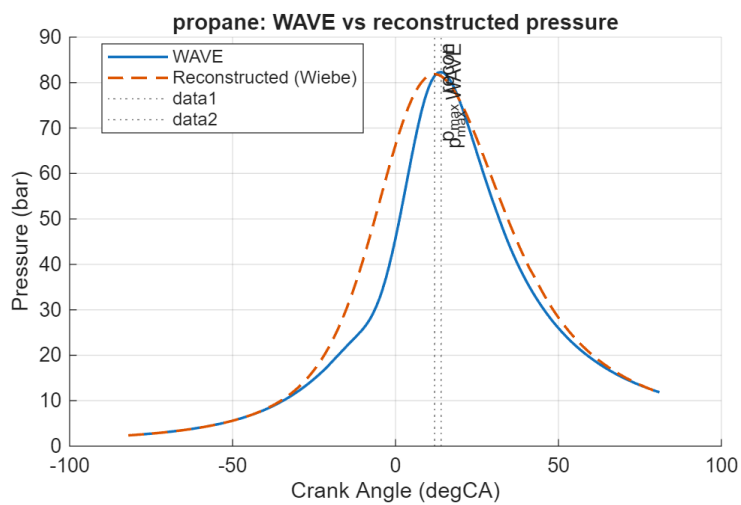
figure('Name', [fuel ' - Pressure residual']);
plot(CAr, (p_rec - Pw)/1e5, 'LineWidth', 1.3);
xlabel('Crank Angle (degCA)'); ylabel('\Delta p (bar)');
title([fuel ': Reconstruction error (\Delta p)']);
grid on;

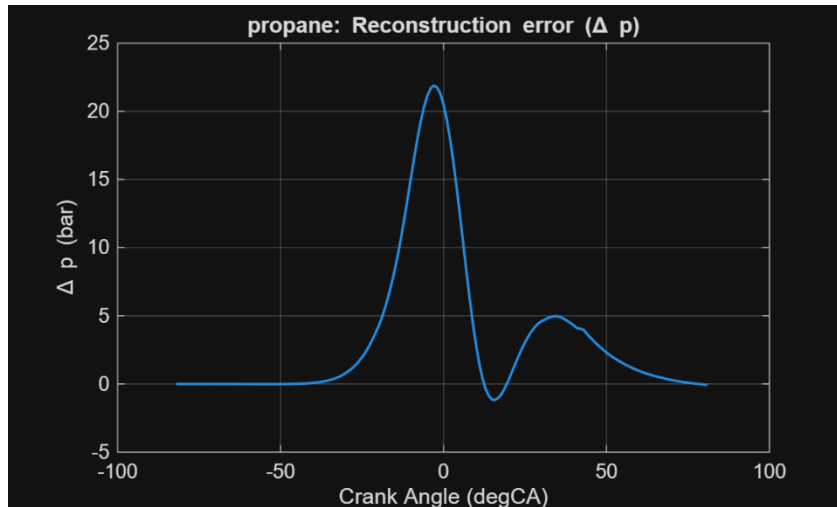
fprintf('%s: pmax_W=%.2f bar @ %.2f | pmax_R=%.2f bar @ %.2f | \Delta\theta=%.2f deg |
RMSE=%.2f bar\n', ...
    fuel, pmax_w/1e5, th_pmax_w, pmax_r/1e5, th_pmax_r, (th_pmax_r-th_pmax_w),
    rmse/1e5);
end

```

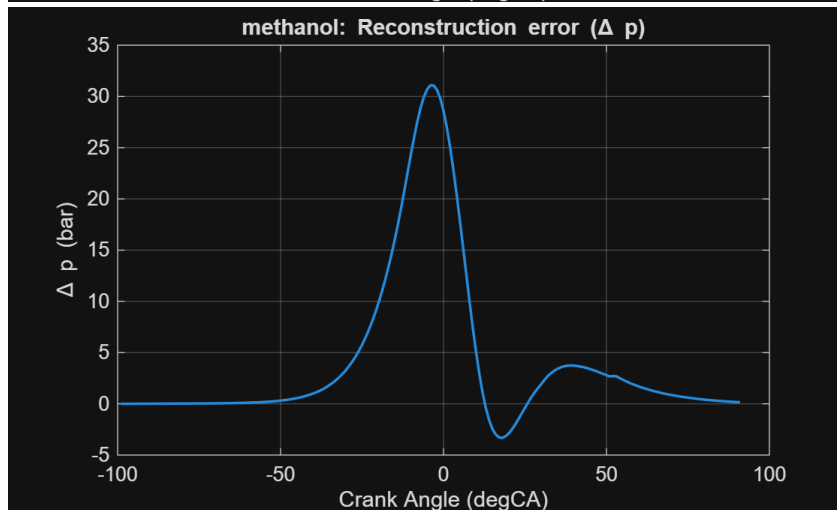
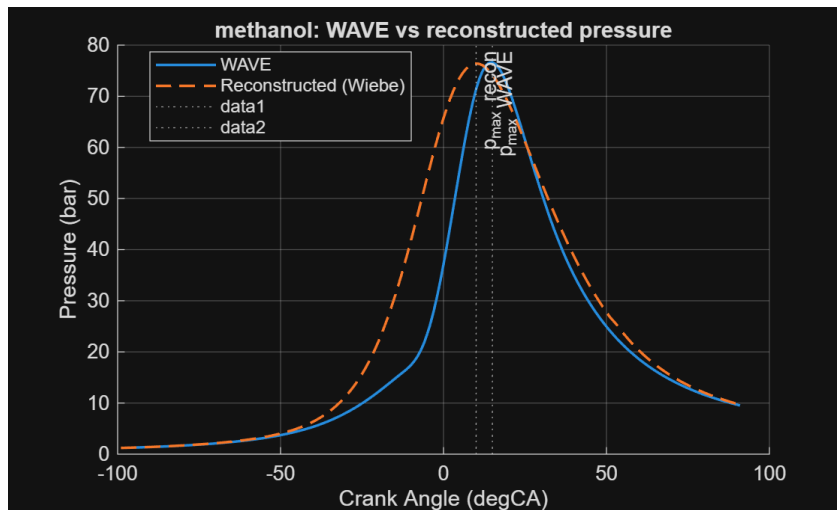


octane: p_{max_W} =67.41 bar @ 14.00 | p_{max_R} =65.03 bar @ 16.00 | $\Delta\theta$ =2.00 deg | RMSE=3.16 bar





propane: $p_{max_W}=82.31$ bar @ 14.00 | $p_{max_R}=81.86$ bar @ 12.00 | $\Delta\theta=-2.00$ deg | RMSE=6.64 bar



methanol: $p_{max_W}=76.65$ bar @ 15.00 | $p_{max_R}=76.41$ bar @ 10.00 | $\Delta\theta=-5.00$ deg | RMSE=9.12 bar

The reconstruction error exhibits systematic, phase-dependent behaviour, with early overprediction and post-TDC underprediction of pressure. These deviations arise from the use of a single-zone Wiebe

combustion model with constant γ and no heat transfer or blow-by losses, rather than numerical instability.

3.7 Sensitivity Analysis of Wiebe Parameters

```
fuel = 'octane';

% Pull base signals
CA = results.(fuel).CA(:);
V = results.(fuel).V(:);
dVdCA = gradient(V, CA);

% WAVE pressure
Praw = results.(fuel).P(:);

% Baseline Wiebe parameters
SOC0 = results.(fuel).SOC;
CA10 = results.(fuel).CA10;
CA90 = results.(fuel).CA90;

a0 = 4.605;      % xb_end = 0.99
m0 = 2;         % baseline shape factor
dth0 = (CA90 - CA10)/0.8; % baseline duration estimate

% Total heat released (prefer windowed if you stored it earlier)
if isfield(results.(fuel), 'Q_total_comb')
    Q_total = results.(fuel).Q_total_comb;
else
    Q_total = max(results.(fuel).Q) - min(results.(fuel).Q);
end

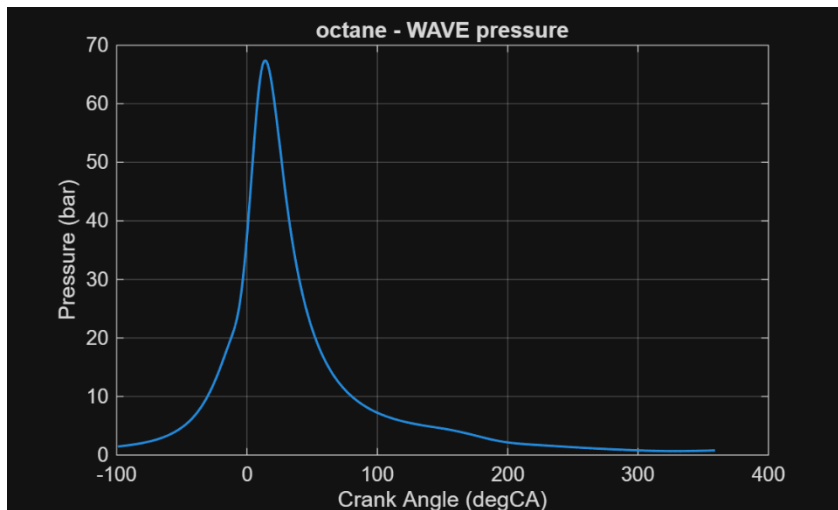
% Baseline prints
fprintf('Section 7 baseline (%s): SOC=%.2f degCA, CA10=%.2f, CA90=%.2f, dtheta=%.2f, m=%.2f\n', ...
    fuel, SOC0, CA10, CA90, dth0, m0);
fprintf('CA range: [%.1f %.1f], N=%d\n', CA(1), CA(end), numel(CA));
fprintf('Praw range: [%.2f %.2f] bar\n', min(Praw)/1e5, max(Praw)/1e5);
```

```
Section 7 baseline (octane): SOC=-43.37 degCA, CA10=-1.59, CA90=34.28, dtheta=44.85, m=2.00
CA range: [-99.0 359.0], N=459
Praw range: [0.67 67.41] bar
```

```

% Visual check of WAVE pressure
figure('Name','WAVE pressure confirmation');
plot(CA, Praw/1e5, 'LineWidth', 1.2);
xlabel('Crank Angle (degCA)'); ylabel('Pressure (bar)');
title([fuel ' - WAVE pressure']);
grid on;

```



```

theta = CA - SOC0;

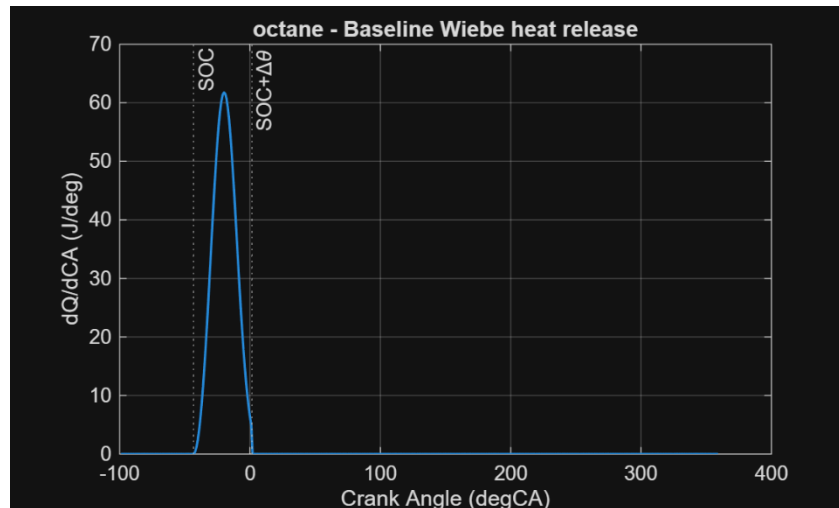
% Analytic dxb/dCA for Wiebe within [0, dth0]
dxb_dCA = zeros(size(theta));
idx = theta >= 0 & theta <= dth0;
z = theta(idx) ./ dth0;

dxb_dCA(idx) = (a0*(m0+1)/dth0) .* (z.^m0) .* exp(-a0 .* (z.^(m0+1)));

dQdCA0 = Q_total .* dxb_dCA;

```

```
figure('Name','Baseline heat release');
plot(CA, dQdCA0, 'LineWidth', 1.2);
xlabel('Crank Angle (degCA)'); ylabel('dQ/dCA (J/deg)');
title([fuel ' - Baseline Wiebe heat release']);
grid on; xline(SOC0,':','SOC'); xline(SOC0+dth0,':','SOC+\Delta\theta');
```



```
SOC_set = SOC0 + [-2 0 2];

figure('Name','SOC sensitivity: Pressure');
plot(CA, Praw/1e5, 'LineWidth', 1.2); hold on;

figure('Name','SOC sensitivity: Heat release'); hold on;

for j = 1:numel(SOC_set)
    SOCj = SOC_set(j);

    % Heat release for SOCj
    theta = CA - SOCj;
    dxb_dCA = zeros(size(theta));
    idx = theta >= 0 & theta <= dth0;
    z = theta(idx) ./ dth0;
    dxb_dCA(idx) = (a0*(m0+1)/dth0) .* (z.^m0) .* exp(-a0 .* (z.^(m0+1)));
    dQdCA = Q_total .* dxb_dCA;

    % Pressure reconstruction
    p = zeros(size(CA));
    p(1) = Praw(1);
    for i = 1:numel(CA)-1
        dCA = CA(i+1) - CA(i);
        dpdCA = ((gamma-1)/V(i))*dQdCA(i) - gamma*(p(i)/V(i))*dVdCA(i);
```

```

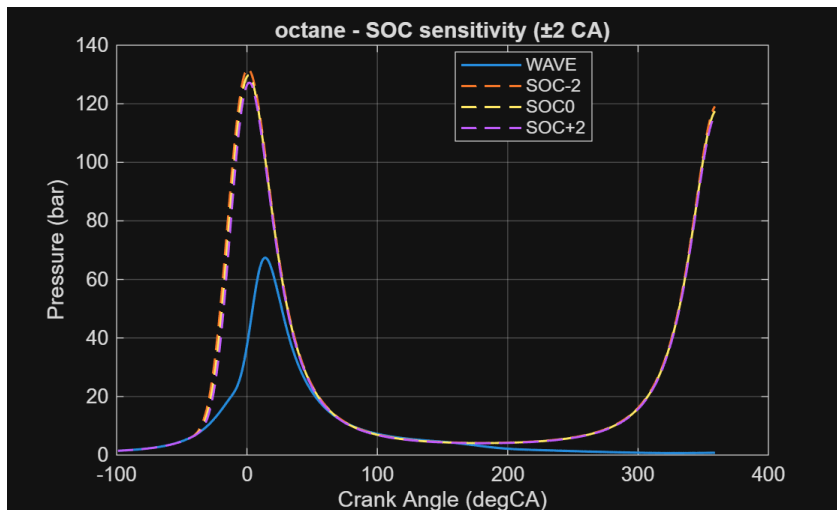
    p(i+1) = p(i) + dpdCA*dCA;
end

% Plot pressure
figure(findobj('Name','SOC sensitivity: Pressure'));
plot(CA, p/1e5, '--', 'LineWidth', 1.1);

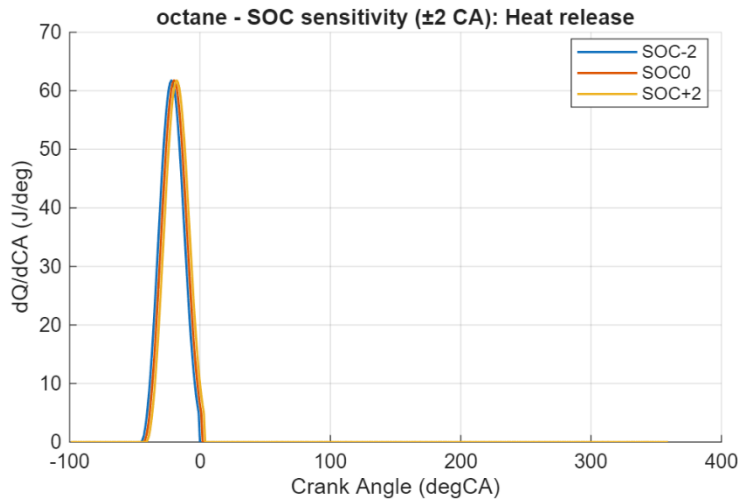
% Plot heat release
figure(findobj('Name','SOC sensitivity: Heat release'));
plot(CA, dQdCA, 'LineWidth', 1.1);
end

figure(findobj('Name','SOC sensitivity: Pressure'));
xlabel('Crank Angle (degCA)'); ylabel('Pressure (bar)');
title([fuel ' - SOC sensitivity ( $\pm 2$  CA)']);
legend('WAVE', 'SOC-2', 'SOC0', 'SOC+2', 'Location', 'best'); grid on;

```



```
figure(findobj('Name','SOC sensitivity: Heat release'));
xlabel('Crank Angle (degCA)'); ylabel('dQ/dCA (J/deg)');
title([fuel ' - SOC sensitivity ( $\pm 2$  CA): Heat release']);
legend('SOC-2','SOC0','SOC+2','Location','best'); grid on;
```



SOC has the strongest influence on pressure phasing and peak pressure, with ± 2 CA shifts causing clear changes in peak timing and magnitude. Burn duration mainly governs heat-release width and pressure rise rate, while the Wiebe shape factor m sharpens the combustion profile, affecting peak pressure with minor impact on total heat release. Overall, combustion phasing dominates pressure reconstruction accuracy.

3.8 Discussion: Model Accuracy and Limitations

The Wiebe-based reconstruction captures overall combustion trends but is limited by sensitivity to combustion phasing and simplified burn representation. Fuel effects are expressed primarily through SOC and burn duration, with propane and methanol requiring earlier phasing to align with WAVE pressure. Errors in peak pressure timing demonstrate why default WAVE Wiebe parameters struggle across fuels. Since in-cylinder temperature follows pressure phasing, these inaccuracies directly affect temperature-dependent emissions models such as thermal NO formation.

3.9 Conclusions and Recommendations

Wiebe calibration was worthwhile, as pressure reconstruction accuracy depended more on combustion phasing than generic burn-shape assumptions. Methanol deviated most from the single Wiebe representation, reflecting its rapid main burn and highlighting limitations of simplified combustion models for fast-burning, oxygenated fuels. Default WAVE Wiebe parameters provide reasonable first-order trends but are insufficient for accurate fuel-to-fuel comparison without recalibration.

For improved predictive accuracy, fuel-specific Wiebe parameters are recommended when simplified combustion models are used. Where pressure phasing and emissions prediction are critical, multi-zone or turbulence-based combustion models should replace single-zone Wiebe formulations.

4 References

Stone, R. and Macmillan, E.M. 2012. *Introduction to Internal Combustion Engines*.

McAllister, S., Fernandez-Pello, A.C. and Chen, J.-Y. 2011. *Fundamentals of Combustion Processes*. Available at: <http://www.springer.com/series/1161>.

Realis WAVE Software. 2025. Ricardo WAVE – User Manual and Fuel Property Database.

5 Appendix

Appendix A: Supporting WAVE Diagnostic Outputs

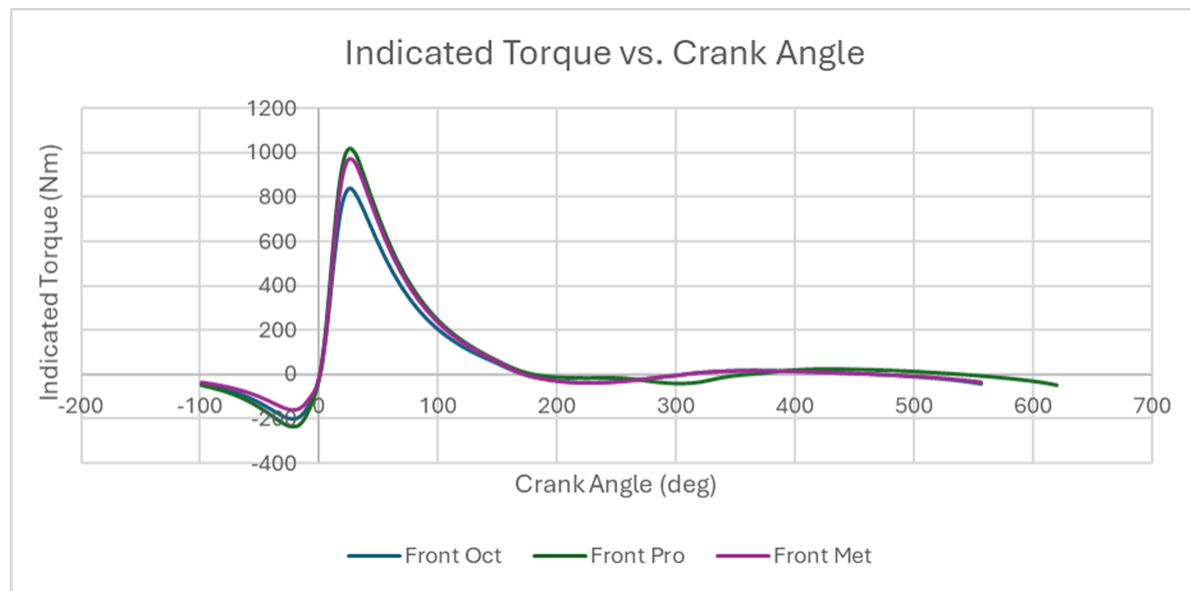


Figure A 1: Instantaneous indicated engine torque as a function of crank angle for a representative operating point (diagnostic output).

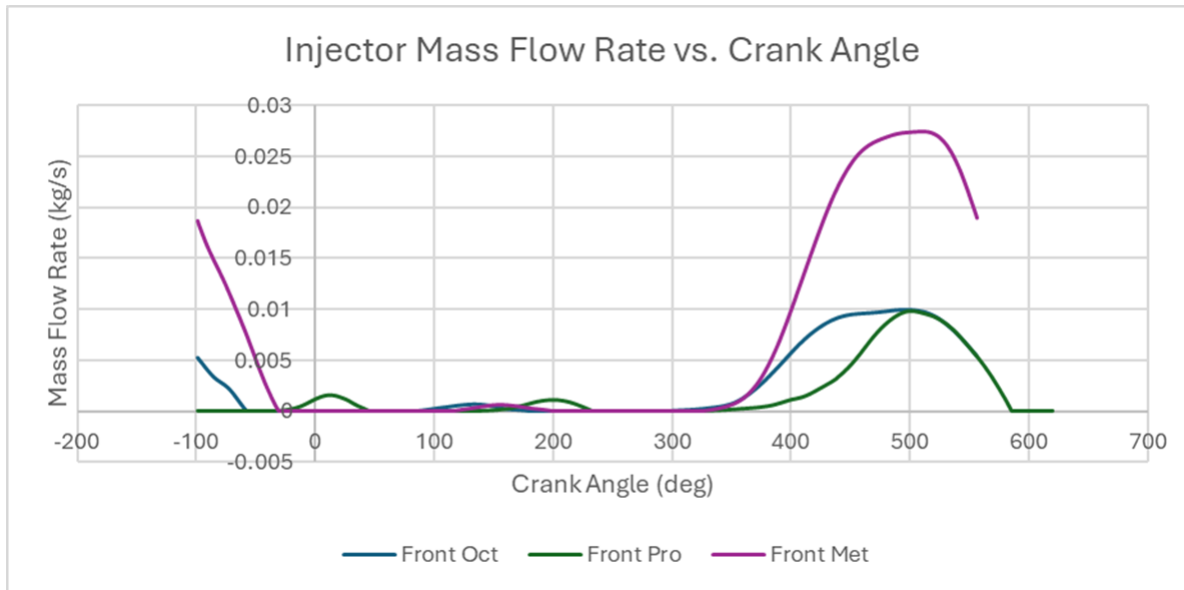


Figure A 2: Injector fuel mass flow rate as a function of crank angle for a representative operating point (diagnostic output).

Appendix B: Performance Metric Definitions

$$\text{BMEP} = \frac{4\pi T}{V_d}$$

$$\text{BSFC} = \frac{\dot{m}_{fuel}}{P_b}$$

Appendix C: Helper Functions

```
% --- Local functions for Task 1 Section 3 ---
function [ca, p] = cleanTrace(ca, p)
    % Remove non-finite values
    idx = isfinite(ca) & isfinite(p);
    ca = ca(idx);
    p = p(idx);

    % Sort by crank angle
    [ca, iSort] = sort(ca);
    p = p(iSort);

    % Remove duplicate crank angles
    [ca, iU] = unique(ca, 'stable');
    p = p(iU);
```

```
end
```

```
function [ca, p] = trimWindow(ca, p, win)
    idx = ca >= win(1) & ca <= win(2);
    ca = ca(idx);
    p = p(idx);
```

```
end
```

```
%% --- Local functions used in Task 1 Section 5 ---
```

```
function [Vd, Vc] = sweptClearance(g)
    Ap = pi*(g.Bore^2)/4;
    Vd = Ap * g.Stroke;
    Vc = Vd / (g.CR - 1);
```

```
end
```

```
function V = cylVolume(theta, g)
    R = g.Stroke / 2;
    L = g.Conrod;
    Ap = pi*(g.Bore^2) / 4;

    Vd = Ap * g.Stroke;
    Vc = Vd / (g.CR - 1);

    x = R*(1 - cos(theta)) + (L - sqrt(L^2 - (R*sin(theta)).^2));
    V = Vc + Ap .* x;
```

```
end
```