Advanced Fuel Consumption and Emission Modeling using Willans line scaling techniques for engines

Bastiaan Zuurendonk

DCT 2005.116

Traineeship report

Coach(es): Theo Hofman

Supervisor: Maarten Steinbuch

Technische Universiteit Eindhoven
Department Mechanical Engineering
Dynamics and Control Technology Group

Eindhoven, June 3, 2005

Contents

1 Introduction

This is the final report of the internship project titled "Advanced Fuel Consumption and Emission Modeling using Willans line scaling techniques for engines". The internship was performed within the framework of the NWO research programme "Impulse Drive", which focusses on the design of an hybrid vehicle drivetrain aiming at significant reduction of emissions and fuel consumption (50%-75%) on a representative drive cycle. This internship report seeks to investigate the influence of engine type, displacement, stroke, temperature and catalyst on fuel consumption and emissions.

1.1 Outline of report

The expected output of the internship was defined as:

- Analysis of constraints defined by Euronorm 4 (2005) (see § 2)
- A MATLAB tool for finding the weighted optimal operation line of an engine using quasistatic maps for fuel and emissions (see § 3)
- Analysis of displacement, temperature effects and weight factors on emissions (NO_x, HC, CO and PM) and fuel consumption (see § 4)
- A MATLAB tool for scaling engine efficiency maps to engine displacement using Willans line[1] method (see § 5)

[1] The Willans line method is a method for scaling efficiency maps using only several engine size parameters. This method is explained in more detail in § 5

2 Norm specifications

2.1 Need for emission norms?

The increased use of cars during the last 60 years has caused an enormous growth in emissions of CO, HC, NO_x and particles. Some of these emissions are poisonous and others cause the earth to warm up more and more. In order to reduce the total amount of emissions several norms were introduced. These norms define a maximum amount of emissions for new cars.

2.2 Types of norms

The most important norms are the European "Euronorms", the "EPA-norms" in North America and the Japanese norms. These norms will be discussed next. The information given is for the greater part extracted from *http://www.dieselnet.com/standards* except for the Euro 5 norm proposal.

2.2.1 Euronorms

The Euronorms are split into diesel and petrol norms. For diesel cars the limited emissions are CO, HC+NO_x and PM and since January 2000 also NO_x. For petrol cars CO and HC+NO_x are limited and since January 2000 HC and NO_x are split in separate limits. The newest Euro 4 norm has been introduced several months ago and for it's successor, Euro 5, a proposal is to combine diesel and petrol limits. The final Euro 5 will be announced spring 2005. In Table 1 Euronorm 1-4 are shown as well as the proposal for Euro 5.

Table 1: Euronorms for passenger cars in [g/km]

Tier	*Year*	*CO*	*HC*	*HC+NO_x*	*NO_x*	*PM*
Diesel						
Euro 1	1992.07	2.72	–	0.97	–	0.14
Euro 2	1996.01	1.0	–	0.7	–	0.08
Euro 3	2000.01	0.64	–	0.56	0.50	0.05
Euro 4	2005.01	0.50	–	0.30	0.25	0.025
Euro 5[2]	2010.01	1.0	0.050	–	0.08	0.0025
Petrol						
Euro 1	1992.07	2.72	–	0.97	–	–
Euro 2	1996.01	2.2	–	0.5	–	–
Euro 3	2000.01	2.30	0.20	–	0.15	–
Euro 4	2005.01	1.0	0.10	–	0.08	–
Euro 5[2]	2010.01	1.0	0.050	–	0.08	0.0025

2.2.2 EPA-norms

The EPA-norms are a bit more complex than the Euronorms. There are two sets of standards, Tier 1 and Tier 2. Tier 1 standards were phased-in beginning 1994 and fully implemented in 1997 while the Tier 2 standards were published December 1999 and phased-in beginning in 2004. The full implementation of Tier 2 will be in 2007. Both standards apply to a full vehicle life of 100.000 [mi] (= 160.934 [km]). There is only a difference between diesel and petrol for NO_x in Tier 1. For Tier 2 there is no difference between diesel and petrol. While Tier 1 limited emissions per weight category, Tier 2 applies the same standards to all weight categories. The Tier 2 standard is build-up from 8 emission levels the so called "bins". Vehicle manufacturers have to certify their cars in a certain bin.

[2] This is what is proposed for Euro 5 and not the final norm

The entire fleet sold by each manufacturer will have to meet the average NO_x standard of 0.07 [g/mi] (= 0.043 [g/km]). Due to the impossibility of making cars immediately satisfy the new standards a temporary set of bins (MDPV, 9 and 10) is also defined. They will expire after 2008. In Table 2 the EPA-norms are summarized and shown in [g/km].

Table 2: EPA-norms for passenger cars in [g/km]

Tier	*NMOG*	*THC*	*NMHC*	*CO*	*NO_x diesel*	*NO_x petrol*	*PM*	*HCHO*
1 (50,000 mi/5 years)	–	0.25	0.16	2.11	0.62	0.25	0.05	–
1 (100,000 mi/5 years)	–	–	0.19	2.61	0.78	0.37	0.06	–
2 Bin MDPV[3](120,000 mi)	0.174	–	–	4.536	0.559	0.559	0.075	0.020
2 Bin 10[3](120,000 mi)	0.078	–	–	2.113	0.249	0.249	–	0.009
2 Bin 10[3](120,000 mi)	0.097	–	–	2.610	0.373	0.373	0.050	0.011
2 Bin 9[3](120,000 mi)	0.047	–	–	2.113	0.124	0.124	–	0.009
2 Bin 9[3](120,000 mi)	0.056	–	–	2.610	0.186	0.186	0.037	0.011
2 Bin 8 (50,000 mi)	0.062	–	–	2.1	0.087	0.087	–	0.009
2 Bin 8 (120,000 mi)	0.078	–	–	2.61	0.124	0.124	0.012	0.011
2 Bin 7 (50,000 mi)	0.047	–	–	2.1	0.068	0.068	–	0.009
2 Bin 7 (120,000 mi)	0.056	–	–	2.61	0.093	0.093	0.012	0.011
2 Bin 6 (50,000 mi)	0.047	–	–	2.1	0.050	0.050	–	0.009
2 Bin 6 (120,000 mi)	0.056	–	–	2.61	0.062	0.062	0.006	0.011
2 Bin 5 (50,000 mi)	0.047	–	–	2.1	0.031	0.031	–	0.009
2 Bin 5 (120,000 mi)	0.056	–	–	2.61	0.043	0.043	0.006	0.011
2 Bin 4 (120,000 mi)	0.043	–	–	1.3	0.025	0.025	0.006	0.007
2 Bin 3 (120,000 mi)	0.034	–	–	1.3	0.019	0.019	0.006	0.007
2 Bin 2 (120,000 mi)	0.006	–	–	1.3	0.012	0.012	0.006	0.002
2 Bin 1 (120,000 mi)	0.000	–	–	0.0	0.000	0.000	0.000	0.000

2.2.3 Japanese norms

In Japan there are 2 laws which need to regulate the emissions. Since 1992 there is a "Law Concerning Special Measures to Reduce the Total Amount of Nitrogen Oxides Emitted from Motor Vehicles in Specified Areas" or in short "The Motor Vehicle NO_x Law". This law has amendment since 2001 to tighten NO_x requirements and to add PM limits. The amended rule is called "Law Concerning Special Measures to Reduce the Total Amount of Nitrogen Oxides and Particulate Matter Emitted from Motor Vehicles in Specified Areas" or in short "Automotive NO_x and PM Law". The amended rules came into force in October 2002. The Japanese laws only apply to diesel passenger cars so emissions from petrol passenger cars are not taken into account. In Table 3 the Japanese norms are shown.

[3]These bins are temporary and will expire after 2008

Table 3: Japanese norms for passenger cars in [g/km]

Vehicle Weight	*Year*	*CO mean(max)*	*HC mean(max)*	*NO_x mean(max)*	*PM mean(max)*
<1250 [kg]	1986	2.1 (2.7)	0.40 (0.62)	0.70 (0.98)	–
	1990	2.1 (2.7)	0.40 (0.62)	0.50 (0.72)	–
	1994	2.1 (2.7)	0.40 (0.62)	0.50 (0.72)	0.20 (0.34)
	1997	2.1 (2.7)	0.40 (0.62)	0.40 (0.55)	0.08 (0.14)
	2002[4]	0.63	0.12	0.28	0.052
	2005	0.63	0.024	0.14	0.013
>1250 [kg]	1986	2.1 (2.7)	0.40 (0.62)	0.90 (1.26)	–
	1992	2.1 (2.7)	0.40 (0.62)	0.60 (0.84)	–
	1994	2.1 (2.7)	0.40 (0.62)	0.60 (0.84)	0.20 (0.34)
	1998	2.1 (2.7)	0.40 (0.62)	0.40 (0.55)	0.08 (0.14)
	2002[4]	0.63	0.12	0.30	0.056
	2005	0.63	0.024	0.15	0.014

2.3 How to interpret these norms?

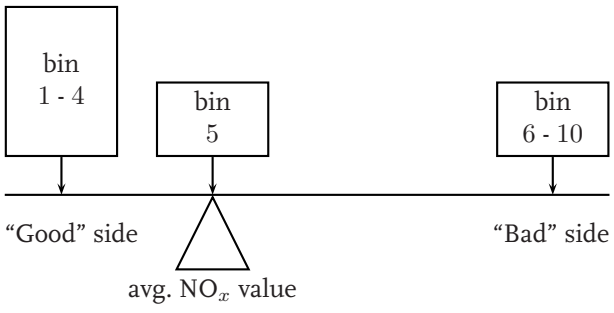
2.3.1 Euronorms

All Euronorms will be applied to new cars only. So the current car fleet will not be affected by the new rules and car owners do not have to worry about them. Car manufacturers on the contrary will have to apply to new rules from the moment they come into force. Because the emission norms apply to a very broad range of car weights they will be much easier to implement for light vehicles than heavier vehicles. As soon as car manufacturers have all weight classes apply to the new norms the car market will be determined by the demand of consumers and not by limitations in supply. So on the long term there will be no significant change in the car market.

2.3.2 EPA-norms

The EPA-norms are also only applied to new cars. So car manufacturers will be the only ones who have to deal with them. The big difference between the EPA-norms and the Euronorms is the 'bin'-structure which allows heavier cars to have more emissions. The average NO_x emission of a car manufacturers entire fleet has to be 0.07 [g/mi] (= 0.043 [g/km]) or less. In order to achieve this a car manufacturer needs to make sure he produces enough cars in the 'lower' bins. The final car market can be influenced by this. Every sold heavy car will have to be compensated for with lighter vehicles (see Figure 1). So the norm can have influence on the car weight distribution.

Figure 1: NO_x balancing for the EPA-norms



[4] 2002.10 for domestic cars, 2004.09 for imports

2.3.3 Japanese norms

The Japanese norms apply to new cars as well as to the current car fleet. Due to this, some older cars may need to be upgraded in order to comply with the new rules or replaced by newer ones. The Japanese norms are only defined for diesel cars so the petrol car market is not influenced. As a result the market could change towards more petrol cars and/or an improvement and renewal of the diesel car fleet.

3 Determination of the Optimal Operation Line

A possible step in the reduction of emissions (CO, HC, NO_x, particles, etc.) from a passenger car is to run the engine in an optimal operation point. This optimal point can be found by minimizing a cost function of fuel consumption and emissions over a combination of torque and speed values:

$$\min_{T,\omega} J = \min_{T,\omega} \sum_i w_i e_i \quad \text{with} \quad T_{min} \leq T \leq T_{WOT} \quad \text{and} \quad \omega_{e_{min}} \leq \omega \leq \omega_{e_{max}}$$

Note the difference between ω and w in above formula. In which w_i is a weighting factor for a certain kind of emission, e_i is the amount of this emission in a certain operating point, T and ω define a torque and speed range. The optimal values for T and ω are found within the torque speed envelope defined by the constraints on T and ω. The maximum torque is described by the wide-open throttle torque, T_{WOT}, which depends on the engine speed. After defining a set of weighting parameters the cost function, J, for several operating points needs to be calculated for a certain required engine power. The torque speed combination which minimizes the cost function forms the optimal operation point. The line drawn through the operation points which minimizes J forms the Optimal Operation Line (OOL).

3.1 Use of MATLAB in determination of the Optimal Operation Line

3.1.1 MATLAB Algorithm procedure

The algorithm procedure used can be summarized as follows (see Figure 2 for a graphical representation of this algorithm procedure)

- A fuel or emission map is loaded
- The data is interpolated and spread over a user-defined grid given by a number of steps in torque, n_T, and speed, n_ω (see Figure 3 for an example)
- For a certain requested power value, P^*, an isopower curve is determined within the bounds of the grid
- For all points on the isopower curve the amount of emissions is determined and a minimum can be found[5]

One can use just one emission type or fuel map for determination of the OOL or several emission maps. In this last case a weighted optimal operation line is determined. In this case also the weighting factors need to be taken into account. The algorithm procedure used in this case is as follows:

- All used emission maps are loaded
- The data is interpolated and spread over a user-defined grid
- For a certain requested power value an isopower curve is determined within the bounds of the grid
- For all points on the isopower curve the amount of emissions is determined
- For each type of emission the minimum and maximum value is determined and all values on the isopower curve are scaled from 0 to 1
- All normalized emissions are multiplied by their weighting factor and summed
- The torque and speed combination minimizing J is determined

Now the algorithm procedures are defined, they can be used for determining the influence of displacement, temperature effects and weighting factors on the OOL and weighted optimal operation line. In the next section these influences will be inspected in more detail.

[5] Note: the maximum torque curve is taken into account as well while determining the minimum

Figure 2: Graphical representation of the algorithm procedure

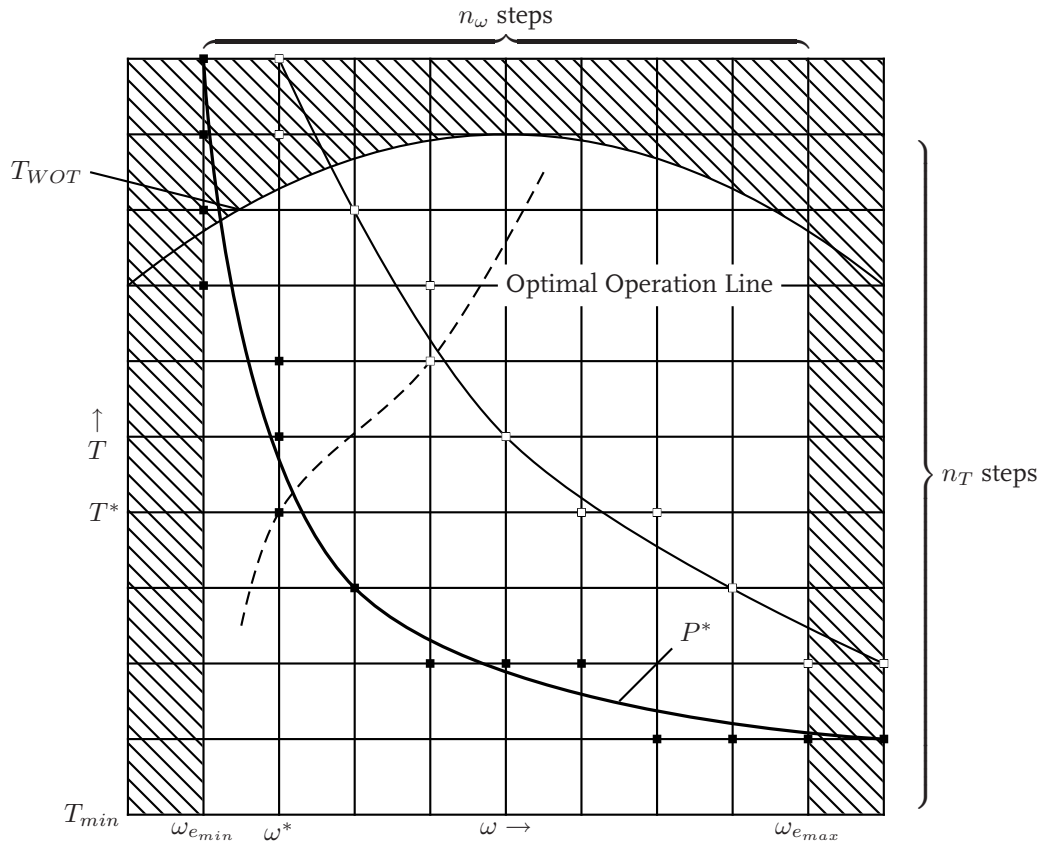
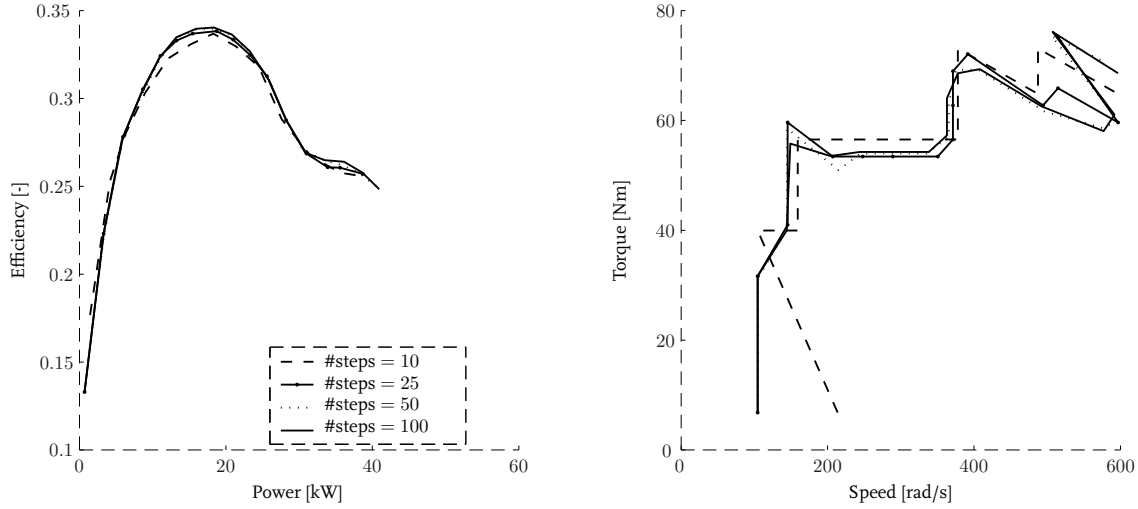


Figure 3: Influence of grid size on Optimal Operation Line



4 Influence of displacement, temperature effects and weighting factors on the Optimal Operation Line

The output of the algorithm procedure described in the previous section is a set of power values and for each power value a matching torque and speed value for which the fuel and or emissions are minimally. Each emission type has an own 'characteristic' shape of OOL. In this section the influence of displacement, temperature effects and weighting factors on the amount of emissions and this 'characteristic' shape will be discussed in more detail.

In this section not only the emissions CO, HC, NO_x and PM are considered, but also efficiency, η. Because efficiency and fuel use, $\dot{m}_f$, are related through:

$$\eta = \frac{1}{h_{lhv}\dot{m}_f} \quad [\text{-}]$$

it is easy to see we can interpret the inverse of efficiency as a kind of emission as well. Minimizing $\frac{1}{\eta}$ will after all produce the same OOL as when minimizing $\dot{m}_f$.

4.1 Used engine maps

Determination of the OOL was carried out in MATLAB using static engine maps from ADVISOR 2002. These engine maps contain measured emission data for several engines in several stationary operating points. In Table 4 the used engines are shown including some specifications and the .m-files in which they can be found.

Table 4: Some specifications of the used engines

.m-file	*Fueltype*	*Engine*	*Displ. [L]*	*Torque [Nm]* T_{min} - T_{WOT}	*Speed [rad/s]* $\omega_{e_{min}}$ - $\omega_{e_{max}}$
FC_SI41_emis	Gasoline	Geo	1.0	7 - 81	105 - 597
FC_SI63_emis	Gasoline	Saturn	1.9	0 - 136	73 - 576
FC_SI102_emis	Gasoline	Dodge Caravan	3.0	27 - 217	129 - 511
FC_CI60_emis	Diesel	Mercedes	1.7	0 - 183	126 - 440
FC_CI67_emis	Diesel	VW Turbo Diesel	1.9	0 - 224	84 - 461
FC_CI92_emis	Diesel	Mercedes	2.2	10 - 300	131 - 445

These engine maps were used to create Figure 4, 5 and 6. These figures will be used for a graphical analysis of the influence of displacement and temperature effects on the OOL. See Table 5 for used line styles.

Table 5: Line styles used in Figure 4, 5 and 6

Engine	*Line style*
Gasoline 1.0	dotted
Gasoline 1.9	dashed
Gasoline 3.0	solid
Diesel 1.7	dotted
Diesel 1.9	dashed
Diesel 2.2	solid

Figure 4: OOL for several petrol engines for several temperatures

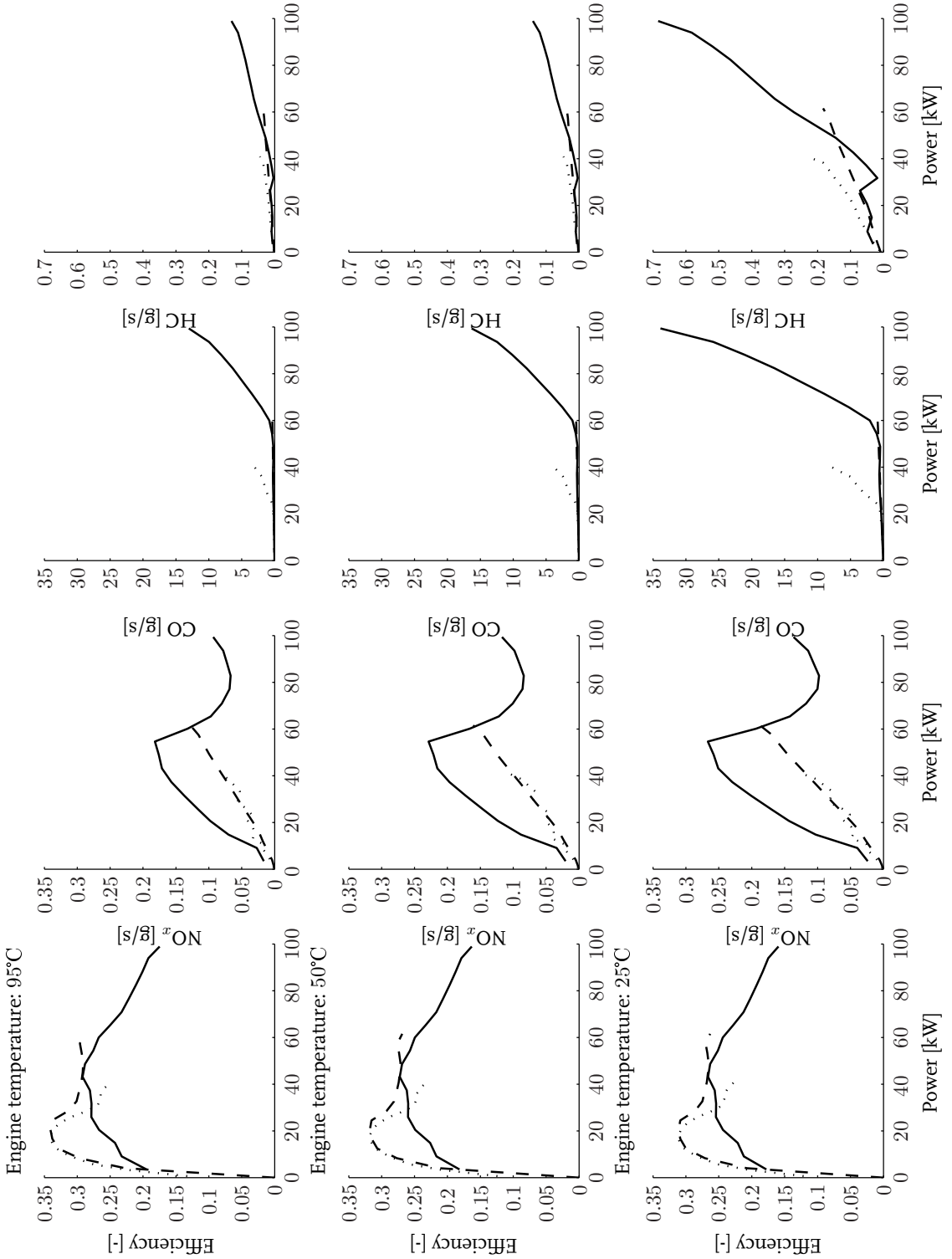


Figure 5: OOL for several diesel engines for several temperatures

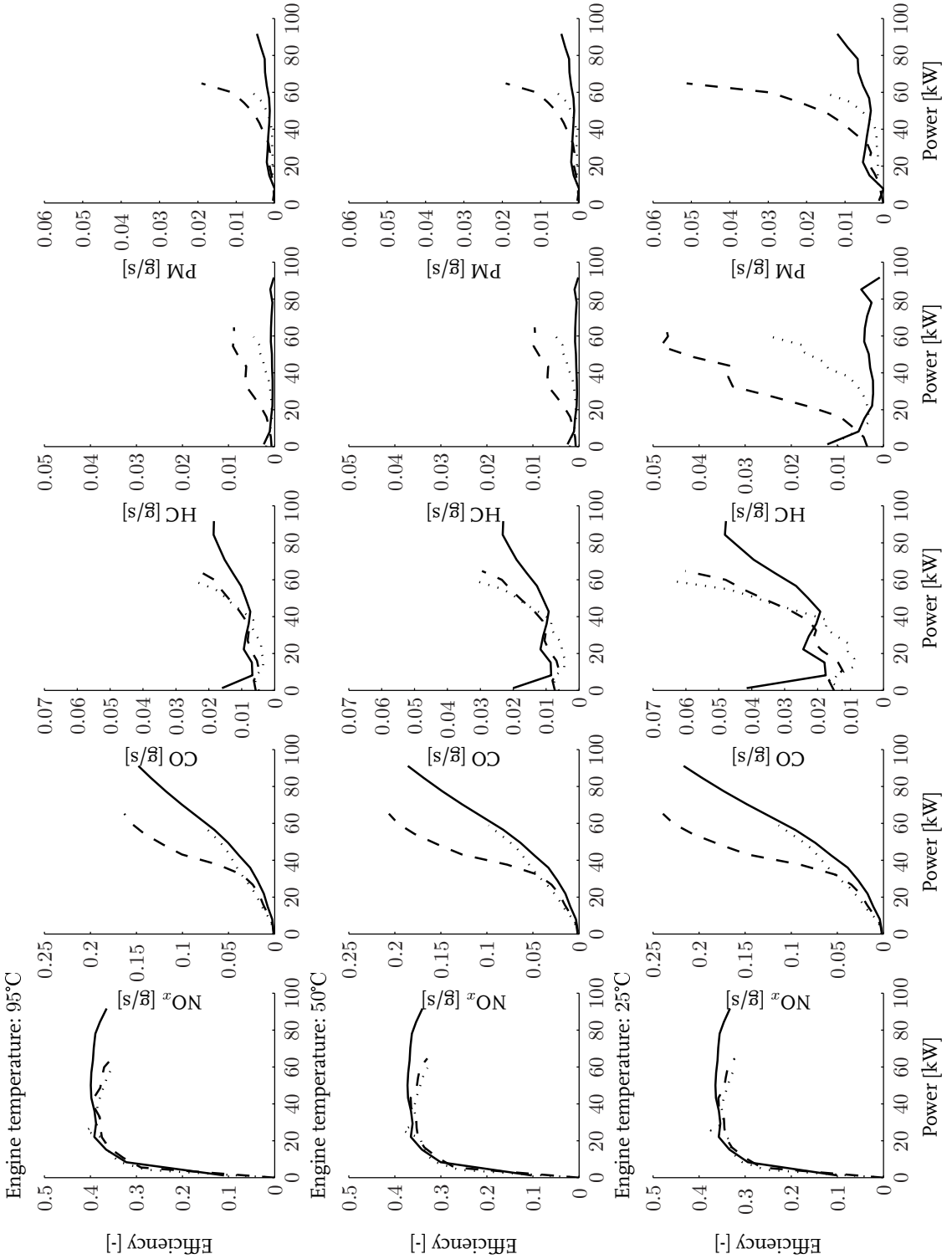


Figure 6: OOL as function of speed

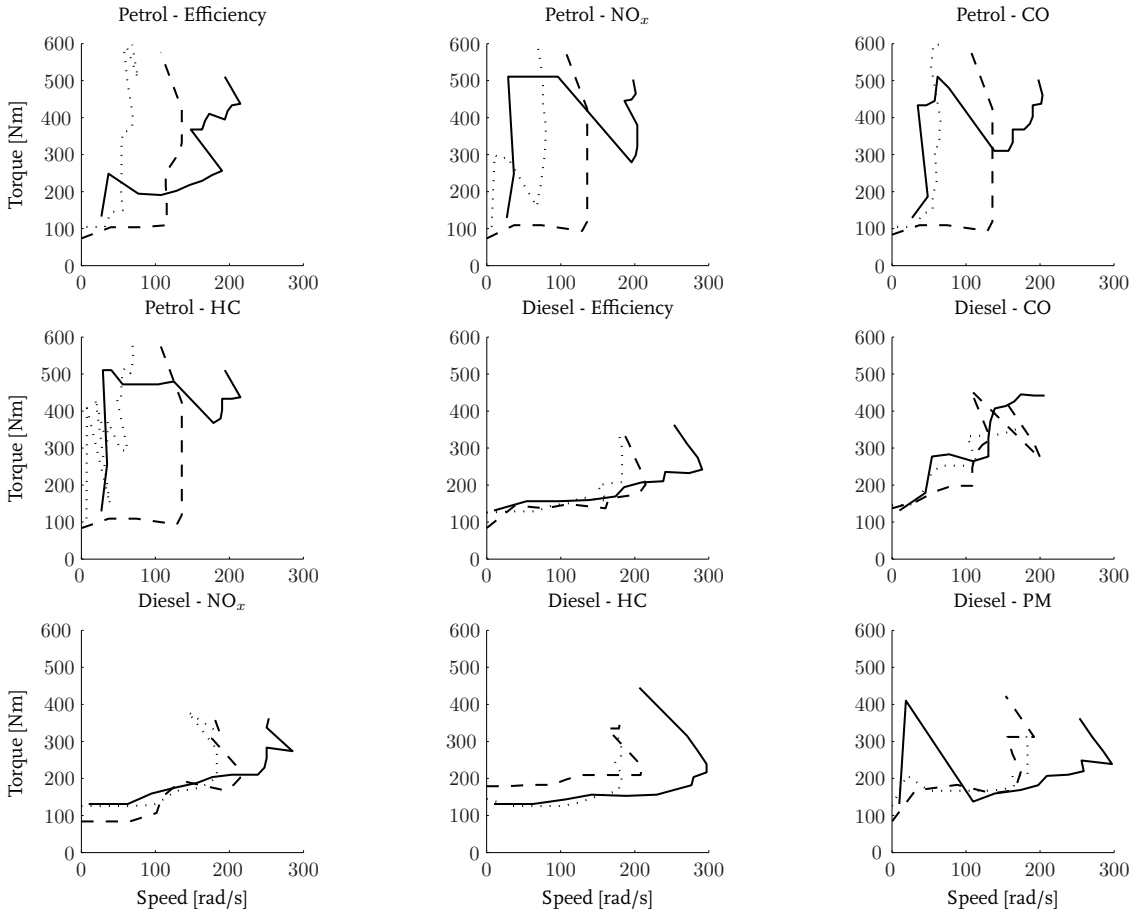
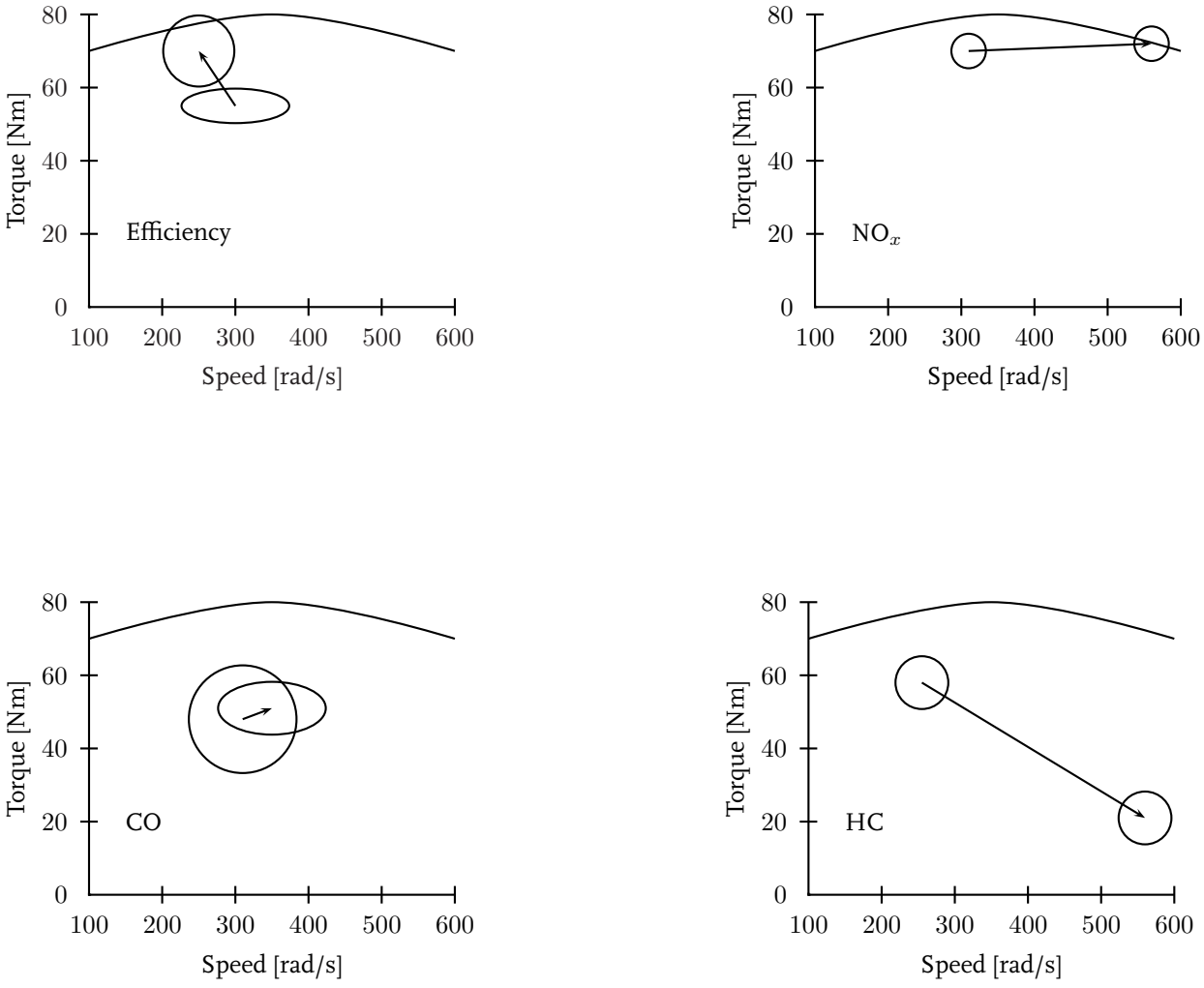


Figure 7: Global influence of displacement on emission islands



4.2 Influence of displacement

In Figure 7 a global sketch is given for the location of 'low emission islands' in petrol emission maps as function of displacement. The shifting of these islands when changing from small to large displacement is indicated by ellipses and arrows. In fact this figure is a *scaled* summary of Figure 6. From Figure 7 it can be easily seen that:

- The efficiency and CO islands do hardly move. So displacement will not have a large effect on the OOL determined with weighting factors efficiency and CO
- The islands for NO_x and HC on the contrary show a very large shifting. The OOL will change a lot as a function of displacement if only NO_x and/or HC emissions are taken into account

One should realize that these conclusions were drawn from a very small number of available engine maps. So the results are not very accurate. They should be used only as a very global indication of what happens for displacement changes and certainly not be used as if they were 100% truth.

4.3 Influence of temperature

While looking at emissions it is very important to take into account the temperature of an engine. Cold engines work less efficient en produce more emission due to thermal efficiency, viscous friction losses, etc. Since emission maps are not available for every temperature an empirical cold-to-hot fit function to estimate emissions for non-measured temperatures is used. The cold-to-hot ratio is used as a multiplication factor for "hot" maps to create "cold" maps. This cold-to-hot ratio is determined as follows:

$$r = 1 + c\lambda^e \quad [\text{-}]$$

In which r is the cold-to-hot-ratio, c and e a coefficient and exponent which are emission specific and λ is a normalized temperature defined as:

$$\lambda = \frac{T_s - T_c}{T_s - 20} \quad [\text{-}]$$

In which T_s [°C] is the set thermostat temperature and T_c [°C] the temperature of the coolant fluid. So T_s is the temperature that is varied while T_c is held constant. The values for c and e have been determined empirically. The updated values from the ADVISOR 2002 documentation have been used (*Engine Cold-Hot Comparison.htm*). The values are summarized in Table 6.

Table 6: Fit coefficients and exponents for determination of cold-to-hot ratio

Emission type	*Coefficient [-]*	*Exponent [-]*
Fuel	0.10	0.65
HC	8.05	9.46
CO	2.13	4.26
NO_x	0.51	1.37
PM	5.14	17.04

These values were used to create the 50 °C and 25 °C maps from the 95 °C maps in Figure 4 and 5. Using these figures again the following conclusions can be drawn:

- Temperature effects have no significant influence on efficiency in the considered region for petrol as well as for diesel (about $+10\%$)
- NO_x emissions are influenced slightly by temperature effects (about $+45\%$) and are advised to be taken into account in calculations
- CO emissions show a very clear influence on temperature (about $+160\%$) and need to be taken into account in calculations
- Temperature effects on HC emissions are very strong (about $+400\%$) and need to be taken into account in calculations

4.4 Weighting factors

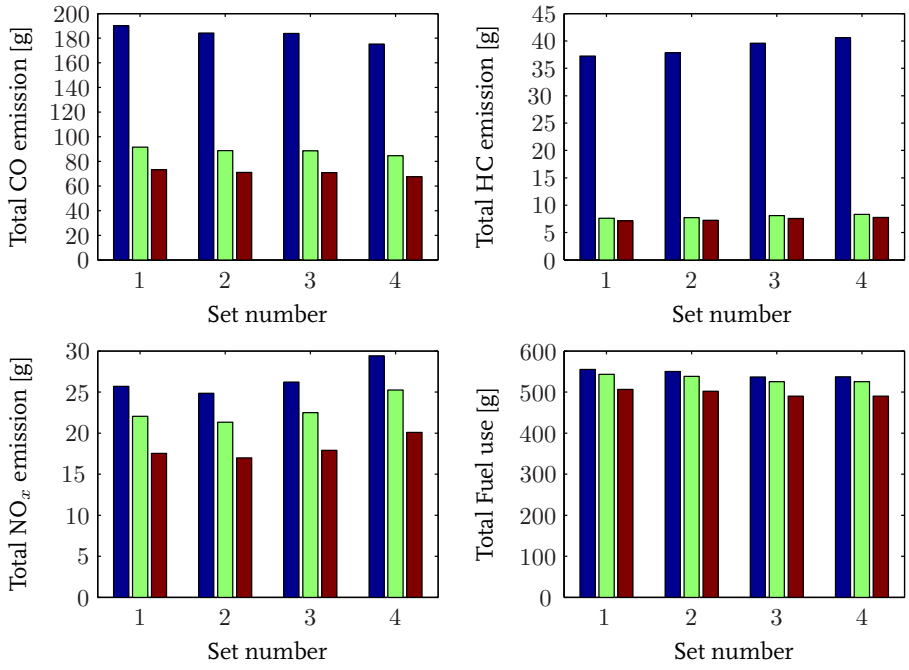
For determination of a weighted optimal operation line one can use many different sets of weighting factors. The problem is however how to determine the optimal set of weighting factors minimizing the amount of emissions. Several suggestions for these sets can be found in the ADVISOR 2002 documentation. In Table 7 some of these combinations are listed. The 3^{rd} set is the set which is used as a standard in the written MATLAB tool.

Table 7: Several weighting factor combinations for petrol engines

Set	*Efficiency*	*CO*	*HC*	NO_x	*PM*
1	1	0	1	0	0
2	1	1	1	1	0
3	0.7	0.1	0.1	0.3	0
4	1	0	0	0	0

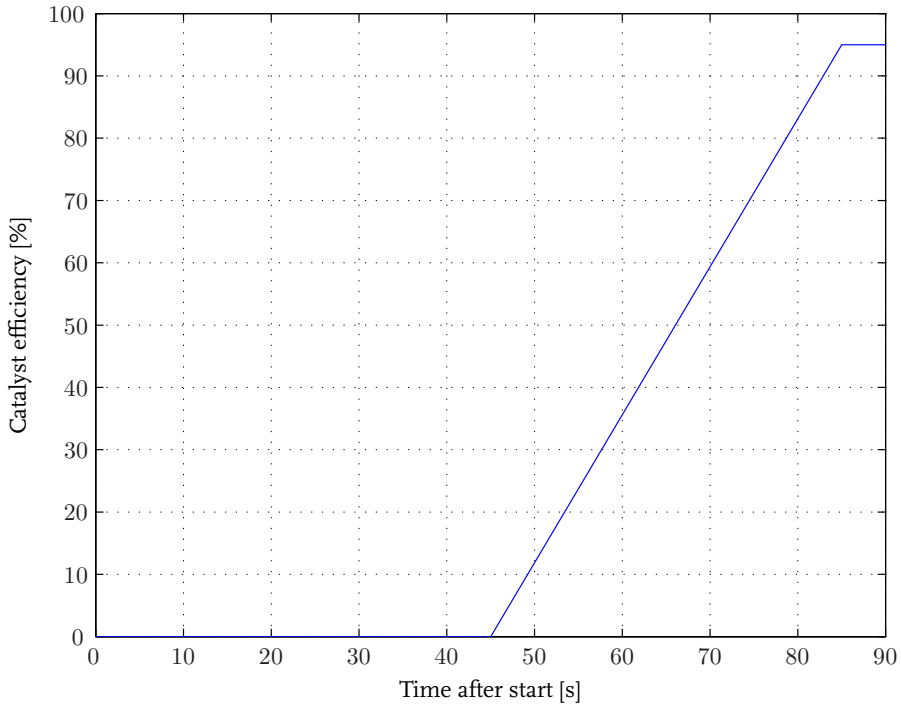
To check what the influence is of these weighting factors on the operating points on the OOL a virtual test drive is made. Input for this test drive is a column with requested power values on the ECE + EUDC drive cycle. For every second during this test a certain amount of power is needed. The requested power is looked-up in the determined OOL and so a torque/speed profile can be generated for this cycle. The total amount of emissions on this test drive is what we are interested in. In Figure 8 the results for the test drives are shown. The blue, green and red line indicate respectively the 25°C, 50°C and 95°C situation. It can be easily seen that there are no enormous differences between the various suggested weighting factor sets. So the final choice will depend on what is considered important.

Figure 8: Results for Geo 1.0 [L] engine on traject



On the drive cycle the OOL is used to determine the optimal operating strategy in order to minimize emissions. One would expect that using the OOL would give results which comply with the norms introduced in § 2. If we take for example the total emission of CO under warm conditions (95°C) and divide it by the total length of the test drive (10.81 [km]) we find an emission of 6.8 [g/km]. Comparing this with the Euro 4 norm of 1.0 [g/km] we could conclude something went terribly wrong or that the norms are very strict, but incorporation of catalyst effects in this situation will show a surprising effect. A fully warmed-up catalyst has an efficiency of about 95% (see Figure 9). So the total emission on this track will be about 0.34 [g/km] and will be satisfactory for the Euro 4 norm.

Figure 9: Estimate for catalyst efficiency



5 Scaling using Willans line method

If during the design proces of a car the optimal size of the engine needs to be determined one can use available engine maps to choose from. The problem is however the limited number of available engine maps. If one really needs to find the optimal size a scaling method for engine maps can be used. One method for scaling engine maps is the Willans line method. This method suggests an affine relationship between available energy (the chemical energy in the fuel) and the useful energy (the energy which is present at the engines output axis). We can write this as:

$$W_{out} = eW_{in} - W_{loss}$$

In which e [-] is the energy conversion efficiency in the engine. In practice this line is not straight at all due to the strong nonlinear behavior of e and W_{loss}. We can however write these variables as functions of speed and torque as we'll see next. We start by defining:

$$\omega \cdot T_e = e \cdot \dot{m} \cdot H_{LV} - W_{loss}$$

Dividing on both sides by ω this results in:

$$T_e = e \cdot H_{LV} \cdot \frac{\dot{m}}{\omega} - T_{loss} = e \cdot T_a - T_{loss}$$

In which T_a [Nm] is the available torque and T_{loss} [Nm] are the inner losses of the engine. Next the engine size parameters are introduced for making above equation directly dependent on these size parameters. V_d [m^3], is the engine displacement and S [m] is the piston stroke. Now we can write:

$$\begin{aligned} p_{me} &= \frac{4 \cdot \pi}{V_d} \cdot T_e \\ p_{ma} &= \frac{4 \cdot \pi \cdot H_{LV}}{V_d} \cdot \frac{\dot{m}}{\omega} \\ c_m &= \frac{S}{\pi} \cdot \omega \end{aligned}$$

In which p_{me} [Pa], p_{ma} [Pa] and c_m [m/s] are respectively the mean effective pressure (the engines ability to produce mechanical work), the available mean effective pressure (the maximum mean effective pressure which could be produced if $\eta = 1$) and the mean piston speed (measure for the engines operating speed). Now we can rewrite the equation for T_e as:

$$p_{me} = e \cdot p_{ma} - p_{mloss}$$

In which the mean effective pressure loss, p_{mloss} is defined as:

$$p_{mloss} = \frac{4 \cdot \pi}{V_d} \cdot T_{loss}$$

Finally from literature the following suggestion is made for parameterization:

$$\begin{aligned} p_{me} &= [e_0(c_m) - e_1(c_m) \cdot p_{ma}] \cdot p_{ma} - p_{mloss}(c_m) \\ e_0(c_m) &= e_{00} + e_{01} \cdot c_m + e_{02} \cdot c_m^2 \\ e_1(c_m) &= e_{10} + e_{11} \cdot c_m \\ p_{mloss}(c_m) &= p_{mloss0} + p_{mloss2} \cdot c_m^2 \end{aligned}$$

Now a least squares fit on real data can be done in order to acquire the introduced parameters. This leat squares fit can be formulated as:

$$\min_{e_{00},e_{01},e_{02},e_{10},e_{11},p_{mloss0},p_{mloss2}} \sum_{i=1}^{n} ([e_0(c_{m,i}) - e_1(c_{m,i}) \cdot p_{ma,i}] \cdot p_{ma,i} - p_{mloss}(c_{m,i}) - p_{me,i})^2$$

This minimum is found by setting the partial derivatives with respect to e_{00}, e_{01}, e_{02}, e_{10}, e_{11}, p_{mloss0} and p_{mloss2} equal to zero. This yields the following equations:

$$\begin{aligned}
0 &= \sum_{i=1}^{n}(R \cdot p_{ma,i}) \\
0 &= \sum_{i=1}^{n}(R \cdot c_{m,i} \cdot p_{ma,i}) \\
0 &= \sum_{i=1}^{n}(R \cdot c_{m,i}^2 \cdot p_{ma,i}) \\
0 &= \sum_{i=1}^{n}(R \cdot p_{ma,i}^2) \\
0 &= \sum_{i=1}^{n}(R \cdot c_{m,i} \cdot p_{ma,i}^2) \\
0 &= \sum_{i=1}^{n}(R) \\
0 &= \sum_{i=1}^{n}(R \cdot c_{m,i}^2)
\end{aligned}$$

In which R is shorthand notation for:

$$R = 2 \cdot ([e_0(c_{m,i}) - e_1(c_{m,i}) \cdot p_{ma,i}] \cdot p_{ma,i} - p_{mloss}(c_{m,i}) - p_{me,i})$$

The seven equations above are finally rewritten in terms of e_{00}, e_{01}, e_{02}, e_{10}, e_{11}, p_{mloss0} and p_{mloss2} and all summations are split up. Then MATLAB is used to find an explicit expression in $c_{m,i}$, $p_{ma,i}$ and $p_{me,i}$ for all parameters. Because the resulting expressions cover several pages each they are not printed in this report.

Using these expressions a set of parameters was created from VW 1.9 [L] TDI engine[6] data as an example. Cropping the data to a smaller region first and then fitting results in the following parameter values:

Table 8: Willans line parameters found from VW 1.9 [L] TDI engine

Parameter	*Value*
e_{00}	0.3123
e_{01}	0.0042
e_{02}	0.0012
e_{10}	$-3.3575 \cdot 10^{-8}$
e_{11}	$4.5728 \cdot 10^{-9}$
p_{mloss0}	$-1.3138 \cdot 10^{4}$
p_{mloss2}	$2.7774 \cdot 10^{3}$

In Figure 10 a comparison is made between the data found in ADVISOR 2002 and the model. The ADVISOR 2002 data and model data correspond respectively to the circles and lines. Next the model is scaled up to an Mercedes 2.2 [L] engine[7] and compared again with the ADVISOR 2002 data. The results can be seen in Figure 11.

The scaling procedure consists of the following steps:

- The stroke, S_{scaled}, and displacement, $V_{d,scaled}$, of the scaled engine are chosen.

[6] This engine has a 95.5 [mm] stroke.
[7] This engine has a 88.4 [mm] stroke.

Figure 10: Comparison of the calculated model to the published VW 1.9 [L] TDI engine data

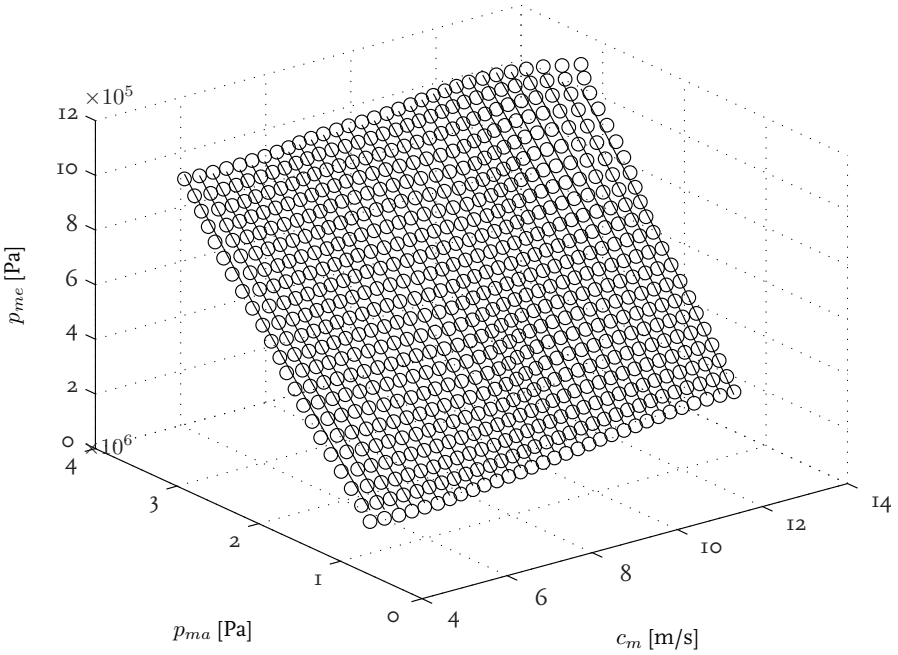
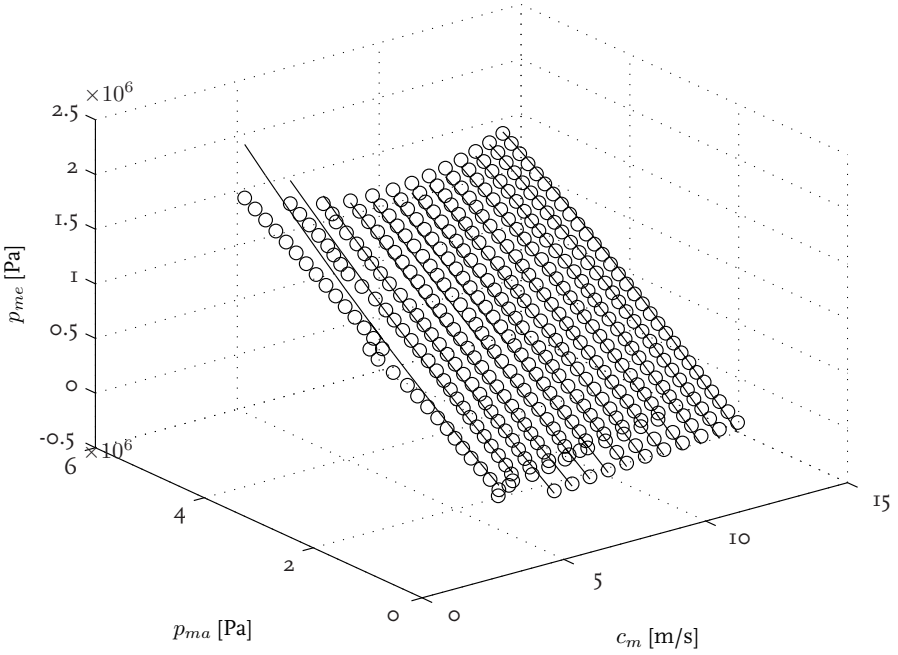


Figure 11: Comparison of the scaled up model to the published Mercedes 2.2 [L] engine data



- The mean piston speed and available mean effective pressure are scaled with:

$$c_{m,scaled} = c_{m,model} \cdot \frac{S_{scaled}}{S_{model}}$$

$$p_{ma,scaled} = p_{ma,model} \cdot \frac{V_{d,model}}{V_{d,scaled}}$$

- The mean effective pressure for the scaled engine is calculated using the previously calculated model parameters:

$$p_{me,scaled} = [e_0(c_{m,scaled}) - e_1(c_{m,scaled}) \cdot p_{ma,scaled}] \cdot p_{ma,scaled} - p_{mloss}(c_{m,scaled})$$

- The torque and speed of the scaled engine are determined:

$$T_{e,scaled} = p_{me,scaled} \cdot \frac{V_{d,scaled}}{4 \cdot \pi}$$

$$\omega_{scaled} = c_{m,scaled} \cdot \frac{\pi}{S_{scaled}}$$

- The scaled fuel map is finally calculated with:

$$\dot{m}_{scaled} = p_{ma,scaled} \cdot \omega_{scaled} \cdot \frac{V_{d,scaled}}{4 \cdot \pi \cdot H_{LV}}$$

So the final result of the scaling procedure is a fuel map again. Which can be used to determine for example the OOL of the scaled engine.

6 Conclusions

Throughout this report several subjects were discussed. First of all in § 2 the constraints defined by the Euronorms, EPA-norms and Japanese norms were investigated. The most important norm for use in Europe at this moment, Euronorm 4, can be summarized as:

> The maximum allowed total amount of CO, HC+NO_x, NO_x and PM emissions for new diesel passenger cars are respectively 0.50, 0.30, 0.25 and 0.025 [g/km]. For new petrol passenger cars the maximum allowed total amounts for CO, HC and NO_x are respectively 1.0, 0.10 and 0.08 [g/km].

In § 3 a MATLAB algorithm procedure was introduced for finding the OOL of an engine. This algorithm procedure was implemented in the MATLAB files `eline.m` and `weline.m`.

In § 4 the influences of displacement, temperature effects and weight factors on emissions were investigated. The following main conclusions were drawn:

- No clear relationship between displacement and the amount of emissions could be found. Only a slight trend could be found in some cases. This trend showed a shifting of the islands in the emission maps. Due to the small number of available emission maps these trends are very uncertain and one should avoid using them. The trends can be summarized as follows:
  - The efficiency and CO islands do hardly move. So displacement will not have a large effect on the OOL determined with weighting factors efficiency and CO
  - The islands for NO_x and HC on the contrary show a very large shifting. The OOL will change a lot as a function of displacement if only NO_x and/or HC emissions are taken into account
- The influence of temperature effects on fuel use is very small so they do not have to be taken into account as long as the cold (worst case) conditions are used. NO_x and especially CO and HC emissions show a clear increase for low temperatures in comparison with hight temperatures so thermal effects should certainly be taken into account for these emissions
- Several sets of weighting factors were investigated but none of them showed a very large deviation from the rest. So all introduced weighting factor combinations can be used without having to worry about enormous increase of emissions. So one can choose a set of weighting factors which optimizes for a special kind of emission as well as combination of emissions
- Scaling engine maps using Willans line method was investigated in § 5. A generic MATLAB tool was not written due to a lack of time during the internship, but the techniques introduced will enable the reader to write one such tool for himself. The theory presented was used to scale up a VW 1.9 [L] engine to a 2.2 [L] engine and compared with actual data of a Mercedes 2.2 [L] engine. The predicted behaviour of this 2.2 [L] engine compared pretty well with the actual data

A Used abbreviations

- *CO* (Carbon monoxide), poisonous gas.
- *ECE* (City cycle - European Commission for Europe), urban drive cycle with maximum speed 50 [km/h].
- *EPA* (Environmental Protection Agency), government ministry in the USA responsible for protection of the environment.
- *EUDC* (Extra urban drive cycle), suburban drive cycle with maximum speed 120 [km/h].
- *HC* (Hydrocarbons), emission which forms together with NO_x smog under influence of sunlight.
- *HCHO* (Formaldehyde), poisonous gas which can be turned into CO and other gases.
- *MDPV* (Medium-duty passenger vehicles), class of vehicles which are heavier than normal and used for personal transportation. For example passenger vans.
- *NMHC* (Non-methane hydrocarbons), special group of HC.
- *NMOG* (Non-methane organic gases), potential ozone forming gas.
- *NO_x* (Nitrogen oxides), emission which forms together with HC smog under influence of sunlight. Consists of NO and NO_2.
- *NWO* (Nederlandse Organisatie voor Wetenschappelijk Onderzoek), Dutch organization for scientific research.
- *OOL* (Optimal Operation Line), Line which shows the optimal (for 1 emission type) operating point of an engine as function of the requested power.
- *PM* (Diesel particle matter), small particles which cause heart- and lung diseases.
- *THC* (Total hydrocarbon emissions), all HC emissions together including NMHC.
- *WOOL* (Weighted Optimal Operation Line), Line which shows the optimal (for a weighted combination of emission types) operating point of an engine as function of the requested power.
- *WOT* (Wide open throttle), indicates maximum torque available in wide open throttle situation.

B MATLAB: eline.m

```
function eline_dat = eline(inp_fuelc,num_int_pnt,num_powr_val)
% ELINE Determine the e-line of a fuelconverter
%   ELINE('inp_fuelc') evaluates 'inp_fuelc' which is the
%   name of an Advisor fuel-converter datafile. For example
%   'FC_CI67_emis' or 'FC_CI60_emis'.
%
%   ELINE('inp_fuelc', num_int_pnt) evaluates 'inp_fuelc' and
%   uses num_int_pnt points for interpolation of the powermap.
%   If omitted a standard value of 100 is used.
%
%   ELINE('inp_fuelc', num_int_pnt, num_powr_val) evaluates
%   'inp_fuelc' using num_int_points for interpolation and
%   finds num_powr_val values of the eline.
%   Note: num_int_pnt > num_powr_val
%   If num_powr_val is omitted a standard value of 20 is used
%
%   ELINE_DAT = ELINE('inp_fuelc') contains several fields;
%   - .eff contains the efficiency data
%   - .co contains the CO data
%   - .hc contains the HC data
%   - .nox contains the NOx data
%   - .pm contains the PM data
%
%   Each of these has 3 subfields:
%   - .val contains the values on the powercurve
%   - .w contains values for w on the powercurve
%   - .T contains values for T on the powercurve
%
%   For example plot(ELINE_DAT.co.w, ELINE_DAT.co.T) plots the e-line for co
%
%   Created on: 16/01/05
%   By: Bastiaan Zuurendonk, b.p.zuurendonk@student.tue.nl
%
%   Revision History
%   ----------------
%   18/03/05 Solved problem with use of '~=' and matrix dimensions

% Check for good input (string must begin with 'FC_' and
% won't be > 20 characters)
if upper(inp_fuelc(1:3))=='FC_' & length(inp_fuelc) < 21

    % Load the file
    eval(inp_fuelc);

    % Determine several meshgrids for interpolation
    trq_min = min(fc_map_trq);
    trq_max = max(fc_map_trq);
    spd_min = min(fc_map_spd);
    spd_max = max(fc_map_spd);
    if nargin > 2 & num_int_pnt < num_powr_val, ...
            error('num_int_pnt < num_powr_val'); end
    if nargin < 2, num_int_pnt = 100; end % Standard value num_int_pnt
```

```
trq_rough = fc_map_trq;
spd_rough = fc_map_spd;
[T_rough,w_rough] = meshgrid(trq_rough,spd_rough);
trq_fine = [trq_min:(trq_max-trq_min)/(num_int_pnt-1):trq_max];
spd_fine = [spd_min:(spd_max-spd_min)/(num_int_pnt-1):spd_max];
[T_fine,w_fine] = meshgrid(trq_fine,spd_fine);

% Determine the power map (kW) for the rough grid
fc_map_kW = T_rough.*w_rough/1000;
fc_map_min = min(min(fc_map_kW));
fc_map_max = max(max(fc_map_kW));

% Determine the efficiency map on the rough grid
fc_eff_map = fc_map_kW*1000./(fc_fuel_map*fc_fuel_lhv);

% Interpolate the data on the fine grid
fc_eff_map_fine = griddata(T_rough,w_rough,fc_eff_map,T_fine,w_fine);
fc_co_map_fine = griddata(T_rough,w_rough,fc_co_map,T_fine,w_fine);
fc_hc_map_fine = griddata(T_rough,w_rough,fc_hc_map,T_fine,w_fine);
fc_nox_map_fine = griddata(T_rough,w_rough,fc_nox_map,T_fine,w_fine);
fc_pm_map_fine = griddata(T_rough,w_rough,fc_pm_map,T_fine,w_fine);

% Determine/interpolate the maximum torque values and create
% a matrix with possible and impossible values
fc_max_trq_fine = interp1(spd_rough,fc_max_trq,spd_fine);
fc_max_trq_map_fine = fc_max_trq_fine'*ones(1,num_int_pnt);
[mtmx,mtmy] = find(T_fine>fc_max_trq_map_fine);
mtm = ones(num_int_pnt);
for i = 1:length(mtmx)

    mtm(mtmx(i),mtmy(i)) = NaN;

end

% Find the optimal point for several powervalues
if nargin < 3, num_powr_val = 20; end % Standard value num_powr_val
for emis_type = {'eff' 'co' 'hc' 'nox' 'pm'}

    % Initialize a data structure for output
    eval(['eline_dat.' emis_type{1} '.val = [];']);
    eval(['eline_dat.' emis_type{1} '.w = [];']);
    eval(['eline_dat.' emis_type{1} '.T = [];']);

end

% Loop from the smallest to the largest possible powervalue
for powr_val = [fc_map_min:(fc_map_max-fc_map_min)/(num_powr_val-1):fc_map_max]

    % Powercurve returns a matrix with ones and zeros corresponding
    % to points on the powercurve
    ipm = powercurve(spd_fine,trq_fine,powr_val*1000);

    % Compute the efficiency first (a maximum is needed in this case)
    eff_ipm = mtm.*ipm.*fc_eff_map_fine;
```

```
        eff_max = max(max(eff_ipm));
        eline_dat.eff.val = [eline_dat.eff.val eff_max];
        [eline_w_index,eline_T_index] = find(eff_ipm==eff_max);
        if length(eline_w_index) ~= 0

            eline_dat.eff.w = [eline_dat.eff.w spd_fine(eline_w_index(1))];
            eline_dat.eff.T = [eline_dat.eff.T trq_fine(eline_T_index(1))];

        end
        % Compute the emissions next (a minimum is needed in this case)
        for emis_type = {'co' 'hc' 'nox' 'pm'}

            eval([emis_type{1} '_ipm = mtm.*ipm.*fc_' emis_type{1} ...
                    '_map_fine;']);
            eval([emis_type{1} '_min = min(min(' emis_type{1} ...
                    '_ipm));']);
            eval(['eline_dat.' emis_type{1} '.val = [eline_dat.' ...
                    emis_type{1} '.val ' emis_type{1} '_min];']);
            eval(['[eline_w_index,eline_T_index] = find(' emis_type{1} ...
                    '_ipm==' emis_type{1} '_min);']);
            if length(eline_w_index) ~= 0

                eval(['eline_dat.' emis_type{1} '.w = [eline_dat.' ...
                        emis_type{1} '.w spd_fine(eline_w_index(1))];']);
                eval(['eline_dat.' emis_type{1} '.T = [eline_dat.' ...
                        emis_type{1} '.T trq_fine(eline_T_index(1))];']);

            end

        end

    end

% Produce some error messages in case of incorrect inputs
elseif upper(inp_fuelc(1:3))~='FC_'

    error([inp_fuelc ' does not start with ''FC_'''])

elseif length(inp_fuelc) >= 21

    error([inp_fuelc ' is >= 21 characters and probably not a fuelconverter'])

end
```

C MATLAB: weline.m

```
function weline_dat = weline(inp_fuelc,weights,num_int_pnt,num_powr_val)
% WELINE Determine the weighted e-line of a fuelconverter
%   WELINE('inp_fuelc') evaluates 'inp_fuelc' which is the
%   name of an Advisor fuel-converter datafile. For example
%   'FC_CI67_emis' or 'FC_CI60_emis'. The output gives the
%   e-line weighted by the following standard weights:
%
%     Efficiency 0.7
%     CO         0.1
%     HC         0.1
%     NOx        0.3
%     PM         0
%
%   These values are adapted from the Advisor documentation:
%     %Advisor root%/documentation/fuzzyemis.htm
%
%   WELINE('inp_fuelc', weights) evaluates 'inp_fuelc' and
%   uses the values in weights as weighting parameters. weights
%   must be a 1x5 row vector.
%
%   WELINE('inp_fuelc', weights, num_int_pnt) evaluates 'inp_fuelc'
%   and uses the weighting parameters from weights and num_int_pnt
%   points for interpolation of the powermap. If omitted a standard
%   value of 100 is used.
%
%   WELINE('inp_fuelc', weights, num_int_pnt, num_powr_val) evaluates
%   'inp_fuelc' using weighting parameters as given in weights and
%   num_int_points for interpolation and finds num_powr_val values
%   of the eline.
%   Note: num_int_pnt > num_powr_val
%   If num_powr_val is omitted a standard value of 20 is used
%
%   WELINE_DAT = WELINE('inp_fuelc') contains several fields;
%
%   - .val contains the values on the powercurve
%   - .w contains values for w on the powercurve
%   - .T contains values for T on the powercurve
%   - .eff contains efficiency data
%   - .co contains CO data
%   - .hc contains HC data
%   - .nox contains NOx data
%   - .pm contains PM data
%
%   For example plot(WELINE_DAT.w, WELINE_DAT.T) plots the
%   weighted e-line
%
%   Created on: 23/01/05
%   By: Bastiaan Zuurendonk, b.p.zuurendonk@student.tue.nl
%
%   Revision History
%   ----------------
%   23/01/05 Created based on eline.m
```

```
%   24/01/05 Solved problem with division by zero -> NaN
%   18/03/05 Solved problem with use of '~=' and matrix dimensions
%   18/03/05 Emissions are included in the output now as well
%   31/03/05 Solved problem with NaN in find structure
%   31/03/05 Fuel use is included in the output now as well

% Check for good input (string must begin with 'FC_' and
% won't be > 20 characters)
if upper(inp_fuelc(1:3))=='FC_' & length(inp_fuelc) < 21

    % Load the file
    eval(inp_fuelc);

    % Determine several meshgrids for interpolation
    trq_min = min(fc_map_trq);
    trq_max = max(fc_map_trq);
    spd_min = min(fc_map_spd);
    spd_max = max(fc_map_spd);
    if nargin < 2, weights = [0.7 0.1 0.1 0.3 0]; end % Standard weights
    if nargin > 3 & num_int_pnt < num_powr_val, ...
            error('num_int_pnt < num_powr_val'); end
    if nargin < 3, num_int_pnt = 100; end % Standard value num_int_pnt
    trq_rough = fc_map_trq;
    spd_rough = fc_map_spd;
    [T_rough,w_rough] = meshgrid(trq_rough,spd_rough);
    trq_fine = [trq_min:(trq_max-trq_min)/(num_int_pnt-1):trq_max];
    spd_fine = [spd_min:(spd_max-spd_min)/(num_int_pnt-1):spd_max];
    [T_fine,w_fine] = meshgrid(trq_fine,spd_fine);

    % Determine the power map (kW) for the rough grid
    fc_map_kW = T_rough.*w_rough/1000;
    fc_map_min = min(min(fc_map_kW));
    fc_map_max = max(max(fc_map_kW));

    % Determine the efficiency map on the rough grid
    fc_eff_map = fc_map_kW*1000./(fc_fuel_map*fc_fuel_lhv);

    % Interpolate the data on the fine grid
    fc_eff_map_fine = griddata(T_rough,w_rough,fc_eff_map,T_fine,w_fine);
    fc_co_map_fine = griddata(T_rough,w_rough,fc_co_map,T_fine,w_fine);
    fc_hc_map_fine = griddata(T_rough,w_rough,fc_hc_map,T_fine,w_fine);
    fc_nox_map_fine = griddata(T_rough,w_rough,fc_nox_map,T_fine,w_fine);
    fc_pm_map_fine = griddata(T_rough,w_rough,fc_pm_map,T_fine,w_fine);

    % Determine/interpolate the maximum torque values and create
    % a matrix with possible and impossible values
    fc_max_trq_fine = interp1(spd_rough,fc_max_trq,spd_fine);
    fc_max_trq_map_fine = fc_max_trq_fine'*ones(1,num_int_pnt);
    [mtmx,mtmy] = find(T_fine>fc_max_trq_map_fine);
    mtm = ones(num_int_pnt);
    for i = 1:length(mtmx)

        mtm(mtmx(i),mtmy(i)) = NaN;
```

```
end

% Find the optimal point for several powervalues
if nargin < 4, num_powr_val = 20; end % Standard value num_powr_val

% Initialize a data structure for output
weline_dat.val = [];
weline_dat.w = [];
weline_dat.T = [];
weline_dat.eff = [];
weline_dat.co = [];
weline_dat.hc = [];
weline_dat.nox = [];
weline_dat.pm = [];

% Loop from the smallest to the largest possible powervalue
for powr_val = [fc_map_min:(fc_map_max-fc_map_min)/...
            (num_powr_val-1):fc_map_max]

    % Powercurve returns a matrix with ones and zeros corresponding
    % to points on the powercurve
    ipm = powercurve(spd_fine,trq_fine,powr_val*1000);

    % Compute the efficiency first (1-eff must be minimized)
    eff_ipm_raw = mtm.*ipm.*fc_eff_map_fine;
    eff_ipm = eff_ipm_raw-min(min(eff_ipm_raw));
    eff_max = max(max(eff_ipm));
    if eff_max ~= 0

        eff_ipm = eff_ipm/max(max(eff_ipm));
        eff_ipm = 1-eff_ipm;

    end

    % Compute the emissions next
    for emis_type = {'co' 'hc' 'nox' 'pm'}

        eval([emis_type{1} '_ipm_raw = mtm.*ipm.*fc_' emis_type{1} ...
                '_map_fine;']);
        eval([emis_type{1} '_ipm = ' emis_type{1} ...
                '_ipm_raw-min(min(' emis_type{1} '_ipm_raw));']);
        eval([emis_type{1} '_max = max(max(' emis_type{1} '_ipm));']);
        % Check for division by zero
        eval(['if ' emis_type{1} '_max ~= 0, ' emis_type{1} '_ipm = ' ...
                emis_type{1} '_ipm/max(max(' emis_type{1} '_ipm)); end']);

    end
    all_ipm = weights(1)*eff_ipm+weights(2)*co_ipm+weights(3)*...
        hc_ipm+weights(4)*nox_ipm+weights(5)*pm_ipm;
    all_min = min(min(all_ipm));
    if ~isnan(all_min)

        [weline_w_index,weline_T_index] = find(all_ipm==all_min);
        weline_w_index = weline_w_index(1);
```

```
            weline_T_index = weline_T_index(1);
            weline_dat.val = [weline_dat.val all_min];
            weline_dat.eff = [weline_dat.eff eff_ipm_raw(weline_w_index,...
                    weline_T_index)];
            weline_dat.co = [weline_dat.co co_ipm_raw(weline_w_index,...
                    weline_T_index)];
            weline_dat.hc = [weline_dat.hc hc_ipm_raw(weline_w_index,...
                    weline_T_index)];
            weline_dat.nox = [weline_dat.nox ...
                    nox_ipm_raw(weline_w_index,weline_T_index)];
            weline_dat.pm = [weline_dat.pm pm_ipm_raw(weline_w_index,...
                    weline_T_index)];
            weline_dat.w = [weline_dat.w spd_fine(weline_w_index)];
            weline_dat.T = [weline_dat.T trq_fine(weline_T_index)];

        end

    end
    weline_dat.pow = weline_dat.T.*weline_dat.w;
    weline_dat.fuel = weline_dat.pow./(weline_dat.eff*fc_fuel_lhv);
    fc_eff_map = fc_map_kW*1000./(fc_fuel_map*fc_fuel_lhv);

% Produce some error messages in case of incorrect inputs
elseif upper(inp_fuelc(1:3))~='FC_'

    error([inp_fuelc ' does not start with ''FC_''']);

elseif length(inp_fuelc) >= 21

    error([inp_fuelc ' is >= 21 characters and probably not a ...
        fuelconverter']);

end
```

D MATLAB: powercurve.m

```
function ipm = powercurve(w,T,powr)
% POWERCURVE Determine a powercurvematrix
%   ipm = POWERCURVE(w,T,powr) returns a matrix corresponding
%   to the isopowercurve for powr on a grid defined by
%   vectors w and T
%
%   The isopowercurve is given by: w*T=powr
%
%   If the isopowercurve is not found whithin the given bounds
%   an error is returned.
%
%   w is a row with speed values
%
%   T is a row with torque values
%
%   ipm is a matrix with size according to the length of w and T
%   in which all elements are NaN except the ones for which
%   the powercurve passes

% Initialize ipm
ipm = NaN*ones(length(w),length(T));

% Compute several used values
w_min = min(w);
w_max = max(w);
w_step = w(2)-w(1);
T_min = min(T);
T_max = max(T);
T_step = T(2)-T(1);

% Loop through all values for w and find corresponding T
w_index = 1;
for w_loop = w

    if w_loop ~= 0

        % Compute the value for T
        T_comp = powr/w_loop;

        % If T_comp is in the right range, mark it in ipm
        if T_comp < T_max+.5*T_step & T_comp > T_min-.5*T_step

            % Find the corresponding element in ipm
            [min_val,T_index] = min(abs(T-T_comp));
            ipm(w_index,T_index) = 1;

        end

    end
    w_index = w_index+1;

end
```

```
% Loop through all values for T and find corresponding w
T_index = 1;
for T_loop = T

    if T_loop ~= 0

        % Compute the value for w
        w_comp = powr/T_loop;

        % If w_comp is in the right range, mark it in ipm
        if w_comp < w_max+.5*w_step & w_comp > w_min-.5*w_step

            % Find the corresponding element in ipm
            [min_val,w_index] = min(abs(w-w_comp));
            ipm(w_index,T_index) = 1;

        end

    end
    T_index = T_index+1;

end
```

E MATLAB: cutnan.m

```
function y = cutNaN(x)

if min(size(x)) ~= 1

    error('x is not a row or column vector');

else

    nans = find(isnan(x));
    x(nans) = -1000;
    y = nonzeros(x+1000) - 1000;
    if size(x,1) == 1
        y = y';
    end

end
```