

Control and Constraint of NO_x Emissions during Transient Manœuvres, Based on an Engine Combustion Model

Peter Fussey
Oxford University and Ricardo UK

David Limebeer
Oxford University

ABSTRACT

Setting up engines to meet emissions limits often involves extensive steady-state calibration activities combined with ad-hoc strategies to compensate for transient operation. As engines become more complex and acceptable emissions levels ever lower, this task is becoming increasingly time consuming and expensive.

The inclusion of models in the engine control units offers a way to reduce some of this calibration effort. Model-based control is an active area of research with advanced approaches now being proposed. One example is the use of real-time models to regulate the burn angle during transient manœuvres. This paper describes the application of a control-orientated combustion model to control directly emissions during transients. The model is used to optimize and constrain the NO_x emissions directly, rather than controlling an inferred variable such as the burn angle. This has the benefit that calibration engineers will be able to set the emissions trade-off directly.

CITATION: Fussey, P. and Limebeer, D., "Control and Constraint of NO_x Emissions during Transient Manœuvres, Based on an Engine Combustion Model," *SAE Int. J. Engines* 7(2):2014, doi:10.4271/2014-01-1355.

INTRODUCTION

The emissions legislation for internal combustion engines is being progressively updated to cover 'real world' driving conditions; particularly transient manœuvres. For example, EURO 7 emissions limits will be measured using a proposed new drive cycle, the World harmonised Light duty Test Procedure (WLTP), which is significantly more dynamic than the previous New European Drive Cycle (NEDC). In addition, off-cycle Real Driving Emissions (RDE) are expected to be regulated with requirements to demonstrate compliance through Portable Emissions Measurements Systems (PEMS).

Most of the new opportunities for reducing emissions during a drive cycle by improving engine control relate to transient manœuvres, since steady-state operating points are already extensively optimized and calibrated.

This paper considers the optimization of engine control parameters for a Diesel engine to reduce emissions during transient manœuvres. Earlier work has sought to reduce emissions by controlling inferred variables such as the angle at which a certain percentage of the fuel had been burnt (burn angle), for example [1] and [2]. Whilst these approaches

successfully reduced emissions, they rely on the assumption that emissions can be reduced by controlling the burn angle. In the course of calibrating engines, it is helpful to understand the extent to which the emissions are reduced, and secondly to constrain the emissions during transient manœuvres. This work explores the use of an emissions-prediction model to control and constrain emissions during transient manœuvres.

A key feature of this work is the recognition that emissions models are complex and computationally expensive. Nonetheless, transient control has to be achieved with limited computational resources and a fast update rate. This research considers the exploitation of parallel computing opportunities for both the modelling of emissions and the optimization algorithm.

This paper starts with a review of the background to this work, where two development questions are considered:

- 1). Can an emissions prediction model be used to control and constrain emissions during transient manœuvres?
- 2). How can parallel computing be used to improve the optimization?

The answers to these questions are developed in simulation and then combined to suggest new direct emissions control algorithms that may be applied to combustion engines.

BACKGROUND

Diesel Combustion Control

A typical Diesel engine is shown in Figure 1. The engine includes many systems to control the combustion events in the cylinders; Exhaust Gas Recirculation (EGR), turbocharging, and the cooling of intake gases. The systems can be grouped into the fuel and air paths. The control of the fuel path can influence the next combustion event, since the fuel is injected directly into the cylinder, giving a response time of ~0.01 seconds [1]. The control of the gaseous mixture entering the cylinder is slower due to the transport delays from the EGR valve to the cylinder (~1 second). Boost pressure control is slower still, and is constrained by the inertia of the turbo-charging system (~a few seconds).

In general, the control of the combustion process is achieved by varying the fuel injection pressure, the fuel quantity and the injection timing together with the state of the gases coming into the engine (boost pressure and EGR ratio). Without direct cylinder pressure measurements, the combustion is controlled in an open-loop manner, with set-points for the fuelling and inlet gases calibrated at steady-state operating points on engine dynamometers. During transient operation, the set-points move in accordance with the driver inputs - typically torque and engine speed - with some transient modifications for features such as smoke limitations. However, the operating conditions of the engine during transients are often different to those used for calibrating the set-points. It is this mismatch that this work seeks to address.

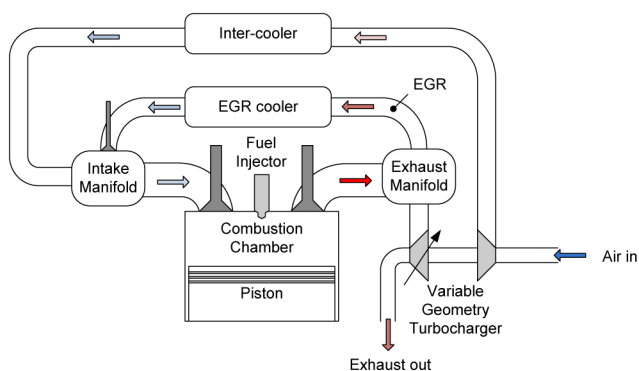


Figure 1. A typical diesel engine architecture focussing on the combustion system showing the flow of gases. An indication of the gas temperatures is given using coloured arrows. The system includes an EGR circuit, an EGR cooler, direct fuel injection, a variable geometry turbo-charger (VGT) and an intercooler.

Model Based Combustion Control

Models are widely used in engine control, however these are mainly centred on air system models and after-treatment models. In the field of combustion control, there has been research into the use of models to control the start of injection (SOI); see for example [1], [2] and [3].

In [1] models are used to estimate the intake gas O_2 concentration and subsequently the ignition timing. The former is controlled by the EGR system and the latter is a function of the estimated O_2 concentration and injection timing. The ignition model includes the spray behaviour, heat exchanges and oxidation reactions, with the start of ignition being defined as the time at which the gas temperature passes a threshold. The ignition timing model is used to estimate the ignition timing for three injection timings. The target injection timing is calculated by fitting a quadratic polynomial to these three points and evaluating the required injection timing to give the target ignition timing. The authors demonstrate this model running in a prototype controller and assess the potential production implementations by capturing the control actions in a Neural Network, avoiding the real-time calculation of the model and optimization.

In [2], a combustion model is developed as a series of differential equations with transitions from one combustion phase to another. The transitions are triggered as the integrated variables from the differential equations cross respective thresholds. The model is used to calculate a Crank Angle corresponding to $x\%$ of fuel burnt, CA_x .

The authors pose the problem in terms of fast (fuel path) and slow (air path) systems. CA_x is therefore a function of the fast and slow states:

$$CA_x = M(x_{slow}, x_{fast})$$

During a transient the slow states deviate from the set-point state and CA_x will differ from the set-point value. Noting set-point values with a bar, the error in CA_x is given as follows:

$$CA_x - \overline{CA_x} = M(x_{slow}, x_{fast}) - M(\bar{x}_{slow}, \bar{x}_{fast})$$

Defining the tracking errors for the states as $\delta x = x - \bar{x}$, one obtains:

$$CA_x - \overline{CA_x} = M(\bar{x}_{slow} + \delta x_{slow}, \bar{x}_{fast} + \delta x_{fast}) - M(\bar{x}_{slow}, \bar{x}_{fast})$$

The control approach involves modifying δx_{fast} such that the error in CA_x is zero. This is achieved using a Taylor expansion and sensitivity analysis. The real time implementation of this approach is achieved by noting that the modification of δx_{fast} can be scheduled on engine speed and engine load and stored as a two dimensional lookup table.

In [3], a combustion model is developed as a series of differential equations with multiple zones within the spray. Rather than controlling to a prescribed burn angle, the control is based on minimizing an objective function of the form:

$$J = R(r^{SOI} - y^{SOI})^2 + (r^{NOx} - y^{NOx})^2 + Q(r^{GIMEP} - y^{GIMEP})^2$$

in which r is the target steady-state set-point, y is the current operating point. The superscripts refer to SOI, NOx emissions and Gross Indicated Mean Effective Pressure (GIMEP) respectively, while Q and R are weights applied to the indicated error terms; the weight on NOx is unity without loss of generality.

This paper seeks to draw from the approaches summarized above by comparing the minimization of engine emissions to the approaches controlling to a prescribed burn angle. The emissions control is extended from [3] to cover multi-variable control of the NOx emissions.

Optimal Control

An optimal control problem involves choosing a control $u(t)$ that minimizes an objective function:

$$J = \varphi[x(t_F), t_F]$$

subject to the state equations,

$$\dot{x} = f[x(t), u(t)]$$

and initial conditions,

$$x(t_i) = x_i$$

where and t_i are t_F the initial and final times.

Optimal control methods usually consider the case where φ is differentiable and $\nabla\varphi$ can be computed, or estimated by finite differences. For these situations, there are a wide range of derivative-based algorithms to solve the problem.

In the application of Optimal Control to emissions control, the emissions model is large and computationally expensive. It can be assumed that the function can only be evaluated a small number of times, typically less than five times, between control actions. However, there are opportunities to exploit parallel computing which may allow multiple evaluations at each time step; 10-50 evaluations may be possible. The limitation on function evaluations has motivated a study of other minimization algorithms such as those classed as derivative-free optimization (DFO).

Derivative-Free Optimization (DFO)

DFO methods are a class of optimization algorithms, which seek to find the minimum of a function without evaluating the gradient; see the review [4]. One area where these methods have been adopted is in the field of 'simulation based optimization' [5]. These are situations where the design and

development process includes a detailed physical model, e.g. finite element model. The model is run repeatedly with different inputs to establish the optimum design. For these simulations, the cost of estimating the derivatives is too high and DFO approaches have been developed.

Three classes of DFO methods are described in [6]; *Stochastic methods* involve the iterative selection of randomly distributed points, *Lipschitz optimization methods* which involve deriving bounding functions based on the Lipschitz constant or an estimate of it and *Pattern search methods* which evaluate the function at a pattern of points (simplex) and then update the pattern for the next iteration. The best known of the direct search algorithms is the Nelder-Mead simplex algorithm which executes a series of expansions and contractions of the search space to locate the minimum.

DFO methods can have a slow rate of convergence which should be balanced with their ease of implementation and the avoidance of gradient calculations. Parallel applications of these algorithms, see [7], present an opportunity for achieving a low number of iterations which are required for the emissions control problem, however, they typically have a variable number of function evaluations for each iteration, or a number of iterations which is still too large.

Parallel Computing

The application of an algorithm (usually sequential) to a parallel computing architecture needs to consider both the target hardware configuration and the method for reconfiguring the sequential algorithm to the parallel architecture.

There are many hardware configurations available for parallel computing and are often defined by how the computations are divided up, how the parallel processors communicate and how the memory is shared. This work considers the Field Programmable Gate Array (FPGA), which is available for implementation in automotive control units. The FPGA can be configured to calculate functions as a pipeline with one calculation being started before the previous one has completed.

Methods for exploiting parallel computing architectures have been studied extensively, and seek to identify independent tasks, requiring approximately equal computational resources, to be evaluated concurrently.

When considering configuring a sequential algorithm for implementation on parallel computing architectures, there are often three areas of parallelism to consider, see [7]. The first is *Parallelism across the method* in which computations for each step can be solved in parallel, *parallelism across the system* in which the problem to be solved is broken into separate sub-problems which can be solved in parallel and *parallelism across the steps* in which techniques are used to solve the problem simultaneously across a large number of steps.

The model based engine control problem being studied here has two parts, both of which are candidates for parallelization; the model of combustion (which typically consists of a system of differential equations) and the optimization algorithm itself.

Parallel Solution of Ordinary Differential Equations (ODEs)

The application of parallel computing to the solution of ODEs has been the subject of many years of research; see [8] for a review. Opportunities for parallelising the solution are limited by the inherent sequential nature of the problem, however there are some techniques which can be applied in the three areas of parallelism described above.

Parallel Computing Applied to Optimization

Parallel computing architectures offer significant opportunities to improve the solution of optimization problems, where they can be configured appropriately. The iterative nature of many of the approaches means that they must be reviewed in the light of parallel computing opportunities. In particular, the solution of an optimization problem with a low number of iterations and a fixed number of function evaluations at each iteration has so far received little or no attention.

Summary of Background Material

The review of Model-Based Engine Control has outlined existing approaches using Start of Injection control, which will provide a baseline with which to compare to an approach based on minimizing emissions.

The review of the application of parallel computing to the control of emissions has provided a number of starting points for optimization algorithms. This paper explores optimization problems with low numbers of iterations and a fixed number of function evaluations per iteration. We will only consider the application of parallel computing to the optimization, since the application of parallel computing to the solution of the combustion model (ODEs) is more mature and offers fewer opportunities for improvement.

NOx CONTROL DURING TRANSIENT MANŒUVRES

The purpose of this section is to develop a model-based combustion control algorithm using an emissions model. The focus is on the control of NOx emissions during transient manœuvres.

Approach

A detailed simulation model has been used to develop an emissions control algorithm. This model includes a Mean Value Engine Model (MVEM) to capture the dynamics of the air path, an emissions model and the optimal controller for the Exhaust Gas Recirculation (EGR) and Start Of Injection (SOI).

Initially, the MVEM was used to quantify the response of the EGR actuator and to see if EGR could also be considered as a 'fast' actuator. The step response in [Figure 3](#) shows that the response for this engine is of the order of 0.25 seconds, and therefore with respect to an engine transient manœuvre, the EGR actuator can be considered 'fast'. This is to be expected, since the EGR enters close to the intake manifold and will affect the subsequent intake strokes.

The control investigation starts by using the SOI as the control input, as described in [1,2]. This is followed by a similar analysis using the EGR as the control input. Finally a two dimensional optimization is performed with both SOI and EGR being adjusted.

The objective function, J , selected for this study is derived from [3] and seeks to minimize the difference between the NOx emissions calculated with the current cylinder conditions and the control action, $y(\mathbf{x}, \mathbf{u})$, and the NOx emissions at the steady-state set-point, $y(\bar{\mathbf{x}}, \bar{\mathbf{u}})$, and the difference in control parameters, $\mathbf{u}$, from those at the steady-state set-point, $\bar{\mathbf{u}}$.

$$J(\mathbf{u}) = (y(\mathbf{x}, \mathbf{u}) - y(\bar{\mathbf{x}}, \bar{\mathbf{u}}))^2 + Q(u_1 - \bar{u}_1)^2 + R(u_2 - \bar{u}_2)^2$$

where Q and R are relative weightings between the NOx emissions error and the differences in the control inputs, $\mathbf{x}$ are the in-cylinder states. Note this objective function will tend to drive the system to the set-point as the engine states stabilize to steady-state conditions, even if the model predicts a better NOx emission with different control settings. This takes advantage of the extensive steady-state calibration, whilst allowing a deviation from these points during a transient when the engine states do not match those at the steady-state operating point.

As described earlier, the emissions control application requires an optimization algorithm that caters for the case where computing time is limited, whilst parallel computing is available, and a number of function evaluations can be executed at each iteration. One has to contemplate iterative procedures that converge in a small number of steps (approximately five). The majority of optimization algorithms require several sequential evaluations of the objective function to estimate the minimum, and so a new approach has been developed that is called the Parallel Trust Region (PTR) algorithm, which is detailed in the [Appendix](#).