Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2014, Article ID 286538, 21 pages
http://dx.doi.org/10.1155/2014/286538

Research Article

Optimal Control of Diesel Engines: Numerical Methods, Applications, and Experimental Validation

Jonas Asprion,[1] Oscar Chinellato,[2] and Lino Guzzella[1]

[1] *Institute for Dynamic Systems and Control, ETH Zurich, Sonneggstraße 3, 8092 Zurich, Switzerland*
[2] *FPT Motorenforschung AG, Schlossgasse 2, 9320 Arbon, Switzerland*

Correspondence should be addressed to Jonas Asprion; jonas.asprion@alumni.ethz.ch

Received 6 October 2013; Revised 20 November 2013; Accepted 20 November 2013; Published 5 February 2014

Academic Editor: Hui Zhang

In response to the increasingly stringent emission regulations and a demand for ever lower fuel consumption, diesel engines have become complex systems. The exploitation of any leftover potential during transient operation is crucial. However, even an experienced calibration engineer cannot conceive all the dynamic cross couplings between the many actuators. Therefore, a highly iterative procedure is required to obtain a single engine calibration, which in turn causes a high demand for test-bench time. Physics-based mathematical models and a dynamic optimisation are the tools to alleviate this dilemma. This paper presents the methods required to implement such an approach. The optimisation-oriented modelling of diesel engines is summarised, and the numerical methods required to solve the corresponding large-scale optimal control problems are presented. The resulting optimal control input trajectories over long driving profiles are shown to provide enough information to allow conclusions to be drawn for causal control strategies. Ways of utilising this data are illustrated, which indicate that a fully automated dynamic calibration of the engine control unit is conceivable. An experimental validation demonstrates the meaningfulness of these results. The measurement results show that the optimisation predicts the reduction of the fuel consumption and the cumulative pollutant emissions with a relative error of around 10% on highly transient driving cycles.

1. Introduction

Optimal control of diesel engines becomes increasingly important. More stringent emission regulations [1, 2] require the exploitation of the remaining potential of reducing the emissions, not only in stationary operation but also especially during transients [3, 4]. Simultaneously, the fuel consumption has to be minimised for economic and environmental reasons.

The classical approach to parameterise an engine control unit (ECU) is to first derive stationary lookup maps. Subsequently, transient corrections are added to these static maps, and heuristic feedforward parts are included. This approach, which relies mainly on engineering experience, leads to a highly iterative procedure when transient driving cycles need to be considered. With the increasing complexity of modern engine systems, it becomes difficult for the calibration engineer to conceive all the cross couplings between the many actuators. A high demand for test-bench time results. Furthermore, each new calibration of an engine requires this manual procedure to be executed again. Particularly for heavy-duty engines, which are usually employed in a variety of applications, a fast and partially automated calibration process is desirable.

Model-based approaches and mathematical optimisation tools provide the means for such an automation. For example, during the calibration of the ECU, the static lookup maps need to be optimised. This optimisation can be performed directly on the engine [5], or mathematical models may be used in place of the physical engine [6–8]. An overview of modelling principles suitable for this approach can be found in [9]. The information about the driving cycle to be considered may be included by a weighting of the operating range of the engine. This weighting matrix represents the time during which the engine is operated at a certain engine speed and load torque. Alternatively, a few relevant operating points can be identified, and the optimisation is restricted to those points [10, Chapter 7]. Identifying the parameters of a feedback controller can also be cast as an optimisation problem. Again, the solution can be obtained directly on the engine [11] or by a model-based approach [12]. Adaptive methods aim to automatically improve the engine calibration during normal operation [13].

Dynamics may be incorporated either online by model-predictive control [14–16] or offline by an optimisation of the control-trajectories over a representative driving profile. The solution of such optimal control problems (OCPs), which in the past were also obtained directly on the engine [17], provides implications for well-suited control structures [18–21]. The optimal control profiles with the corresponding fuel consumption and pollutant emissions can be used as a benchmark to disclose scenarios in which the performance of the actual control system is suboptimal. An analysis of these cases provides guidelines for the improvement of the control structure, the feedforward control, and the reference-trajectory generation. Another way of utilising the results from optimal control is to use this data to train function approximators such as artificial neural networks [22, 23].

In the literature referenced, it is stated that due to the complexity of the system, OCPs for diesel engines can only be solved over short time horizons of around 3 s to 10 s. Alternatively, model simplifications such as a quasistationary treatment are performed. However, to derive implications for a suitable control structure or to train function approximators, optimal solutions over long time horizons are required, based on a detailed, fully dynamic engine model. Transient driving cycles, which are used during the homologation procedure and thus emulate a representative operation scenario of the engine, are typically 20 to 30 minutes long.

This paper presents the means to calculate such optimal solutions. The optimisation-oriented modelling of diesel engines is summarised and extended to engines with exhaust-gas recirculation (EGR) in Section 2.2. Subsequently, the problem formulation is detailed, and the numerical optimisation methods are described in Sections 2.3–2.6. The presentation of the results is subdivided into three parts. First, the performance of the numerical methods is analysed in Section 3.1 and, subsequently, two case studies illustrate how the results from the optimisation can be utilised in Section 3.2. Finally, the experimental validation provided in Section 3.3 demonstrates the meaningfulness and the soundness of the approach.

A reader interested in the engineering aspects of the problem may focus on Sections 2–2.2, 2.6, and 3.2. Conversely, a reader interested in the details of the optimisation method and its performance may concentrate on Sections 2.3–2.5 and 3.1.

Throughout the text, the derivative with respect to time is denoted by $\dot{x} = dx/dt$, whereas $\overset{*}{x}$ designates a flow. Lower and upper bounds are denoted by $\underline{x}$ and $\overline{x}$, and $\widehat{x}$ indicates a reference value. Bold-face symbols such as $\mathbf{x}$ or $\mathbf{X}$ represent vectors and matrices, respectively. The abbreviations and symbols are summarised in the nomenclature section at the end of the text, except for the specific notation used in Sections 2.4 and 2.5.

Table 1: Main data of the engines used.

Engine		A	B
Displ. volume V_d	(l)	8.7	3.0
Cylinders n_{cyl}	(—)	6	4
Bore/stroke	(mm)	117/135	96/104
Compression ratio	(—)	15.8	17.6
EGR		—	High pressure
Rated power	(kW)	300 (1,650 rpm)	130 (2,900 rpm)
Max. torque	(Nm)	1,720 (1,200 rpm)	420 (1,400 rpm)

2. Materials and Methods

This section introduces the engines used and presents all methods required for the optimal control of diesel engines. The numerical framework for optimal control as well as the engine model is implemented in MATLAB 2013b (MathWorks, Natick, MA, USA), running under 64bit Windows 8. All computations are performed on a laptop computer with an Intel Core i7-2760QM CPU running at a clock speed of 2.40 GHz.

2.1. Engines. The relevant data of the two engines considered here are provided in Table 1. Engine A is a heavy-duty engine which is used in on-road and off-road applications. It does not have an EGR system but relies on a selective catalytic-reduction system (SCR) to reduce the engine-out NO_x emissions to the level imposed by the legislation. Engine B is a light-duty engine that is used mainly in light commercial vehicles. It has an EGR system but no SCR. Both engines are equipped with a common-rail injection system and a diesel particulate filter (DPF). The soot emissions are low enough such that no active regeneration of the DPF is necessary. The optimisation thus has to maintain a similar level of soot emissions and should not produce large instantaneous soot peaks.

2.2. Engine Modelling. During an optimisation, a vast number of model evaluations are performed. Either the model function itself is evaluated, for example, during static calibration or when employing simultaneous methods for optimal control [24], or a forward simulation of the model is used, for example, for parametric studies or in the context of shooting methods for optimal control [25, Sections 3.2–3.4]. Therefore, the model has to be simple to enable a fast execution. In both cases described, partial derivatives need to be calculated as well. If only standard mathematical operations are used, automatic differentiation is applicable, which enables a fast and accurate evaluation of partial derivatives [26, 27].

Besides being computationally efficient, models apt for optimisation have to be smooth and quantitatively accurate, capture all relevant qualitative trends, and provide plausible extrapolation. These properties qualify the model itself or its simulation as an "accurate and consistent function generator" [25, Section 3.8]. As a fundamental requirement, the model outputs have to be predicted using the control signals, known parameters, and the ambient conditions only. Ideally,

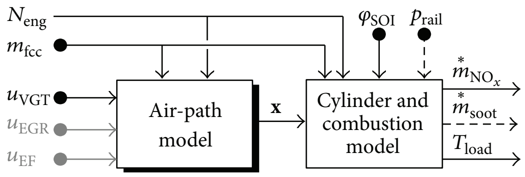


Figure 1: Structure of the engine model. The model for the engine with EGR does include the grey signals, but not the dashed ones.

the model can be identified using a small set of measurements in order not to cancel the reduction of test-bench time gained by the application of model-based approaches. If the model is able to predict the effects induced by influences that can hardly be excited on the test bench, its value for parametric studies is further enhanced.

Optimisation-oriented models are thus classified between control-oriented and phenomenological models [28, 29]. The former only capture the trends relevant to control while being as simple as to be implementable on the ECU. Due to the presence of feedback control, the quantitative accuracy is of minor importance, and the models have to be valid only in a region around the setpoints of the controller. In contrast, phenomenological models are used to perform predictive parametric studies or to analyse specific effects. Since a high execution speed is not critical, a single model evaluation typically takes from a few seconds up to several hours [30]. Due to their complexity, such models often suffer from error propagation [31, 32].

2.2.1. The Model Used. An optimisation-oriented model for a diesel engine without EGR is presented in [29, 33]. A classical mean-value model for the air path is extended by thermal models for the intake and the exhaust manifolds. Physics-based setpoint-relative models are used for the combustion efficiency, the in-cylinder processes, and the NO_x emissions. However, since all setpoint maps are replaced by polynomials, the model is smooth and apt for algorithmic differentiation. The models for the fuel consumption and the NO_x emissions capture the influence of the air-path state, the injection pressure, and the injection timing, that is, the start of injection (SOI). The modelling errors are in the range of 0.6% and 5%, respectively. For the soot emissions, the simple model described in [34] is used. It captures the effects of the injection pressure and the air-to-fuel ratio (AFR).

Figure 1 illustrates the general structure of the model. Only the air path is modelled as a dynamic system, whereas all phenomena occurring in the cylinders are assumed to be instantaneous processes. The state variables **x** of the air path are inputs to this static part of the model. The exhaust-gas aftertreatment system (ATS) is included in the model only as a flow restriction in the air path. The engine-out emissions are limited directly in the optimisation problem.

2.2.2. Extension to EGR. Several submodels need to be added or adapted for an engine with EGR. The dynamics in the intake manifold comprise balances for the mass, the energy, and the burnt-gas fraction. The corresponding differential equations read

$$\frac{dp_{\mathrm{IM}}}{dt} = \frac{R\kappa}{V_{\mathrm{IM}}} \left(\overset{*}{m}_{\mathrm{CP}}\vartheta_{\mathrm{IC}} + \overset{*}{m}_{\mathrm{EGR}}\vartheta_{\mathrm{EGR}} - \overset{*}{m}_{\mathrm{cyl}}\vartheta_{\mathrm{IM}}\right), \tag{1a}$$

$$\frac{d\vartheta_{\mathrm{IM}}}{dt} = \frac{R\vartheta_{\mathrm{IM}}}{p_{\mathrm{IM}}V_{\mathrm{IM}}c_v} \left[c_p \left(\overset{*}{m}_{\mathrm{CP}}\vartheta_{\mathrm{IC}} + \overset{*}{m}_{\mathrm{EGR}}\vartheta_{\mathrm{EGR}} - \overset{*}{m}_{\mathrm{cyl}}\vartheta_{\mathrm{IM}}\right) - c_v\vartheta_{\mathrm{IM}}\left(\overset{*}{m}_{\mathrm{CP}} + \overset{*}{m}_{\mathrm{EGR}} - \overset{*}{m}_{\mathrm{cyl}}\right)\right], \tag{1b}$$

$$\frac{dx_{\mathrm{BG,IM}}}{dt} = \frac{R\vartheta_{\mathrm{IM}}}{p_{\mathrm{IM}}V_{\mathrm{IM}}} \left(\overset{*}{m}_{\mathrm{EGR}}\left(x_{\mathrm{BG,EM}} - x_{\mathrm{BG,IM}}\right) - x_{\mathrm{BG,IM}}\overset{*}{m}_{\mathrm{CP}}\right). \tag{1c}$$

The burnt-gas fraction in the exhaust gas is

$$x_{\mathrm{BG,EM}} = \frac{x_{\mathrm{BG,IM}} \cdot \overset{*}{m}_{\mathrm{cyl}} + \left(1 + \sigma_0\right) \cdot \overset{*}{m}_{\mathrm{fuel}}}{\overset{*}{m}_{\mathrm{cyl}} + \overset{*}{m}_{\mathrm{fuel}}}, \tag{2}$$

where $\sigma_0 \approx 14.5$ is the stoichiometric AFR.

The mass-flow through the EGR valve is modelled by a simplified flow function for compressible fluids [1, Section 2.3.5]:

$$\overset{*}{m}_{\mathrm{EGR}} = A_{\mathrm{EGR}} \cdot \frac{p_{\mathrm{EM}}}{\sqrt{R \cdot \vartheta_{\mathrm{EM}}}} \cdot \Psi\left(\frac{p_{\mathrm{EM}}}{p_{\mathrm{IM}}}\right), \tag{3a}$$

$$\Psi\left(\Pi\right) = \sqrt{\frac{2}{k_\Pi \cdot \Pi} \cdot \left(1 - \frac{1}{k_\Pi \cdot \Pi}\right)}, \tag{3b}$$

$$A_{\mathrm{EGR}} = k_{\mathrm{EGR},1} \cdot u_{\mathrm{EGR}} + k_{\mathrm{EGR},2} \cdot u_{\mathrm{EGR}}^{k_{\mathrm{EGR},e}}. \tag{3c}$$

The factor $k_\Pi \approx 1.04$ accounts for flow phenomena that change the effective pressure ratio over the EGR valve. The model for the EGR cooler has the same structure as the intercooler model presented in [33].

An exhaust flap (EF) is installed directly after the turbine. Its purpose is to choke the flow of exhaust gas such that the pressure in the exhaust manifold increases. This higher back-pressure enables higher EGR mass-flows. The EF is modelled by a factor that reduces the effective opening area of the ATS:

$$x_{\mathrm{EF}} = 1 - k_{\mathrm{EF},1} \cdot u_{\mathrm{EF}}^{k_{\mathrm{EF},2}}. \tag{4}$$

Due to the lower oxygen concentration when EGR is applied, the combustion efficiency is slightly reduced. The factor

$$\eta_{\xi_{\mathrm{O}_2}} = 1 - a_{\xi_{\mathrm{O}_2}}\left(N_{\mathrm{eng}}, m_{\mathrm{fcc}}\right) \cdot \left\|\xi_{\mathrm{O}_2} - \widehat{\xi}_{\mathrm{O}_2}\right\|^{b_{\xi_{\mathrm{O}_2}}} \tag{5a}$$

is appended as an additional multiplicative efficiency reduction. The reference oxygen mass-fraction before the combustion, $\widehat{\xi}_{\mathrm{O}_2}$, as well as the exponential factor $b_{\xi_{\mathrm{O}_2}}$ are scalar parameters. The efficiency reduction is a function of the engine operating point:

$$a_{\xi_{\mathrm{O}_2}} = a_0 + a_1 \cdot N_{\mathrm{eng}} + a_2 \cdot m_{\mathrm{fcc}}. \tag{5b}$$

The ignition-delay model also has a multiplicative structure. The additional correction factor

$$\tau_{\xi_{O_2}} = 1 - k_{\text{lin}} \cdot \left(\xi_{O_2} - \hat{\xi}_{O_2}\right) - k_{\text{quad}}\left(N_{\text{eng}}, m_{\text{fcc}}\right) \cdot \left(\xi_{O_2} - \hat{\xi}_{O_2}\right)^2 \tag{6a}$$

is introduced, where the coefficient of the second-order term is again a function of the engine operating point:

$$k_{\text{quad}} = k_{\text{quad},0} + k_{\text{quad},1} \cdot N_{\text{eng}} + k_{\text{quad},2} \cdot m_{\text{fcc}} + k_{\text{quad},3} \cdot N_{\text{eng}}^2 + k_{\text{quad},4} \cdot m_{\text{fcc}}^2 + k_{\text{quad},5} \cdot N_{\text{eng}} \cdot m_{\text{fcc}}. \tag{6b}$$

The NO_x model inherently accounts for the change of the temperature and of the composition of the intake air by the EGR. However, since the combustion speed is changed by the reduced oxygen availability, a smaller region in the cylinder reaches a temperature that is sufficiently high for thermal NO_x formation. This effect is found to depend on the engine speed and therefore the factor

$$x_{O_2} = \xi_{O_2}^{k_{O_2} \cdot N_{\text{eng}}} \tag{7}$$

is applied to the formation volume in the original NO_x model [29]. The effect of the EGR, which reduces the NO_x emissions by a factor of up to 18, is predicted by the model with an average magnitude of the relative error of 8%.

The simple soot model that is suitable for engine A does not yield plausible results for engine B with EGR. Therefore, no model for the soot emissions is used. In the OCP, the corresponding limit is replaced by a lower bound on the AFR.

2.3. Numerical Optimal Control. A general formulation of an OCP reads

$$\min_{\mathbf{x}(\cdot),\mathbf{u}(\cdot)} \int_0^T L\left(\mathbf{x}(t), \mathbf{u}(t), \boldsymbol{\pi}(t)\right) dt \tag{8a}$$

$$\text{s.t.} \quad \dot{\mathbf{x}}(t) - \mathbf{f}\left(\mathbf{x}(t), \mathbf{u}(t), \boldsymbol{\pi}(t)\right) = 0, \quad t \in [0, T], \tag{8b}$$

$$\int_0^T \mathbf{g}\left(\mathbf{x}(t), \mathbf{u}(t), \boldsymbol{\pi}(t)\right) dt - \hat{\mathbf{g}} \le 0, \tag{8c}$$

$$\mathbf{c}\left(\mathbf{x}(t), \mathbf{u}(t), \boldsymbol{\pi}(t)\right) \le 0, \quad t \in [0, T], \tag{8d}$$

$$\underline{\mathbf{x}}(t) \le \mathbf{x}(t) \le \overline{\mathbf{x}}(t), \quad t \in [0, T], \tag{8e}$$

$$\underline{\mathbf{u}}(t) \le \mathbf{u}(t) \le \overline{\mathbf{u}}(t), \quad t \in [0, T]. \tag{8f}$$

The goal is to find trajectories for the state variables $\mathbf{x}(t)$ and the control inputs $\mathbf{u}(t)$ that minimise the integral cost (8a) while satisfying the dynamics of the system (8b), the integral inequality constraints (8c), and the time-variable path constraints (8d). (The integral cost L is called a Lagrange term. Every differentiable end cost, also called a Mayer term, can be replaced by an equivalent Lagrange term. The latter is to be preferred from a numerical point of view [25, Section 4.9].) The simple bounds (8e) and (8f) represent the actuator ranges, mechanical limits, and fixed initial or end conditions. The system has n_x state variables, that is, $\mathbf{x}, \mathbf{f}, \underline{\mathbf{x}}, \overline{\mathbf{x}} \in \mathbb{R}^{n_x}$, and there are n_u control inputs to the system, $\mathbf{u}, \underline{\mathbf{u}}, \overline{\mathbf{u}} \in \mathbb{R}^{n_u}$. Several time-variable parameters $\boldsymbol{\pi} \in \mathbb{R}^{n_\pi}$ may be present. Finally, there are n_g integral constraints and n_c path constraints, that is, $\mathbf{g}, \hat{\mathbf{g}} \in \mathbb{R}^{n_g}$, and $\mathbf{c} \in \mathbb{R}^{n_c}$, respectively.

The most common approaches to tackle continuous-time OCPs are outlined in Figure 2. The top level is inspired by [35]. Partial overviews are provided in [36–38], and [39] presents a brief historical outline. Solving the Hamilton-Jacobi-Bellman equation, which is a partial differential equation subject to the model equations, is practicable only for a system with few state variables and control inputs. Similar to its discrete-time equivalent, dynamic programming, it suffers from the curse of dimensionality for larger systems. Likewise, the indirect approach can hardly be applied to large and complex problems. If the first-order optimality conditions can be derived analytically, still a two-point boundary-value problem (BVP) needs to be solved. The dynamics of the costate variables however are ill-conditioned, and it is impossible to derive a good initial guess for the general case.

Direct methods start by discretising the full OCP. A finite-dimensional, constrained nonlinear optimisation problem results. This type of problem is termed nonlinear program (NLP). Due to the performance and the robustness of current NLP solvers, this approach is a means to an efficient numerical solution of large-scale, complex OCPs.

Within direct methods, two main approaches exist. The *sequential* approach, also called *single shooting*, discretises only the control inputs, and a forward simulation is used to evaluate the model. Although this method is simple to implement, the resulting NLP is highly nonlinear and sensitive [25, Sections 3.3 and 3.4], which limits the length of the time horizon. Furthermore, consistent derivatives have to be available for the solution of the NLP [25, Section 3.8], which requires the use of custom ODE solvers. Finally, instabilities of the model can lead to a failure of the ODE solver and thus a premature termination of the optimisation.

The alternative to the sequential approach is to discretise the state trajectories along with the control inputs. The entire continous-time problem is "transcribed" into a large but extremely sparse NLP, which fully includes the discretised state trajectories. No forward simulation is performed, and the model ODEs are only satisfied after the solution of the NLP. This *simultaneous* optimisation and simulation resolves the problems encountered by the sequential approach. One drawback is that the accuracy of the solution of the ODEs is coupled to the discretisation of the control inputs. An iterative mesh refinement may be necessary to obtain a sufficiently accurate solution. A recent overview of simultaneous approaches is provided in [24, 40], while [41] unveils the beginnings of these methods.

2.3.1. Direct Collocation. Collocation methods are a family of integration schemes often used for the direct transcription of OCPs [42–47]. In contrast to other Runge-Kutta schemes, they represent the state trajectories on each integration interval as polynomials and thus provide a continuous solution.

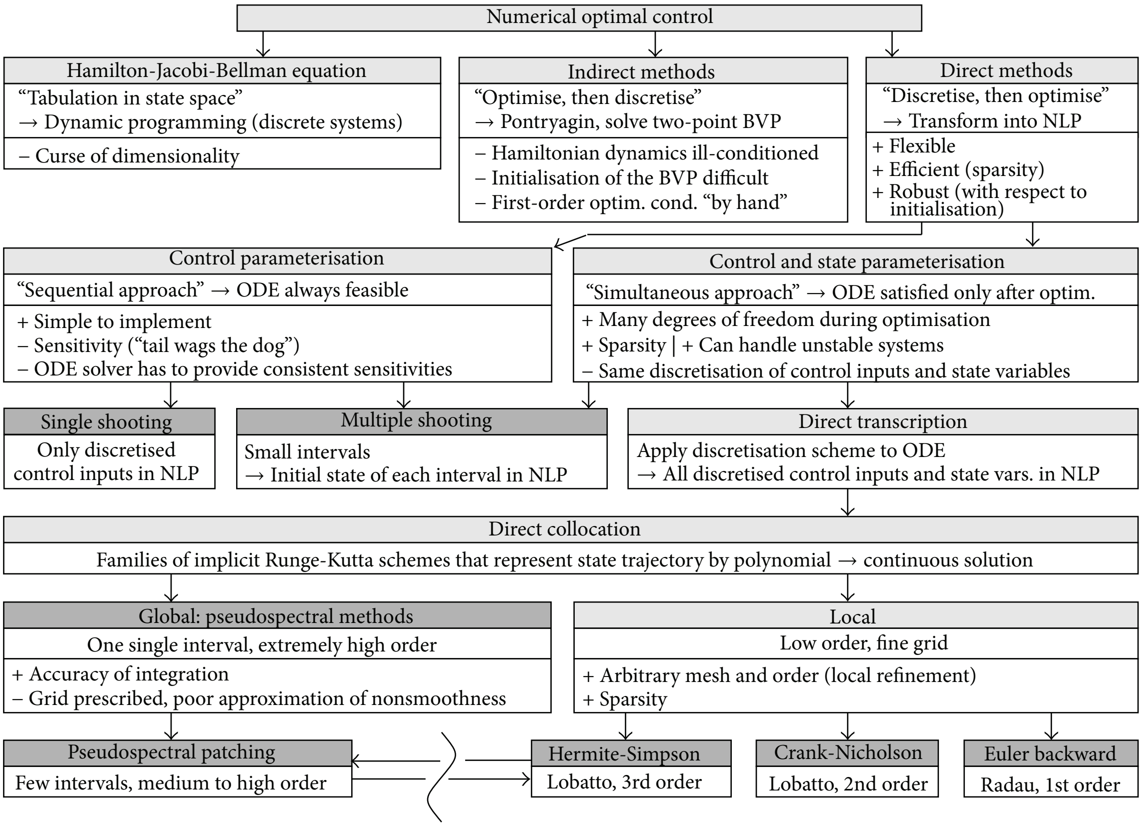

Numerical optimal control
Hamilton-Jacobi-Bellman equation
"Tabulation in state space"
→ Dynamic programming (discrete systems)
− Curse of dimensionality
Indirect methods
"Optimise, then discretise"
→ Pontryagin, solve two-point BVP
− Hamiltonian dynamics ill-conditioned
− Initialisation of the BVP difficult
− First-order optim. cond. "by hand"
Direct methods
"Discretise, then optimise"
→ Transform into NLP
+ Flexible
+ Efficient (sparsity)
+ Robust (with respect to initialisation)
Control parameterisation
"Sequential approach" → ODE always feasible
+ Simple to implement
− Sensitivity ("tail wags the dog")
− ODE solver has to provide consistent sensitivities
Control and state parameterisation
"Simultaneous approach" → ODE satisfied only after optim.
+ Many degrees of freedom during optimisation
+ Sparsity | + Can handle unstable systems
− Same discretisation of control inputs and state variables
Single shooting
Only discretised control inputs in NLP
Multiple shooting
Small intervals
→ Initial state of each interval in NLP
Direct transcription
Apply discretisation scheme to ODE
→ All discretised control inputs and state vars. in NLP
Direct collocation
Families of implicit Runge-Kutta schemes that represent state trajectory by polynomial → continuous solution
Global: pseudospectral methods
One single interval, extremely high order
+ Accuracy of integration
− Grid prescribed, poor approximation of nonsmoothness
Local
Low order, fine grid
+ Arbitrary mesh and order (local refinement)
+ Sparsity
Pseudospectral patching
Few intervals, medium to high order
Hermite-Simpson
Lobatto, 3rd order
Crank-Nicholson
Lobatto, 2nd order
Euler backward
Radau, 1st order

Figure 2: Overview of the most prominent methods for the numerical solution of optimal control problems.

The method chosen here is the family of Radau collocation schemes [48]. It provides stiff accuracy and stiff decay (or L-stability) [48], [49, Section 3.5]. L-stable schemes approximate the true solution of a stiff system when the discretisation is refined. Furthermore, the integral and the differential formulations of Radau collocation are equivalent [47]. This property is necessary to construct a consistent transcription of the OCP at hand.

Historically, two different branches of direct collocation methods evolved, which is indicated in Figure 2. On the one hand, low-order methods originated from the forward simulation of ODEs. A step-size adaptation is used to compensate for the low order. On the other hand, *pseudospectral methods* originally evolved in the context of partial differential equations within fluid dynamics. In their purest form, these methods do not subdivide the time horizon into intervals but represent the state variables as single high-order polynomials. The order of these polynomials may be increased to achieve a higher accuracy. In the context of direct collocation, these two branches can be cast in a unified framework that allows for an arbitrary order on each collocation interval. Nowadays, approaches that selectively refine the step size and the collocation order represent the state of the art [50, 51].

2.3.2. Nonlinear Programming. Before the equations for direct collocation and the structure of the resulting NLP are derived, the fundamentals of solving NLPs are summarised. This outline enables the interpretation of the performance of different solvers on the problem at hand. Neither completeness is claimed nor a proper mathematical foundation is followed. Several textbooks on the subject are available, for example, [52, 53].

An NLP is formulated as

$$\min_{\boldsymbol{\omega}} \quad F(\boldsymbol{\omega}) \tag{9a}$$

$$\text{s.t.} \quad \mathbf{g}(\boldsymbol{\omega}) = 0, \quad \mathbf{g} \in \mathbb{R}^{n_g}, \tag{9b}$$

$$\mathbf{h}(\boldsymbol{\omega}) \leq 0, \quad \mathbf{h} \in \mathbb{R}^{n_h}. \tag{9c}$$

By the introduction of the Lagrangian function

$$\mathscr{L}\left(\boldsymbol{\omega}, \boldsymbol{\lambda}_g, \boldsymbol{\lambda}_h\right) := F(\boldsymbol{\omega}) + \boldsymbol{\lambda}_g^T \mathbf{g}(\boldsymbol{\omega}) + \boldsymbol{\lambda}_h^T \mathbf{h}(\boldsymbol{\omega}), \tag{10}$$

the first-order necessary conditions for the NLP (9a)–(9c), also known as the Karush-Kuhn-Tucker (KKT) conditions, become

$$\nabla_{\boldsymbol{\omega}} \mathscr{L}\left(\boldsymbol{\omega}^*, \boldsymbol{\lambda}_g^*, \boldsymbol{\lambda}_h^*\right) = 0, \tag{11a}$$

$$\nabla_{\boldsymbol{\lambda}_g} \mathscr{L}\left(\boldsymbol{\omega}^*, \boldsymbol{\lambda}_g^*, \boldsymbol{\lambda}_h^*\right) = \mathbf{g}\left(\boldsymbol{\omega}^*\right) = 0, \tag{11b}$$

$$\nabla_{\boldsymbol{\lambda}_h} \mathscr{L}\left(\boldsymbol{\omega}^*, \boldsymbol{\lambda}_g^*, \boldsymbol{\lambda}_h^*\right) = \mathbf{h}\left(\boldsymbol{\omega}^*\right) \leq 0, \tag{11c}$$

$$\boldsymbol{\lambda}_h^* \geq 0, \tag{11d}$$

$$\lambda_{h,i}^* \cdot h_i(\boldsymbol{\omega}^*) = 0, \quad i = 1, \ldots, n_h. \tag{11e}$$

The matrix of the first partial derivatives of all constraints is called the Jacobian. The second-order sufficient conditions require the Hessian of the Lagrangian, $\nabla^2_{\boldsymbol{\omega}}\mathscr{L}(\boldsymbol{\omega}, \boldsymbol{\lambda}_g, \boldsymbol{\lambda}_h)$, projected on the null-space of the Jacobian, to be positive definite. This projection is termed the "reduced Hessian."

Equality-constrained problems can be solved by applying Newton's method to the nonlinear KKT conditions. After some rearrangements, one obtains the KKT system

$$\begin{pmatrix} \nabla F(\boldsymbol{\omega}_k) \\ \mathbf{g}(\boldsymbol{\omega}_k) \end{pmatrix} + \begin{bmatrix} \nabla^2_{\boldsymbol{\omega}}\mathscr{L}(\boldsymbol{\omega}_k, \boldsymbol{\lambda}_{g,k}) & \nabla\mathbf{g}(\boldsymbol{\omega}_k) \\ \nabla\mathbf{g}(\boldsymbol{\omega}_k)^T & 0 \end{bmatrix} \cdot \begin{pmatrix} \boldsymbol{\omega} - \boldsymbol{\omega}_k \\ \boldsymbol{\lambda}_g \end{pmatrix} = 0, \tag{12}$$

which is solved during each Newton iteration k. The solution of the quadratic program (QP)

$$\min_{\mathbf{p}} \quad \frac{1}{2}\mathbf{p}^T\nabla^2_{\boldsymbol{\omega}}\mathscr{L}(\boldsymbol{\omega}_k, \boldsymbol{\lambda}_{g,k})\,\mathbf{p} + \nabla F(\boldsymbol{\omega}_k)^T\mathbf{p} \tag{13a}$$

$$\text{s.t.} \quad \nabla\mathbf{g}(\boldsymbol{\omega}_k)^T\mathbf{p} + \mathbf{g}(\boldsymbol{\omega}_k) = 0, \tag{13b}$$

with $\mathbf{p} = \boldsymbol{\omega} - \boldsymbol{\omega}_k$, is equivalent to the solution of (12). Applying Newton's method to the KKT conditions thus can be interpreted as *sequential quadratic programming* (SQP).

To ensure progress from remote starting points, two different *globalisation strategies* are used. The *line search* approach first defines a direction and searches along this direction until an acceptable step length is found. The most popular search direction is the *Newton direction* defined by (12) or (13a) and (13b). In contrast, the *trust-region* approach defines a region, for example, a ball, around the current point in which a model for F—usually also a quadratic approximation—is assumed to be reliable. The trust-region radius is adjusted by assessing the model accuracy observed over the last step. In proximity of the solution, both approaches reduce to the standard Newton iteration on the KKT conditions, which exhibits quadratic convergence.

Quasi-Newton Approximations. In the KKT system (12) or the QP (13a) and (13b), the exact Hessian of the Lagrangian can be replaced by a quasi-Newton (QN) approximation. This substitution is possible since the current Lagrange multipliers $\boldsymbol{\lambda}_{g,k}$ only occur implicitly in the Hessian itself. The basic idea is to utilise the curvature information obtained along the NLP iterations to construct an approximation of the exact Hessian. The most prominent method is the limited-memory BFGS update [54], named after its discoverers C. G. Broyden, R. Fletcher, D. Goldfarb, and D. F. Shanno. This update preserves the positive definiteness of a usually diagonal initialisation.

Inequality Constraints. To solve inequality-constrained (IC) problems, two fundamentally different approaches are used nowadays. On the one hand, the idea of SQP can be extended to the IC case. These methods model (9a)–(9c) as an IC QP at each iteration. The search direction $\mathbf{p}_k$ for a line search is thus the solution of the QP

$$\min_{\mathbf{p}} \quad \frac{1}{2}\mathbf{p}^T\nabla^2_{\boldsymbol{\omega}}\mathscr{L}(\boldsymbol{\omega}_k, \boldsymbol{\lambda}_{g,k}, \boldsymbol{\lambda}_{h,k})\,\mathbf{p} + \nabla F(\boldsymbol{\omega}_k)^T\mathbf{p} \tag{14a}$$

$$\text{s.t.} \quad \nabla\mathbf{g}(\boldsymbol{\omega}_k)^T\mathbf{p} + \mathbf{g}(\boldsymbol{\omega}_k) = 0, \tag{14b}$$

$$\nabla\mathbf{h}(\boldsymbol{\omega}_k)^T\mathbf{p} + \mathbf{h}(\boldsymbol{\omega}_k) \leq 0. \tag{14c}$$

On the other hand, *interior-point* (IP) methods penalise the violation of the ICs by a logarithmic barrier function. The problem

$$\min_{\boldsymbol{\omega}} \quad F(\boldsymbol{\omega}) - \tau\sum_{i=1}^{n_h}\log(-h_i) \tag{15a}$$

$$\text{s.t.} \quad \mathbf{g}(\boldsymbol{\omega}) = 0, \tag{15b}$$

is solved, while the barrier parameter τ is decreased iteratively. The solution for one value is used to initialise the next iteration.

Both approaches to handle IC problems exhibit some advantages but also suffer from specific drawbacks. The main difference is that, within an SQP method, the structure of the QP approximation changes from iteration to iteration, whereas for IP methods, this structure is invariant. As a consequence, an IP method can afford to derive a good factorisation of the KKT system once to ensure a fast calculation of the Newton steps. Direct solvers for large, sparse linear systems are readily available. In contrast, an SQP method relies on a QP solver that detects the set of active ICs for the current QP approximation by itself. This solver thus has to deal with a constantly changing problem structure, which is usually handled by updates of the initial factorisation.

This difference is closely related to the two basic approaches to solve the KKT system. On the one hand, the full-matrix approach relies on a direct, symmetric indefinite factorisation of the KKT matrix. In this context, various software packages are available, for example, MA27/57/97 [55], MUMPS [56], or PARDISO [57]. On the other hand, the decomposition approach calculates and updates the null-space basis matrix. However, for large sparse problems, the reduced Hessian is much denser than the Hessian itself. Since dense linear algebra has to be applied to the reduced system, this approach is only computationally efficient when the number of the degrees of freedom is small.

The most important points concerning the advantages and the drawbacks of the SQP and IP methods are stated next [58].

(i) It is difficult to implement an exact-Newton SQP method. The main pitfalls are the nonconvexity of the QP subproblems when the exact Hessian is used and that SQP methods often rely on custom-tailored linear algebra. The latter limits the flexibility of those algorithms to adopt the latest developments in software and hardware technology.

(ii) IP methods are most efficient when relying on the exact second derivatives. Furthermore, they usually converge in fewer inner iterations, even for very large problems, and may utilize the latest "off-the-shelf" linear algebra software.

(iii) Applying IP solvers to the QPs in an SQP framework had limited success because they are hard to warm-start [25, Section 4.13]. In short, an IP solver, which follows a central path and approaches the constraint surface orthogonally, faces sensitivity problems when forced to step onto the solution path perpendicularly.

NLP Solvers Used. The following list characterises the NLP solvers used here.

SNOPT 7.2 [59], a proprietary solver implementing an SQP algorithm based on a decomposition approach (LU factorisation) and a line-search globalisation. It cannot use exact second derivatives but relies on a BFGS update.

IPOPT 3.11.0 [60], an open-source solver implementing a primal-dual IP method using a line-search globalisation. As linear solver, MUMPS [56] with METIS preordering [61] is used. IPOPT provides a BFGS update but can also exploit the exact second derivatives.

WORHP 1.2-2533 [62]. This solver tackles the QPs within an SQP framework by an IP solver that can be efficiently warm-started. A line-search globalisation and various partitioned BFGS updates are implemented. The default linear solver MA97 is used. WORHP is free for academic use.

KNITRO 8.0 [63], a proprietary solver that provides three algorithms, namely, an SQP trust-region method and two IP methods. The latter rely either on a direct solver and a line search or on a projected conjugate-gradient (CG) method to solve trust-region subproblems. Exact and QN methods are implemented.

2.3.3. Direct Collocation of Optimal Control Problems.

Radau collocation represents the state $x(t)$ of the scalar ODE $\dot{x} = f(x)$ on the interval $[t_0, t_f]$ as a polynomial, say of degree s. The time derivative of this polynomial is then equated to the values of f at s collocation points $t_0 < \tau_1 < \tau_2 \cdots < \tau_N = t_f$. The left boundary $\tau_0 = t_0$ is a noncollocated point. The notation $x_j := x(\tau_j)$ is adopted. The resulting system of equations reads

$$\begin{pmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \vdots \\ \dot{x}_s \end{pmatrix} \approx \underbrace{[\mathbf{d}_0, \widetilde{\mathbf{D}}]}_{=:\mathbf{D}} \cdot \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_s \end{pmatrix} \overset{!}{=} \begin{pmatrix} f(x_1) \\ f(x_2) \\ \vdots \\ f(x_s) \end{pmatrix}. \tag{16}$$

Each of these s equations (index i) is a sum of linear terms and one nonlinear term:

$$d_{0,i} \cdot x_0 + \sum_{j=1}^{s} \widetilde{D}_{ij} \cdot x_j - f(x_i) = 0. \tag{17}$$

The collocation nodes are defined as the roots of Legendre polynomials and have to be computed numerically [64, Section 2.3]. Lagrange interpolation by barycentric weights is used to calculate the differentiation matrix $\mathbf{D}$ along with the vector of quadrature weights $\mathbf{w}$ [65]. Here, the step length $h = t_f - t_0$ is assumed to be included in $\mathbf{D}$ and $\mathbf{w}$. The latter is used to approximate the definite integral of a function $g(t)$ as $\int_{t_0}^{t_f} g(t)dt \approx \sum_{j=1}^{s} w_j g(\tau_j)$.

The collocation naturally extends to a system of ODEs; that is, $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x})$. To simplify the notation, the state variables are assumed to be stacked in a row vector; that is, $\mathbf{x}, \mathbf{f} \in \mathbb{R}^{1\times n_x}$. Equation (16) becomes a matrix equation in $\mathbb{R}^{s\times n_x}$:

$$\mathbf{D} \cdot \begin{bmatrix} \mathbf{x}_0 \\ \mathbf{X} \end{bmatrix} = \mathbf{F}(\mathbf{X}), \tag{18}$$

where the rows of $\mathbf{X}$ and $\mathbf{F}$ correspond to one collocation point each. In turn, the columns of $\mathbf{X}$ and $\mathbf{F}$ represent one state variable and its corresponding right-hand side at all collocation points.

When the OCP (8a)–(8f) is transcribed, all functions and integrals have to be discretised consistently. Here, $k = 1, \ldots, m$ integration intervals $[t_{k-1}, t_k]$ are used with $0 = t_0 < t_1 \cdots < t_m = T$. The collocation order s_k can be different for each interval. Summing up the collocation points throughout all integration intervals results in a total of $M = l(m, s_m)$ discretisation points. The "linear index" l thereby corresponds to collocation node i in interval k:

$$l := l(k, i) = i + \sum_{\alpha=1}^{k-1} s_\alpha. \tag{19}$$

The following NLP results:

$$\min_{\mathbf{x}_\cdot, \mathbf{u}_\cdot} \quad \sum_{l=1}^{M} W_l \cdot L(\mathbf{x}_l, \mathbf{u}_l) \tag{20a}$$

$$\text{s.t.} \quad \mathbf{D}^{(k)} \cdot \begin{bmatrix} \mathbf{X}^{(k-1)}_{s_{k-1},\cdot} \\ \mathbf{X}^{(k)} \end{bmatrix} - \mathbf{F}\left(\mathbf{X}^{(k)}, \mathbf{U}^{(k)}\right) = 0, \quad k = 1, \ldots, m, \tag{20b}$$

$$\sum_{l=1}^{M} W_l \cdot \mathbf{g}(\mathbf{x}_l, \mathbf{u}_l) - \widehat{\mathbf{g}} \le 0, \tag{20c}$$

$$\mathbf{c}(\mathbf{x}_l, \mathbf{u}_l) \le 0, \quad l = 1, \ldots, M. \tag{20d}$$

The simple bounds (8e) and (8f) are imposed at each discretisation point and are not restated here. The vector of the "global" quadrature weights $\mathbf{W}$ results from stacking the vectors of the quadrature weights $\mathbf{w}^{(k)}$ of each interval k after removing the first element, which is zero.

The Lagrangian of the NLP (20a)–(20d) is the sum of the objective (20a) and all constraints (20b), (20c), and (20d), which are weighted by the Lagrange multipliers $\boldsymbol{\lambda}$. To simplify the notation, the Lagrange multipliers are grouped according to the problem structure. The $n_x \cdot s_k$ multipliers for the discretised dynamic constraints on each integration interval

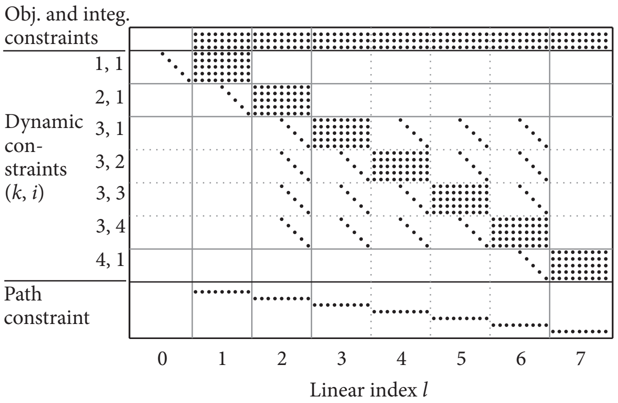


Figure 3: Sparsity structure of the Jacobian of the NLP resulting from the transcription of a continuous-time OCP by Radau collocation. Four collocation intervals are used, all with collocation order 1 except for the third interval which is of order 4. Characteristically for Radau collocation, the zeroth point only contributes linearly to the first dynamic constraint of the first interval. The variables at each discretisation point are ordered as $(\mathbf{u}_l^T, \mathbf{x}_l^T)^T$.

k are denoted by $\boldsymbol{\lambda}_d^{(k)}$, the n_g multipliers for the integral inequalities are stacked in $\boldsymbol{\lambda}_g$, and the n_c multipliers for the path constraints at each discretisation point l are gathered in the vector $\boldsymbol{\lambda}_{c,l}$.

The Lagrangian can be separated in the primal variables $\boldsymbol{\omega}_l = \boldsymbol{\omega}_i^{(k)} := (\mathbf{x}_i^{(k)}, \mathbf{u}_i^{(k)})$. The element Lagrangian $\mathscr{L}_i^{(k)}$ at collocation node l reads

$$\begin{aligned}\mathscr{L}_i^{(k)} &= W_l \cdot L(\boldsymbol{\omega}_l) \\ &\quad + \mathbf{x}_i^{(k)} \cdot \left(\sum_{j=1}^{N_k} \widetilde{D}_{ji}^{(k)} \boldsymbol{\lambda}_{d,j}^{(k)} + \delta_i^{(k)} \sum_{j=1}^{N_{k+1}} d_{0,j}^{(k+1)} \boldsymbol{\lambda}_{d,j}^{(k+1)} \right) \\ &\quad - \mathbf{f}\left(\boldsymbol{\omega}_i^{(k)}\right) \boldsymbol{\lambda}_{d,i}^{(k)} + W_l \cdot \boldsymbol{\lambda}_g^T \mathbf{g}(\boldsymbol{\omega}_l) + \boldsymbol{\lambda}_{c,l}^T \mathbf{c}(\boldsymbol{\omega}_l).\end{aligned} \tag{21}$$

The Lagrangian of the full NLP is obtained by summing these element Lagrangians. The Hessian of the Lagrangian thus is a perfect block-diagonal matrix with uniformly sized square blocks of size $(n_u + n_x)$.

Similarly, the Jacobian of the objective function and the constraints exhibits a sparse structure. Figure 3 provides an example. Exploiting the fact that only the derivatives of the model functions at each discretisation point are required to construct all derivative information of the NLP is crucial for an efficient solution [66]. In fact, the least number of model evaluations possible and thus a perfect sparsity exploitation are achieved by this procedure. Shared-memory parallel computing can be applied to further speed up the model evaluations, and a sparse matrix format has to be used in order not to be restricted by memory limitations for large problems.

The KKT system (12) is constructed from the Hessian of the Lagrangian and the Jacobian of the constraints. Therefore, it is very sparse in the case of direct collocation, and direct solvers are able to efficiently solve this linear system of equations.

2.4. Mesh Refinement. The idea of an iterative mesh refinement is to first solve a coarse approximation of the continuous-time OCP. This solution is used to identify regions where the discretisation needs refinement. Step-size refinement for relatively low-order collocation is considered here, and, consistently, the error is estimated by the local truncation error. For Radau collocation, a more detailed analysis of the convergence of the transcribed problem towards the continuous one is provided in [46].

The truncation error $\boldsymbol{\tau}$ is of order $s + 1$:

$$\mathbf{x} = \mathbf{x}^{(1)} + \boldsymbol{\tau}^{(1)}, \tag{22a}$$

$$\boldsymbol{\tau}^{(1)} = \mathbf{c} \cdot h^{s+1}. \tag{22b}$$

Here, $\mathbf{x}$ denotes the state at the end of the interval, that is, $\mathbf{x} := \mathbf{x}(t_k + h)$, and $\mathbf{x}^{(p)}$ is its approximation using p integration steps. The factors $\mathbf{c}$ depend on the model function $\mathbf{f}$ and therefore are not constant. However, over the time horizon of one integration step, assuming a constant value of $\mathbf{c}$ is reasonable. (Another common assumption is that the solution has a "memory" and thus $c_{i,k+1}/c_{i,k} \approx c_{i,k}/c_{i,k-1}$. In step-size control for ODE solvers, the two models may be combined [48].)

By subdivision of the interval into p equal steps, a more accurate approximation of the exact solution is obtained:

$$\mathbf{x} = \mathbf{x}^{(p)} + \boldsymbol{\tau}^{(p)}, \tag{23a}$$

$$\boldsymbol{\tau}^{(p)} = p \cdot \mathbf{c} \cdot \left(\frac{h}{p}\right)^{s+1} = \frac{\mathbf{c} \cdot h^{s+1}}{p^s} = \frac{\boldsymbol{\tau}^{(1)}}{p^s}. \tag{23b}$$

By equating the exact solution $\mathbf{x}$ in (22a) and (23a), with $p = 2$, an estimate for the absolute and the relative truncation error of the original approximation is obtained:

$$\boldsymbol{\tau}^{(1)} \approx \frac{\mathbf{x}^{(2)} - \mathbf{x}^{(1)}}{1 - 2^{-s}}, \qquad \widetilde{\tau}_i^{(1)} = \frac{\tau_i^{(1)}}{x_i^{(1)}}, \quad \text{for } i = 1, \ldots, n_x. \tag{24}$$

For the refined approximation, the estimate

$$\widetilde{\tau}_i^{(p)} \approx \frac{\tau_i^{(p)}}{x_i^{(1)}} \approx \frac{1}{p^s} \cdot \frac{\tau_i^{(1)}}{x_i^{(1)}} = \frac{\widetilde{\tau}_i^{(1)}}{p^s} \tag{25}$$

results. The magnitude of this relative error has to be smaller than the desired relative tolerance ε_{rel}. Solving for p yields the state-variable individual step refinement

$$p_i = \left\lceil k \cdot \left(\frac{\left|\widetilde{\tau}_i^{(1)}\right|}{\varepsilon_{\text{rel}}} \right)^{1/s} \right\rceil. \tag{26}$$

The "safety margin" k ensures that the desired tolerance is met on the new grid. The final step refinement is chosen as the maximum among all p_i.

In the context of optimal control, a value of $k < 1$ may be convenient. Since again an OCP is solved on the new grid, the control inputs may change. Thus, the exact solution of the ODEs as well as the error of the approximation changes. Due to this changing nature of the problem, it may be advantageous to refine the grid in multiple cautious iterations instead of trying to enforce the desired accuracy by a single refinement step.

2.5. Regularisation. In continuous-time OCPs, singular arcs are defined as time intervals over which the Hamiltonian becomes affine in at least one control input. (The Hamiltonian is the continuous-time equivalent of the Lagrangian. It can be shown that the KKT conditions of the transcribed problem are equivalent to the continuous first-order necessary conditions for optimality [45, 67].) On singular arcs, the optimal solution thus is not defined by the first-order optimality conditions, and the second-order sufficient conditions are not strictly satisfied since the second derivatives are zero. In indirect methods, a workaround is to consider time derivatives of the Hamiltonian of increasing order until the singular control input appears explicitly [68].

The presence of singular arcs introduces problems when a direct method is chosen to numerically solve the OCP. A good overview is provided in [69, Section 4.5]. Loosely spoken, the better the continuous problem is approximated, the more exactly the singular nature of the problem is revealed. Thus, a fine discretisation of the problem may lead to unwanted effects. Namely, spurious oscillations can be observed along singular arcs. This behaviour is intensified whenever the solution follows trajectory constraints.

In [69, Section 4.5], a method based on the "piecewise derivative variation of the control" is proposed to regularise the transcribed singular OCPs. Thereby, the square of the difference in the slope of each control input in two neighbouring intervals, that is, the local curvature, is added to the objective as a penalty term. The regularisation term for a scalar control input u is

$$L_{\text{reg}}(u_\cdot) := c_M \cdot \dot{\text{Var}}^2_{t_N}(u_\cdot) = c_M \cdot \frac{1}{2}\sum_{l=3}^{M-1} \left|s_{l+1} - s_l\right|^2, \qquad (27)$$

where $s_l = (u_l - u_{l-1})/(t_l - t_{l-1})$ is its slope in interval l. The summation starts at $l = 3$ since the first point is not a collocation point in Radau methods and thus the control input at this point does not have a meaning and is usually excluded from the NLP. The factor c_M is chosen as

$$c_M := \frac{c_{\text{reg}}}{(M-3)(M-1)^2}. \qquad (28)$$

The term $(M - 3)$ accounts for the number of summation terms, whereas $(M - 1)^2$ is an approximation of the average step size of the discretisation grid. This formulation scales the regularisation term according to the resolution of the transcription, such that the influence of the user-specified parameter c_{reg} is invariant with respect to a mesh refinement.

The solution of the regularised problem converges to the solution of the original problem for an increasingly fine resolution. It is further shown in the original literature that c_M goes to zero fast enough as $M \to \infty$ such as to ensure that the regularisation term stays bounded.

2.6. Optimal Control of Diesel Engines. The OCP of diesel engines can be cast in the form of the general OCP (8a)–(8f). The objective is to minimise the cumulative fuel consumption; that is, $L = \overset{*}{\dot{m}}_{\text{fuel}}$ in (8a). The dynamic constraints (8b) represent the air-path model. The control inputs comprise the air-path actuators, the fuel injected per cylinder and combustion cycle, as well as the signals that control the combustion, namely, the start of injection (SOI) and the injection pressure delivered by the common-rail system. The engine speed, its time derivative, and the desired load torque are introduced as time-variable parameters. Therefore, in the most general case, the vectors of the state variables, the control inputs, and the time-variable parameters read

$$\mathbf{x} = \begin{pmatrix} p_{\text{IM}} \\ p_{\text{EM}} \\ \omega_{\text{TC}} \\ p_1 \\ p_4 \\ \vartheta_{\text{IM}} \\ x_{\text{BG,IM}} \\ \vartheta_{\text{EMC}} \\ \vartheta_{\text{IMC}} \end{pmatrix}, \qquad \mathbf{u} = \begin{pmatrix} u_{\text{VGT}} \\ u_{\text{EGR}} \\ u_{\text{EF}} \\ m_{\text{fcc}} \\ \varphi_{\text{SOI}} \\ p_{\text{rail}} \end{pmatrix}, \qquad (29)$$

$$\boldsymbol{\pi} = \begin{pmatrix} N_{\text{eng}} \\ \dot{N}_{\text{eng}} \\ \widehat{T}_{\text{load}} \end{pmatrix}.$$

The fuel mass-flow is simply $\overset{*}{\dot{m}}_{\text{fuel}} = m_{\text{fcc}} \cdot N_{\text{eng}}/120 \cdot n_{\text{cyl}}$. The cumulative pollutant emissions are limited by the integral inequality constraints (8c); that is, $\mathbf{g} = (\overset{*}{\dot{m}}_{\text{NO}_x}, \overset{*}{\dot{m}}_{\text{soot}})^T$. The absolute limits $\widehat{\mathbf{g}}$ are calculated from the desired brake-specific values by multiplication with the integral of the nonnegative segments of the engine power $P_{\text{eng}} = N_{\text{eng}} \cdot \pi/30 \cdot \widehat{T}_{\text{load}}$. The desired load torque is imposed as a lower bound by a path constraint; that is, $\widehat{T}_{\text{load}}(t) - T_{\text{load}}(t) \leq 0$ for all t. The rationale for this formulation and a detailed description are provided in [70].

The simple bounds (8e) and (8f) represent various constraints. First, the physical actuator ranges need to be respected. The fuel mass is limited from below by zero and from above by the maximum injection quantity allowed for the engine at hand. Second, mechanical limits are imposed on several state variables such as the maximum turbocharger speed and maximum pressures and temperatures in the intake and exhaust manifolds. Finally, some of the control inputs are limited to a region in which the model is known to deliver plausible results. However, these limits are found not to be active at the optimal solution.

To initialise the OCP, a feedforward simulation of the model is performed. The control signals recorded during a test-bench run using a preseries ECU calibration are used. This initialisation is crucial in order not to obtain an infeasible QP in the first NLP iteration.

2.6.1. Engines with and without EGR. For engine A, which does not have an EGR system, the four state variables from p_1 to $x_{\text{BG,IM}}$, as well as the two control inputs u_{EGR} and u_{EF}, are not present in the model. Conversely, the full state and control vectors presented in (29) are required to represent the air path of engine B. However, since no satisfactory soot

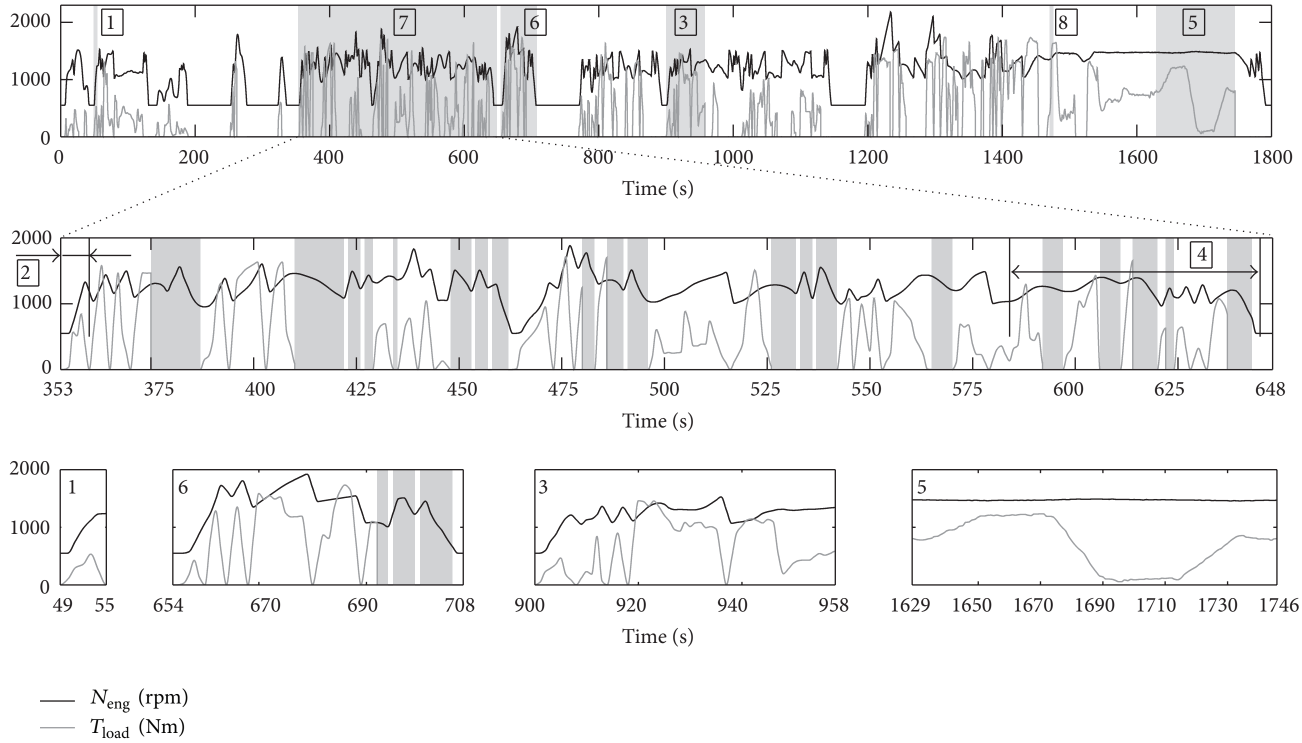


Figure 4: The full WHTC (top), with the individual test cases used in this paper indicated by the shaded areas. The test cycles are shown in the middle and bottom plots, scaled for engine A. Here, the shaded areas indicate drag phases. Test cycles 2 and 4 are contained within cycle 7. The detailed plot of test cycle 8, scaled for engine B, is provided in the left-hand plot in Figure 11.

model could be derived for this engine, the integral constraint for this emission species is not included in the problem formulation. To account for the most prominent influence on the soot emissions, the AFR is limited by the additional path constraint $\lambda_{\mathrm{AFR,min}} - \lambda_{\mathrm{AFR}}(t) \leq 0$ for all t.

The rail pressure defines a tradeoff between the soot emissions and the NO_x emissions. If the rail pressure is too low, the combustion is slow and incomplete, which causes high soot emissions. Conversely, this slow and cool combustion leads to less thermal NO_x formation. The negative effect of a low rail pressure on the combustion efficiency is almost outweighed by the reduced power consumption of the high-pressure pump of the common-rail system. Thus, if the soot emissions are not modelled and limited, there is no tradeoff for the rail pressure and the optimisation would always set it to its lower bound. For this reason, the rail pressure is excluded as a control input to the model for engine B. Instead, the values defined by the calibration map are used.

Dynamic Loops. For the engine with EGR, the optimisation terminates prematurely for all problem instances tested. Either the QP subproblem of some outer iteration is infeasible or the algorithm just diverges. The reason is the additional dynamic loop introduced by the EGR system. It interacts with the dynamic loop of the turbocharger and thus introduces a high degree of nonlinearity to the dynamic system. A similar observation is reported in [71], and the authors propose to break the loops and to apply a homotopy to close them again.

For the problem at hand, a more straightforward solution is implemented. The control inputs and the state variables are restricted to a small "stability region" around the initial trajectories. After solving this problem, the resulting trajectories are used as the new initial guess, and the stability region is relocated around these trajectories. This procedure is repeated until the solution lies entirely inside the current stability region. An SQP method is able to efficiently solve this sequence of related problems; see Section 3.1.2.

2.6.2. Driving Cycle and Test Cases. Various segments of the World-Harmonized Transient Cycle (WHTC) [72] are used as test cases. This driving profile prescribes the engine speed and the load torque over time. (For passenger cars and light commercial vehicles, a trajectory for the vehicle speed is usually prescribed. From this speed profile, an operating-point trajectory for the engine can be calculated by means of a vehicle emulation [73].) Figure 4 displays the full WHTC as well as the segments used here. Since the profile demands values for the engine speed and the load torque at each second only, shape-preserving cubic splines are used for the interpolation. Compared to a linear interpolation, this method smoothes the operating-point trajectory, which improves the numerical properties of the resulting OCP.

The shaded areas in the middle and bottom plots of Figure 4 indicate drag phases. During these intervals, the injection is cut off and the engine is motored at the prescribed

Table 2: Performance of the NLP solvers on test cycle 1, number of outer iterations, and overall time required for the solution (in brackets). For the discretisation with order 5^*, a piecewise-constant control was imposed by additional linear constraints. This variant imitates multiple shooting by resolving the state variables finer than the control inputs. The last row is the pure pseudospectral method. Symbols used are h (length of collocation intervals), s (collocation order), n_{NLP} (number of NLP variables), n_{DOF} (degrees of freedom in the NLP), and QN/EN (quasi/exact Newton method). For WORHP, only the exact Newton method is shown. For the IP method using a direct solver in KNITRO (KN-IPDIR), only the QN method could solve the problem, which is shown here. The IP-CG method always performed worse and thus is not shown. An accuracy of 10^{-6} is requested with respect to optimality and feasibility. If this accuracy was not achieved within 200 iterations, the optimisation was terminated, which is indicated by italic script. Bold script indicates the fastest solution for each case.

h	s	c_{reg}	n_{NLP}	n_{DOF}	SNOPT	IPOPT, QN	IPOPT, EN	WORHP	KN-SQP, QN	KN-SQP, EN	KN-IPDIR
0.25	1	0	225	61	**20 (3.5)**	39 (6.4)	13 (4.5)	18 (6.7)	66 (11.3)	58 (29.6)	54 (7.8)
0.25	3	10	657	190	69 (15.2)	40 (9.9)	**15 (9.8)**	18 (11.9)	151 (49.0)	65 (64.9)	70 (19.3)
0.25	5^*	0	1,089	60	**23 (7.1)**	30 (10.7)	15 (14.3)	21 (23.2)	136 (82.2)	112 (167.5)	137 (86.8)
3.00	15	1	279	78	42 (6.8)	62 (10.2)	**12 (4.8)**	21 (8.5)	*200 (43.8)*	*200 (107.1)*	*144 (32.3)*
6.00	30	0	279	80	154 (27.0)	82 (21.9)	**28 (10.7)**	27 (13.7)	*200 (45.9)*	*200 (118.2)*	*200 (45.9)*

speed. The individual test cases are characterised in the following list.

(1) Short, simple cycle used for first tests of the algorithms and for illustration purposes.

(2) Short segment with a singular arc. The effect of the regularisation is illustrated on this cycle.

(3) Longer, realistic driving profile, but without drag phases.

(4) Longer, realistic driving profile with many drag phases. The presence of motored phases renders the OCP more inhomogeneous and thus more difficult to be solved. Together with cycle 3, this test cycle is used to assess the convergence properties of the various algorithms.

(5) Long test cycle which is easy to be solved since almost no transients occur. The cycle is periodic; that is, it ends at the same operating point as it starts with. By a repetition of the cycle, large-scale OCPs can be constructed that retain the complexity of the single cycle.

(6) Longer, realistic cycle that is used for the experimental validation.

(7) Long, realistic cycle that is used for the experimental validation as well as for the identification of a causal control structure in Section 3.2.1.

(8) A single load step at almost constant engine speed used for the parametric study in Section 3.2.2.

3. Results and Discussion

The results are subdivided into three sections. The first one analyses crucial numerical aspects and states the conclusions that can be drawn from these tests. The second part covers the engineering aspect by presenting various ways of utilising the optimal solutions. The last section describes the experimental validation of the methodology, which indicates that all findings presented in this paper accurately transfer to the real engine.

3.1. Numerical Aspects. First, the results from all tests performed are presented. Based on these data, conclusions are drawn on what discretisation approach is suitable for the problem at hand and which numerical methods are preferable to solve the resulting NLPs.

3.1.1. Test Results. All NLP solvers introduced in Section 2.3.2 are tested for their convergence behaviour and their computational performance when applied to problems of varying complexity and size. Moreover, the basic effects of mesh refinement and regularisation are analysed.

Convergence Behaviour. Table 2 summarises the performance of all NLP solvers on the short, simple test cycle 1. Different discretisation approaches are tested, from first-order collocation to the pseudospectral method. The regularisation parameter c_{reg} is chosen for each case individually such that a smooth solution results. For the solver WORHP, only the exact Newton method is applied. None of the partitioned BFGS updates yields satisfactory results for the problem at hand.

Table 3 provides the same data for the more realistic test cycles 3 and 4. All methods implemented in KNITRO were not able to solve any of the problem instances to the required tolerance within 200 outer iterations. Therefore, this data is not shown. Similarly, IPOPT is most efficient with respect to the number of iterations and the solution time when the exact second derivatives are provided. The QN variant did not converge within 200 iterations for any of the tests on cycle 4.

The first and second partial derivatives of the model functions, from which the KKT matrix of the NLP is constructed, are calculated by forward finite differences (FFD). The convergence behaviour of the NLP solvers is not improved when more accurate derivatives are calculated by algorithmic differentiation (AD). Even the exact Newton methods require the same number of iterations when using AD instead of FFD. Therefore, AD is advantageous only if an implementation is available that is faster than FFD. All implementations of AD for Matlab tested, namely, ADiMat [74], ADMAT 2.0 (Cayuga Research, Waterloo, ON, Canada), INTLAB V6 [75], and the open-source implementation [76], are found to be at least a factor of 1.6 slower than FFD when

Table 3: Performance of the NLP solvers on test cycles 3 (top) and 4 (bottom). Only the exact Newton methods are shown for IPOPT and WORHP. The regularisation is chosen individually for all discretisation variants such that smooth control-input trajectories are obtained.

h	s	c_{reg}	n_{NLP}	n_{DOF}	SNOPT	IPOPT	WORHP
0.25	1	0	2,097	666	35 (29.9)	**16 (26.4)**	24 (54.5)
0.40	3	20	3,924	1,256	37 (114.8)	**19 (61.1)**	24 (108.1)
0.25	5^*	10	10,449	669	**24 (114.5)**	17 (153.6)	33 (338.9)
3.00	15	10	2,574	822	51 (120.8)	**18 (39.0)**	28 (69.7)
9.60	47	30	2,547	810	179 (554.6)	*200 (544.4)*	**26 (92.7)**
58.0	282	30	2,547	791	*200 (5,495.5)*	26 (448.2)	**24 (433.7)**
0.25	1	0	2,205	450	**45 (29.7)**	28 (46.7)	37 (117.4)
0.40	3	500	4,113	1,199	114 (311.5)	**29 (89.0)**	49 (196.5)
0.25	5^*	30	10,989	617	**66 (386.1)**	76 (680.2)	33 (405.8)
3.00	15	200	2,709	790	197 (331.8)	*25*[a] *(65.2)*	**41 (104.5)**
10.1	50	200	2,709	768	*200 (659.9)*	**36 (92.3)**	58 (204.2)
61.0	300	300	2,709	797	*200 (6,219.3)*	*30*[a] *(2,493.7)*	**30 (781.6)**

[a] A segmentation error crashed the optimisation at that iteration.

the first and second partial derivatives of the model functions are calculated.

Large-Scale Performance. Test cycle 5 is repeated multiple times to construct a problem of increasing size that retains the same complexity. Therefore, the performance of the NLP solvers when applied to large problems can be assessed independently of their general convergence behaviour. First-order collocation on a uniform grid with a step size of 0.25 s is used, and the cycle is repeated 1 to 6 times. NLPs with around 4,000 to 25,000 variables result.

The main observations are summarised next. All solvers require a similar number of iterations to solve the differently sized problems. Even the QN methods perform well for the large-scale problems. Therefore, their poor performance on the more difficult test cases is not induced by the size of the problem to be solved, but rather by its complexity and nonlinearity.

The number of model evaluations per iteration is proportional to the number of discretisation points and thus to the size of the NLP. Therefore, the time required to solve the KKT system or to perform updates of a decomposition directly defines the computational performance of the solvers for large-scale problems. The time required for the (re)factorisations and for the dense algebra on the reduced problem in SNOPT grows with a power of about 2.5 with respect to the problem size. All other solvers work on the full but sparse KKT system and exhibit an approximately linear runtime. The proportionality factor between runtime and problem size differs by a factor of 3 from the QN method of IPOPT (fastest) to the exact Newton method in WORHP (slowest). In WORHP, the linear solver has to recalculate or update the indefinite factorisation of the KKT system at the beginning of each outer iteration.

For the full-matrix approaches just described, the fraction of the overall solution time required for the model evaluations, which are performed in parallel on four cores, is between 40% (KNITRO) and 70% (IPOPT). For SNOPT, this fraction is below 1% for large problems.

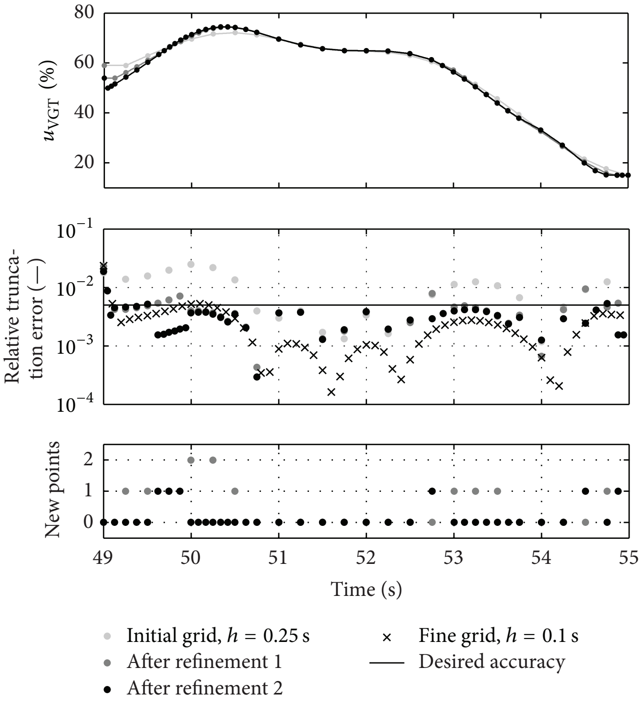


Figure 5: Mesh refinement on test cycle 1. First-order collocation is used, and the initial step size is 0.25 s. Two refinements are performed with $k = 0.5$ and $k = 1$, respectively.

Mesh Refinement. Figure 5 illustrates the effect of the mesh refinement on test cycle 1. First-order collocation is used, and a relative tolerance of $5 \cdot 10^{-3}$ is requested for the ODE solution. This accuracy is achieved by a uniform discretisation with a step size of 0.1 s. WORHP requires 21 iterations and 13.8 s for the solution of the resulting NLP.

When mesh refinement is applied, an initial step size of 0.25 s is used and two refinements are performed. The three solution runs of the coarse initial problem and the two refined problems require 17, 6, and 5 iterations and a total

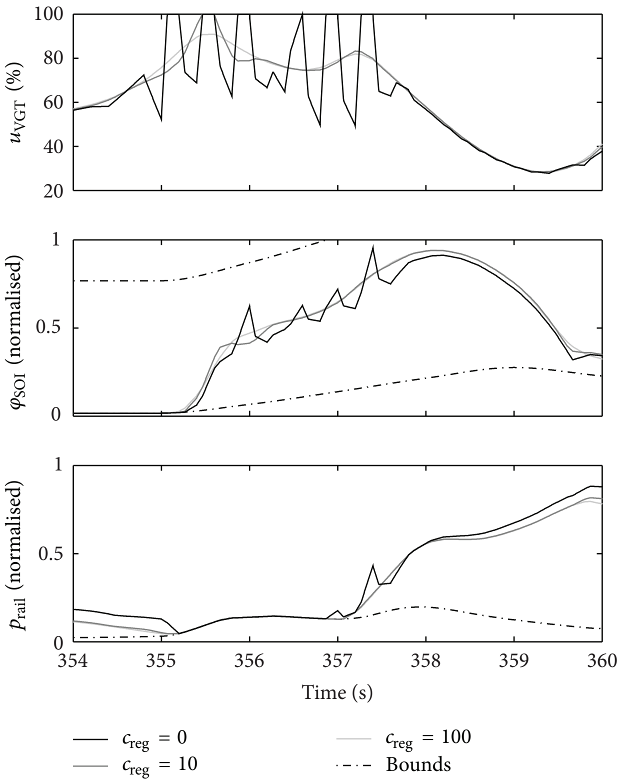


FIGURE 6: Regularisation of the control inputs, test cycle 2. The fuel consumption is reduced by 2.676% by the nonregularised solution and by 2.567% when a regularisation with $c_{\text{reg}} = 100$ is applied. Normalised signals are shown for reasons of confidentiality. A higher value of φ_{SOI} indicates an earlier injection.

of 11.0 s. After the second refinement, the desired accuracy is achieved, and the problem is discretised into 45 intervals only, as compared to the 60 intervals of the uniform discretisation.

When this procedure is applied to test cycle 7, the solution time is reduced from 2609 to 1015 s (or from around 43 to 17 minutes). IPOPT requires 3234 s to solve the uniformly discretised test cycle 7. However, in contrast to the SQP method implemented in WORHP, IPOPT is not able to exploit the good initialisation of the refined problems. In fact, the solution of the initial and the first refined problems requires the same time as the one-off solution of the uniformly discretised problem.

Regularisation. Figure 6 shows the optimal trajectories of the control inputs when a second-order collocation on a uniform grid with 0.2 s step size is applied to test cycle 2. As to be expected, oscillations occur mainly in the region where the trajectory constraint for the rail pressure is active. The fuel consumption and the pollutant emissions predicted by the optimisation are checked by an accurate forward simulation, which yields the same results. For a finer discretisation, faster oscillations result. Also when only the state variables are resolved more accurately, the oscillations persist.

Consequently, an oscillatory solution actually is optimal. Possibly, the gas-exchange losses are slightly reduced by the oscillations of the pressure in the exhaust manifold induced by the oscillations of the VGT. At the same time, the intake pressure is hardly affected due to the slow dynamics of the turbocharger. Therefore, the air mass-flow remains the same and the soot emissions do not increase.

Since a fast oscillating actuation of the mechanical actuators is not sustainable, regularisation has to be applied. As Figure 6 shows, the regularisation does not change the general shape of the solution. In particular, the solution is identical in the nonsingular regions. Furthermore, the loss in fuel efficiency when requiring a smooth solution is negligible.

3.1.2. Discussion. On the simple test case 1, all discretisations and all methods to solve the resulting NLPs seem to work reasonably well. However, especially the QN methods converge faster for a local discretisation than for the pseudospectral approach. The trust-region method implemented in KNITRO always gets stuck at a small trust radius and thus converges only slowly towards the optimum. The line-search globalisation thus seems to be preferable for the problem at hand.

When more meaningful driving profiles such as test cycles 3 and 4 are considered, the QN methods become less efficient also for local discretisation schemes. Surprisingly, SNOPT still manages to solve most problem instances in less than 200 iterations. As the complexity and the problem size increase, the SQP method implemented in WORHP becomes more reliable and consistent than the IP method of IPOPT.

The pseudospectral approach yields a denser NLP as indicated in Figure 3. Although the Hessian of the Lagrangian is still block diagonal, the Jacobian of the constraints does not retain a near-diagonal shape. The decomposition performed by SNOPT as well as the direct linear solvers of the full-matrix approaches become disproportionally slow when the collocation order is increased but the problem size is not changed.

Combined with the effectiveness of a step-size refinement, local discretisation schemes seem to be preferable for the problem at hand. A local discretisation enables a finer resolution of the problem only where necessary, and an SQP solver is able to exploit the good initialisation of the refined problem. Conversely, the pseudospectral method can only increase the order of the collocation polynomial and thus always refines the approximation of the problem over the full time horizon.

Summarising these findings, a full-matrix approach for the solution of the KKT system utilising a direct linear solver should be combined with an exact Newton SQP method and a line-search globalisation. (The effect of the choice of the merit function or a filter was analysed, too. No unique trend could be observed that favours one approach. The problem at hand thus seems not to be susceptible to the Maratos effect [53, Section 15.5].) WORHP implements such a method, and in fact this solver is found to perform well on all test cases, as well as to adapt to large problems best. A relatively low-order collocation scheme and an iterative step-size refinement combine well with this type of NLP solver. Only for small problems arising, for example, in receding horizon control, a decomposition approach and a QN method such as those implemented in SNOPT prove to be more efficient.

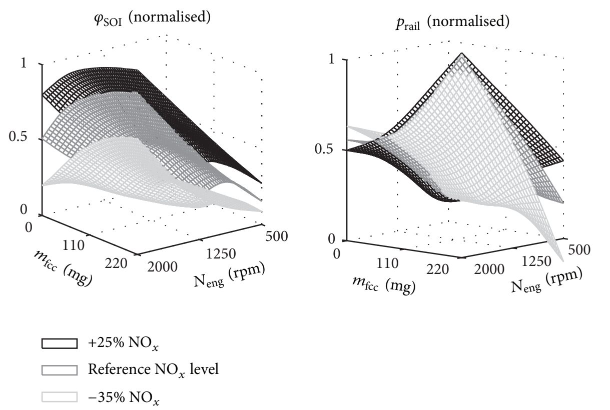


Figure 7: Maps obtained from the optimal control-trajectories and different NO_x levels relative to the one resulting from the current ECU calibration. Normalised signals are shown for reasons of confidentiality.

3.2. Engineering Aspects. Two ways of utilising the results from optimal control are proposed. On the one hand, the optimal solution may be used to identify all maps and even the control parameters of an entire feedback-control structure. On the other hand, control strategies for transient operation can be derived from parametric studies. The latter is illustrated by a case study in which the NO_x emissions of the engine with EGR have to be reduced over a load increase.

Another option is the repeated solution of a receding-horizon OCP online on the ECU. However, in contrast to model-predictive feedback control, where linear models may be used that are valid only around the reference maps [16], the full nonlinear model would have to be considered. Despite the application of custom-tailored algorithms, already the simpler optimisation problems encountered within model-predictive control are difficult to be solved in realtime due to the limited computational power and memory provided by the ECU [77]. Therefore, an online optimisation is not a feasible option at this time.

3.2.1. Model-Based Engine Calibration. Throughout this section, engine A is considered. From the solution of the OCP over a sufficiently long time horizon such as test case 7, implications for the control structure can be derived [34]. The optimal trajectories of the control inputs defining the combustion, that is, the SOI and the rail pressure, can be represented accurately by static maps over the engine operating range. The same finding applies to the boost pressure. Although these quantities might be chosen freely over time by the optimisation, values that can be scheduled over the engine speed and the injected fuel mass result. If a quasistationary representation of the air path is used, a static feedforward map for the VGT position is obtained from the solution of the corresponding OCP.

Two applications of these findings are presented here. First, the maps for the combustion-related control inputs can be derived for different emission levels. The maps for the SOI and the rail pressure for three different NO_x levels are shown in Figure 7. These maps can be parameterised by the requested emission level. On the ECU, the maps could be shifted adaptively, depending on the current performance of the ATS or according to the current driving situation.

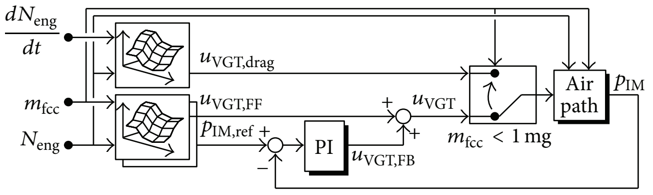


Figure 8: Structure of the boost-pressure controller.

The second application is the implementation of a feedback controller for the boost pressure based on the maps for the boost pressure and the static optimal VGT position. The former serves as reference to be tracked by the controller, and the latter is used as feedforward control signal. Figure 8 shows the structure of the control system. During drag phases, a pure feedforward control is applied. The map for the VGT position during motored operation is spanned by the engine speed and its derivative and is identified using the optimal data from the last 2 seconds of each drag phase [70]. Along with this controller defining the VGT position, the maps presented in Figure 7 are used for the SOI and the rail pressure.

The PI controller for the VGT position is implemented as

$$u_{\mathrm{VGT,FB}} = k_p \cdot e_{p_{\mathrm{IM}}}(t) + \frac{k_i}{10} \cdot \int_0^t e_{p_{\mathrm{IM}}}(\tau)\, d\tau, \qquad (30)$$

with $e_{p_{\mathrm{IM}}} = 10^{-4} \cdot (p_{\mathrm{IM}} - p_{\mathrm{IM,ref}})$. The scaling factors are used to provide a similar magnitude of the two controller parameters. A classical anti-reset windup scheme [78] is applied to handle actuator saturation and the purely feedforward operation during drag phases.

The optimal values for the two PI parameters, which are not scheduled over any quantity, may be obtained automatically. Here, a brute-force approach is proposed. The parameters are varied on a reasonable grid, and a forward simulation of the model is performed for all combinations. Figure 9 displays the results. The NO_x emissions are rather insensitive with respect to the choice of the PI parameters. Similarly, the soot emissions increase rapidly only if a too slow feedback controller is used. In this case, the boost-pressure buildup lags behind during load increases, resulting in low AFRs. An aggressive controller yields the lowest fuel consumption. However, the control signal overshoots and oscillates for this parameter set. Therefore, the parameter set which yields the closest representation of the optimal control-input trajectory is chosen.

The performances of the reference controller currently implemented, the optimal solution, and the fully causal control system derived from the optimal solution are illustrated in Figure 10. The causal controller is able to closely reproduce the optimal solution. Note that this causal control system is identified by a fully automated procedure requiring only the stationary measurement data for the identification of the engine model as input.

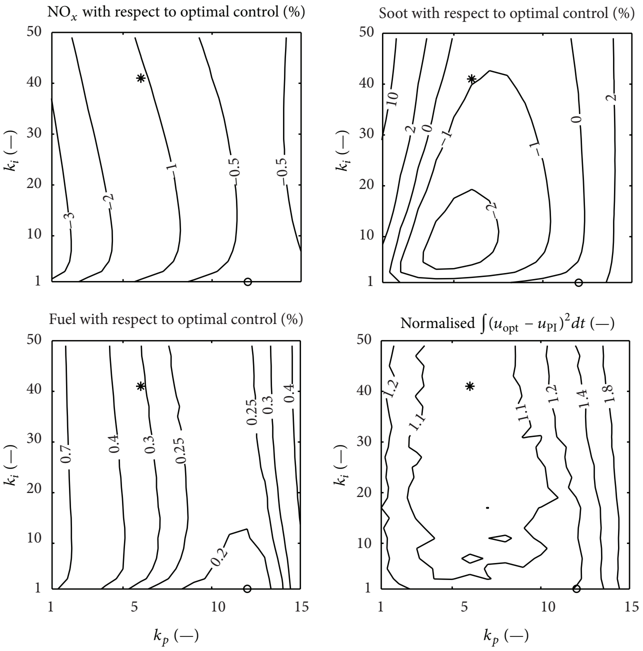


Figure 9: Selection of the parameter values for the PI controller. The circle indicates the fuel-optimal pair, while the asterisk denotes the values that result in the closest possible approximation of the optimal control-input trajectory.

3.2.2. Optimal Transient NO_x-Reduction Strategy. Several control inputs that define a tradeoff between the fuel consumption and the NO_x emissions are present on engine B. The EGR valve controls the flow of exhaust gas into the intake manifold. The burnt gas in the intake mixture is inert and reduces the temperature of the combustion zones in the cylinders, which in turn reduces the thermal NO_x formation. Conversely, the combustion becomes less efficient. The exhaust flap (EF) can be closed to increase the pressure difference over the EGR valve. A higher EGR mass-flow results, and also higher gas-exchange losses have to be overcome. Finally, a retardation of the injection yields a less efficient combustion and less NO_x emissions.

Particularly during transient operation, it is difficult to derive a control strategy that reduces the NO_x emissions to a desired level while maintaining the lowest possible fuel consumption. For a load increase, the additional constraint of a lower limit on the AFR has to be honoured in order not to produce large peaks of soot emissions. It is shown how optimal control can be used to derive a general control strategy. The single load step of test case 8 is considered; see the left-hand plot of Figure 11.

The following procedure is applied. First, optimal trajectories for the VGT and the SOI are derived while keeping the EGR valve closed and the EF fully open. This solution is used as reference for a successive reduction of the NO_x emissions. For each desired level of the NO_x emissions, the minimum fuel consumption is calculated by solving the corresponding OCP, with all four control inputs as free variables. Each resulting pair of cumulative NO_x emissions and fuel consumption is plotted in the right-hand plot of Figure 11. The resulting line defines the optimal tradeoff, that is, the Pareto front, between the NO_x emissions and the fuel consumption for the load step under consideration.

Figure 12 shows the corresponding control-input trajectories. The following control strategy can be derived. The optimisation does not close the EF at any point. Therefore, this is the least efficient way to reduce the NO_x emissions and should be avoided. During the load increase, the VGT as well as the EGR valve needs to be closed such that the AFR stays above the lower limit, which is chosen at 1.4 here. A feedforward part based on the gradient of the torque demand could be derived from the solution of the OCP. Finally, the higher NO_x emissions during the load step can be compensated by a transient shift of the SOI and a higher EGR rate during the stationary operation before and after the step.

By extending this case study to a more representative, longer time horizon, sufficient information could be collected to derive an overall controller calibration as presented for engine A in Section 3.2.1.

3.3. Experimental Validation. In order to validate the results from the dynamic optimisation on the engine test bench, three critical points have to be resolved. First, the control signals have to be transmitted to the ECU and the test-bench brake synchronously and sufficiently fast. A master process writes all signals to a shared memory section of the automation system at a frequency of 100 Hz. Using the iLinkRT protocol developed by AVL (AVL List GmbH, Graz, Austria) and ETAS (ETAS GmbH, Stuttgart, Germany), the control signals are transferred to an ETAS ES910.3 prototyping and interface module at the same frequency. This module immediately transfers the updated values to the ECU using the ETK interface (ETAS). Simultaneously, an additional process transfers the desired engine speed from the shared memory section to the controller (SPARC by HORIBA Ltd., Kyoto, Japan) of the test-bench dynamometer (HORIBA HD 700 LC). To this end, the proprietary OpenSIM CAN message protocol by HORIBA is used, which ensures a time-synchronous transfer at 100 Hz. These CAN messages, as well as all relevant signals of the ECU, are recorded at their natural sampling rate by INCA (ETAS), which runs on a host computer.

The second problem consists in obtaining meaningful and comparable results. When the engine is operated using the ECU, limits such as an operating-point dependent AFR limit are respected. Furthermore, a feedback controller is used to follow the desired load torque, resulting in deviations of up to 3%. Therefore, the engine does not exactly produce the torque desired by the driving cycle. In addition, the optimal solution has to provide the same braking torque during drag phases that results from the current control strategy implemented on the ECU. For these reasons, the effective torque delivered by the engine during a normal run is prescribed during the optimisation. Note that this different torque demand causes the change in the predicted fuel savings as compared to the values provided in the description of Figure 10. During

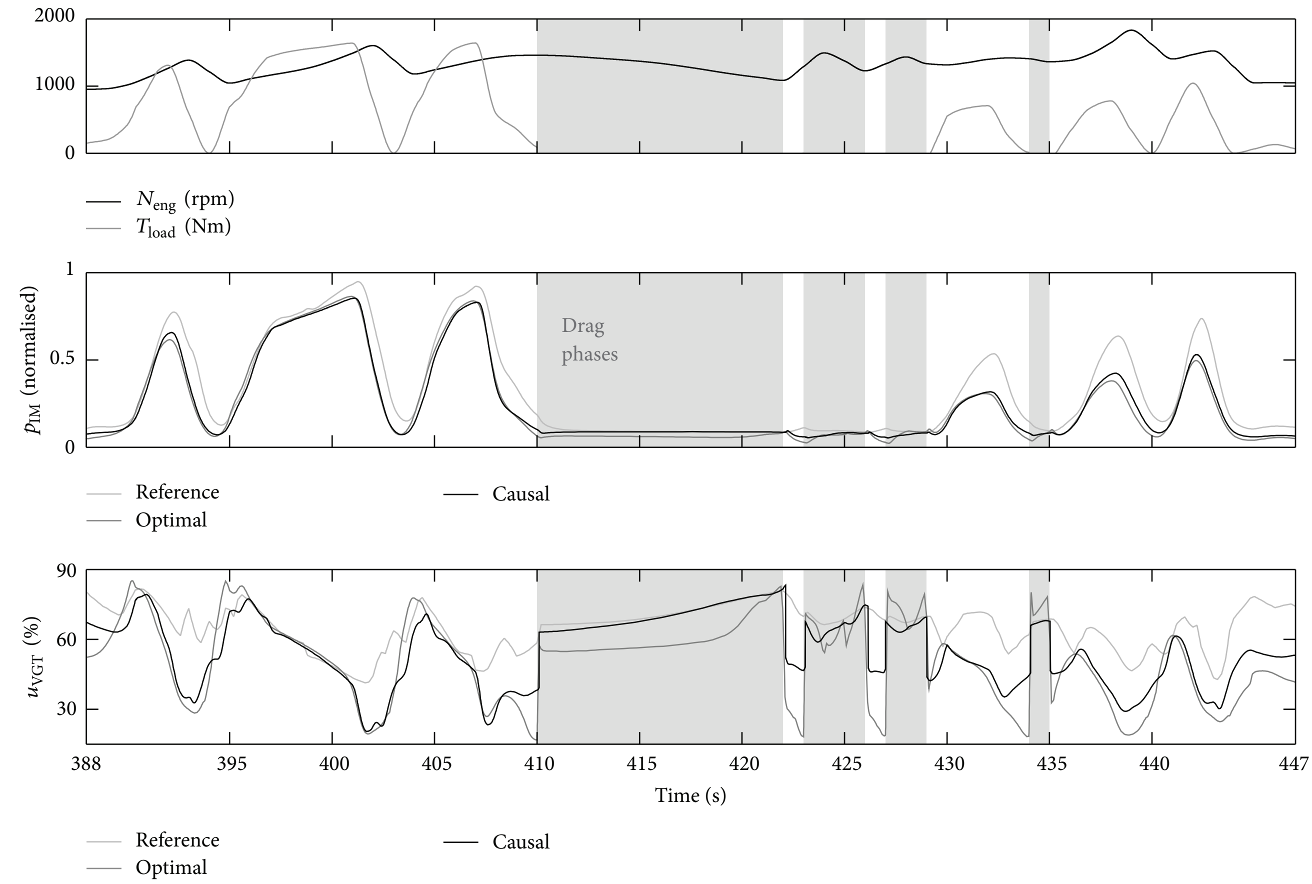
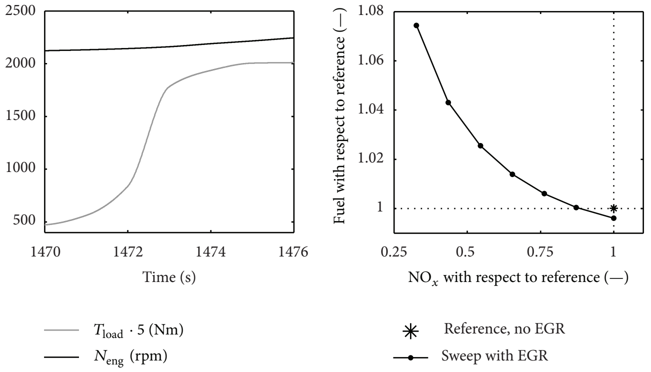


Figure 10: Comparison of the reference (the current ECU calibration), the optimal, and the causal control derived from the optimal solution. Results from a simulation of the model over a segment of test cycle 7 are shown. All three cases produce the same cumulative pollutant emissions. The optimal solution saves 2.21% fuel and the causal one 1.91%. Normalised signals are shown for reasons of confidentiality.



(a)

(b)

Figure 11: The driving profile of test case 8, scaled for engine B (a), and the optimal fuel-NO_x tradeoff for engine B over this cycle (b).

the validation run, the optimised control inputs are prescribed directly, and no bounds are applied.

Finally, the ECU does not inject exactly the amount of fuel demanded by the corresponding control signal. In fact, the injector maps are identified for a narrow operating region and the ECU estimation becomes increasingly inaccurate for larger deviations of the combustion-related control inputs from the current calibration. Contrarily, the model is identified using data recorded by an accurate external fuel scale. Therefore, the torque produced by the engine deviates from the desired one by up to 7%. To resolve this mismatch, a correction is applied after the first validation run. The fuel injection is updated according to a Willans approximation [1, Section 2.5.1]. The net torque is modelled as a time-variable, affine function of the fuel mass:

$$\begin{aligned} T_{\text{load}}(t) = {} & \eta_W(t) \cdot m_{\text{fcc}}(t) \\ & - T_0\left(N_{\text{eng}}(t), p_{\text{EM}}(t) - p_{\text{IM}}(t)\right). \end{aligned} \tag{31}$$

The loss torque T_0 is the sum of the friction, the inertia, and the gas-exchange work. The former two contributions are estimated using the corresponding submodels of the engine model, whereas the latter is estimated from the pressure difference over the engine measured during the first validation run.

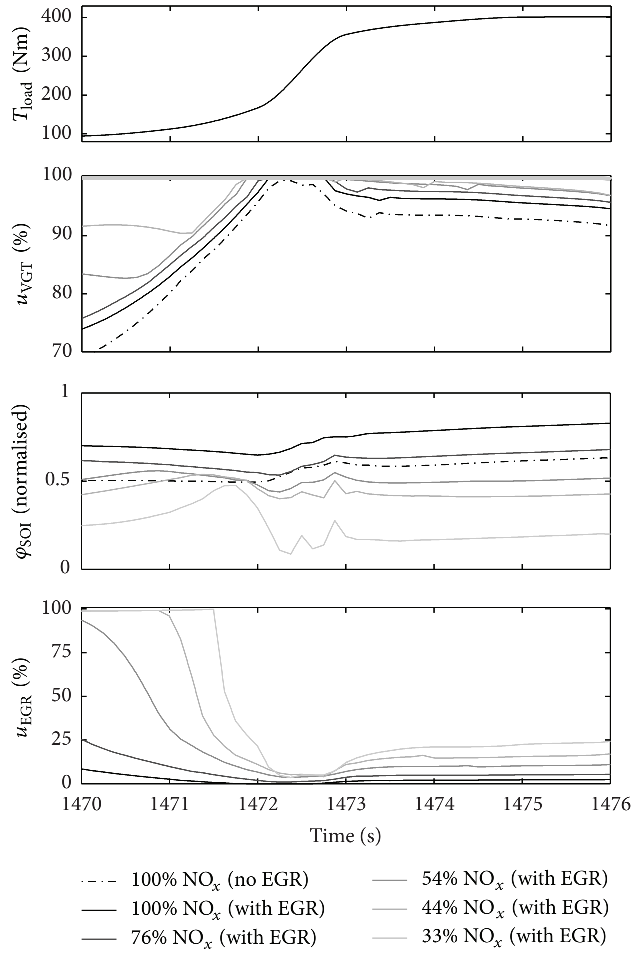


Figure 12: Control-input trajectories for the fuel-NO_x tradeoff depicted in the right-hand plot of Figure 11. The EGR valve is closed at 0%. The exhaust flap remains fully open for all cases and thus is not shown. Normalised signals are shown for reasons of confidentiality. A higher value of φ_{SOI} indicates an earlier injection.

The model is identified using the fuel mass prescribed for the first run, m_{fcc}, and the resulting torque T_{load}. Therefore, the updated fuel injection is calculated by

$$m_{\text{fcc,NEW}}(t) = m_{\text{fcc}}(t) \cdot \frac{\widehat{T}_{\text{load}}(t) - T_0(t)}{T_{\text{load}}(t) - T_0(t)}. \tag{32}$$

After the correction, the deviation of the torque produced by the engine from the desired load torque hardly ever exceeds 1%.

3.3.1. Results. For each of the two test cycles 6 and 7, two OCPs were solved. The first one required the emissions to remain at the level of the current engine calibration (I). The second one allowed a prescribed increase of the NO_x emissions (II). This increase is larger for test cycle 6 since for cycle 7 already high brake-specific (BS) NO_x emissions result from the current calibration. By performing this variation, not only the quantitative accuracy of the model is assessed, but also its ability to reproduce the tradeoff between fuel savings and NO_x emissions is evaluated.

Table 4: Results of the experimental validation. The changes relative to the values measured for the current engine calibration are shown.

Test cycle/OCP	6/I	6/II	7/I	7/II
Fuel				
Predicted	−1.28%	−2.85%	−1.52%	−2.49%
Measured	−0.85%	−2.39%	−1.42%	−2.50%
NO_x				
Predicted	0%	25%	0%	10%
Measured	−1.05%	21.27%	−3.64%	8.31%

For test cycle 6, the measurement is repeated 5 times for each set of control inputs to assess the reproducibility of the measurement. The maximum deviation of any of the 5 measurements from the average of all of them is used as a measure. For the BS fuel consumption, this deviation is 0.09%, and for the cumulative BS NO_x emissions, the figure is 0.64%. Based on this high reproducibility, the longer test cycle 7 was only measured once for each set of control-input trajectories.

Table 4 summarises the results from the experimental validation. On test cycle 6, the prediction overestimates the fuel savings but manages to predict the NO_x emissions accurately. An interesting quantity is the factor that describes by how much the NO_x emissions increase for a desired reduction of the fuel consumption:

$$k_{\text{ntf}} = -\frac{\Delta m_{\text{NO}_x}}{m_{\text{NO}_x,\text{ref}}} \cdot \frac{m_{\text{fuel,ref}}}{\Delta m_{\text{fuel}}}. \tag{33}$$

As can be extracted from the data in Table 4, the model predicts a factor of 15.9 when comparing cases I and II. The measured factor is 14.5.

On test cycle 7, the fuel savings are predicted accurately by the model. However, the increase of the NO_x emissions is underestimated. This shift of the model accuracy is due to the fact that the engine is operated in different operating regions on the two cycles. Cycle 6 comprises high-power operating points to the largest extent, whereas cycle 7 prescribes a low-power profile. For cycle 7, the predicted and the measured values of the NO_x-to-fuel factor k_{ntf} are 10.3 and 11.1, respectively.

Figure 13 shows the opacity of the exhaust gas measured for the reference and the two optimal solutions. As predicted by the model, the overall level remains the same and thus no active regeneration of the DPF becomes necessary. Furthermore, most instantaneous peaks are even slightly reduced.

4. Conclusion

A self-contained set of tools and numerical methods for the efficient solution of optimal control problems for diesel engines are presented. This framework enables the calculation of optimal trajectories of the control inputs over long driving profiles. These solutions provide sufficient information to derive complete dynamic engine calibrations. This

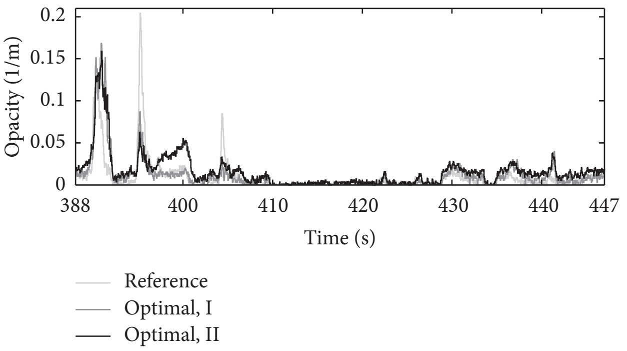


Figure 13: Opacity of the exhaust gas measured for the reference and the two optimal solutions. The same segment of test cycle 7 as in Figure 10 is shown.

fully automated model-based approach is illustrated for an engine without EGR. For an engine with EGR, it is shown how optimal control can be utilised to develop a control strategy that provides an optimal transient fuel-NO_x tradeoff. An experimental validation indicates that these findings accurately transfer to the real engines.

Further work should focus on a more detailed model for the soot emissions and on the analysis of the singular arcs occurring in the optimal solutions. The time-resolved control inputs could be replaced by the parameters of a prescribed control structure in the OCP. The simultaneous nature of the solution process would be preserved, but an optimal controller calibration could be obtained directly. From an engineering point of view, the exhaust-gas aftertreatment system should be included in the model to optimise the interaction between this system and the engine.

Nomenclature

Abbreviations and Indices

AD:	Algorithmic differentiation
AFR:	Air-to-fuel ratio
ATS:	Aftertreatment system
BG:	Burnt gas
BS:	Brake-specific
BVP:	Boundary-value problem
CG:	Conjugate gradient
CP:	Compressor
cyl:	Cylinder
DPF:	Diesel particulate filter
ECU:	Engine control unit
EF:	Exhaust flap
EGR:	Exhaust-gas recirculation
EM(C):	Exhaust manifold (casing)
eng:	Engine
FB, FF:	Feedback, feedforward
fcc:	Fuel per cylinder and cycle
FFD:	Forward finite differences
IC:	Intercooler, inequality-constrained
IM(C):	Intake manifold (casing)
IP:	Interior point
KKT:	Karush-Kuhn-Tucker
lin:	linear
NLP:	Nonlinear program (or programming)
ntf:	NO_x-to-fuel
OCP:	Optimal control problem
ODE:	Ordinary differential equation
opt:	Optimal
QN:	Quasi-Newton
QP:	Quadratic program (or programming)
quad:	Quadratic
ref:	Reference
SCR:	Selective catalytic-reduction system
SOI:	Start of injection
SQP:	Sequential quadratic programming
TC:	Turbocharger
VGT:	Variable-geometry turbine
WHTC:	World-Harmonized Transient Cycle.

Latin Symbols

A:	Area
a, b:	Generic model parameters
c:	Specific heat, path constraint
$\mathbf{D}$:	Differentiation matrix
e:	Control error
f:	Dynamic model function
F:	Objective of a generic NLP
g:	Integrand of a cumulative constraint
h:	Step size, inequality constraint of a generic NLP
k:	Generic model parameter
$L, \mathscr{L}$:	Integrand of the objective, Lagrangian
M:	Total number of collocation points
m:	Mass, number of collocation intervals
n:	Number
N:	Rotational speed (rpm)
p:	Pressure, QP step
R:	Gas constant
s:	Order of the collocation polynomial
T:	Torque, time interval
t:	Time
u:	Control input
V:	Volume
w, W:	Quadrature weights
x:	State variable, fraction.

Greek Symbols

η:	Efficiency
λ:	Lagrange multiplier, AFR
ξ:	Mass fraction
Π:	Pressure ratio
π:	Time-varying parameter
τ:	Barrier parameter, collocation points
φ:	Crank angle
Ψ:	Flow function
ω:	Rotational speed (rad/s), generic NLP variable.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

This work was partially funded by the Swiss Innovation Promotion Agency CTI under Grant no. 10808.1 PFIW-IW.

References

[1] L. Guzzella and C. H. Onder, *Introduction to Modeling and Control of Internal Combustion Engine Systems*, Springer, Berlin, Germany, 2nd edition, 2010.

[2] C. Rakopoulos and E. Giakoumis, *Diesel Engine Transient Operation*, Springer, London, UK, 2009.

[3] M. Lindgren and P.-A. Hansson, "Effects of transient conditions on exhaust emissions from two non-road diesel engines," *Biosystems Engineering*, vol. 87, no. 1, pp. 57–66, 2004.

[4] J. R. Hagena, Z. S. Filipi, and D. N. Assanis, "Transient diesel emissions: analysis of engine operation during a tip-in," SAE Technical Paper 2006-01-1151, 2006.

[5] J. F. Cassidy, "A computerized on-line approach to calculating optimum engine calibrations," SAE Technical Paper 770078, 1977.

[6] M. Hafner and R. Isermann, "Multiobjective optimization of feedforward control maps in engine management systems towards low consumption and low emissions," *Transactions of the Institute of Measurement and Control*, vol. 25, no. 1, pp. 57–74, 2003.

[7] C. Atkinson, M. Allain, and H. Zhang, "Using model-based rapid transient calibration to reduce fuel consumption and emissions in diesel engines," SAE Technical Paper 2008-01-1365, 2008.

[8] M. Guerrier and P. Cawsey, "The development of model based methodologies for gasoline IC engine calibration," SAE Technical Paper 2004-01-1466, 2004.

[9] I. Brahma, M. C. Sharp, and T. R. Frazier, "Empirical modeling of transient emissions and transient response for transient optimization," SAE Technical Paper 2009-01-1508, 2009.

[10] E. Alfieri, *Emissions-controlled diesel engine [Ph.D. thesis]*, ETH Zurich, 2009.

[11] A. de Risi, P. Carlucci, T. Donateo, and A. Ficarella, "A combined optimization method for common rail diesel engines," in *American Society of Mechanical Engineers, Internal Combustion Engine Division (Publication) ICE*, vol. 38, pp. 243–250, 2002.

[12] J. Wahlström, L. Eriksson, and L. Nielsen, "Controller tuning based on transient selection and optimization for a diesel engine with EGR and VGT," SAE Technical Paper 2008-01-0985, 2008.

[13] A. A. Malikopoulos, D. N. Assanis, and P. Y. Papalambros, "Real-time self-learning optimization of diesel engine calibration," *Journal of Engineering for Gas Turbines and Power*, vol. 131, no. 2, Article ID 022803, 2009.

[14] H. J. Ferreau, G. Lorini, and M. Diehl, "Fast nonlinear model predictive control of gasoline engines," in *Proceedings of the IEEE International Conference on Control Applications (CCA '06)*, pp. 2754–2759, Munich, Germany, October 2006.

[15] H. J. Ferreau, P. Ortner, P. Langthaler, L. del Re, and M. Diehl, "Predictive control of a real-world Diesel engine using an extended online active set strategy," *Annual Reviews in Control*, vol. 31, no. 2, pp. 293–301, 2007.

[16] L. del Re, P. Ortner, and D. Alberer, "Chances and challenges in automotive predictive control," in *Automotive Model Predictive Control*, chapter 1, pp. 1–22, Springer, Berlin, Germany, 2010.

[17] A. R. Dohner, "Optimal control solution of the automotive emission-constrained minimum fuel problem," *Automatica*, vol. 17, no. 3, pp. 441–458, 1981.

[18] Y. Urano, Y. Nakano, H. Takada, and M. Sugita, "Optimization technique for transient emission reduction of heavy duty diesel engine," SAE Technical Paper 2005-01-1099, 2005.

[19] A. Westlund and H. E. Angström, "Fast physical emission predictions for offline calibration of transient control strategies," SAE Technical Paper 2009-01-1778, 2009.

[20] C. Ericson, B. Westerberg, I. Odenbrand, and R. Egnell, "Characterisation and model based optimization of a complete diesel engine/SCR system," SAE Technical Paper 2009-01-0896, 2009.

[21] M. Benz, M. Hehn, C. H. Onder, and L. Guzzella, "Model-based actuator trajectories optimization for a diesel engine using a direct method," *Journal of Engineering for Gas Turbines and Power*, vol. 133, no. 3, Article ID 032806, 2011.

[22] R. Omran, R. Younes, J. C. Champoussin, D. Fedeli, F. Masson, and N. Guerrassi, "Genetic algorithm for dynamic calibration of engine's actuators," SAE Technical Paper 2007-01-1079, 2007.

[23] R. Omran, R. Younes, and J.-C. Champoussin, "Optimal control of a variable geometry turbocharged diesel engine using neural networks: applications on the ETC test cycle," *IEEE Transactions on Control Systems Technology*, vol. 17, no. 2, pp. 380–393, 2009.

[24] S. Kameswaran and L. T. Biegler, "Simultaneous dynamic optimization strategies: recent advances and challenges," *Computers and Chemical Engineering*, vol. 30, no. 10–12, pp. 1560–1575, 2006.

[25] J. T. Betts, *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming*, vol. 19, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, Pa, USA, 2nd edition, 2010.

[26] L. B. Rall, *Automatic Differentiation: Techniques and Applications*, vol. 120 of *Lecture Notes in Computer Science*, Springer, Berlin, Germany, 1981.

[27] A. Griewank and A. Walter, *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation*, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, Pa, USA, 2nd edition, 2008.

[28] C. Quérel, O. Grondin, and C. Letellier, "State of the art and analysis of control oriented NO_x models," SAE Technical Paper 2012-01-0723, 2012.

[29] J. Asprion, O. Chinellato, and L. Guzzella, "A fast and accurate physics-based model for the NO_x emissions of Diesel engines," *Applied Energy*, vol. 103, pp. 221–233, 2013.

[30] E. G. Pariotis, G. M. Kosmadakis, and C. D. Rakopoulos, "Comparative analysis of three simulation models applied on a motored internal combustion engine," *Energy Conversion and Management*, vol. 60, pp. 45–55, 2012.

[31] G. A. Weisser, *Modelling of combustion and nitric oxide formation for medium-speed DI diesel engines: a comparative evaluation of zero- and three-dimensional approaches [Ph.D. thesis]*, ETH Zurich, 2001.

[32] A. Schilling, A. Amstutz, C. H. Onder, and L. Guzzella, "A real-time model for the prediction of the NO_x emissions in DI diesel engines," in *Proceedings of the IEEE International Conference on Control Applications (CCA '06)*, pp. 2042–2047, Munich, Germany, October 2006.

[33] J. Asprion, O. Chinellato, and L. Guzzella, "Optimisation-oriented modelling of the NO_x emissions of a Diesel engine," *Energy Conversion and Management*, vol. 75, pp. 61–73, 2013.

[34] J. Asprion, O. Chinellato, C. H. Onder, and L. Guzzella, "From static to dynamic optimisation of Diesel-engine control," in *Proceedings of the 52nd IEEE Conference on Decision and Control*, Florence, Italy, December 2013.

[35] M. Diehl, H. G. Bock, H. Diedam, and P.-B. Wieber, "Fast direct multiple shooting algorithms for optimal robot control," in *Fast Motions in Biomechanics and Robotics*, M. Diehl and K. Mombaur, Eds., vol. 340 of *Lecture Notes in Control and Information Sciences*, pp. 65–93, Springer, Berlin, Germany, 2006.

[36] J. T. Betts, "Survey of numerical methods for trajectory optimization," *Journal of Guidance, Control, and Dynamics*, vol. 21, no. 2, pp. 193–207, 1998.

[37] A. V. Rao, "A survey of numerical methods for optimal control," *Advances in the Astronautical Sciences*, vol. 135, pp. 497–528, 2009.

[38] B. A. Conway, "A survey of methods available for the numerical optimization of continuous dynamic systems," *Journal of Optimization Theory and Applications*, vol. 152, no. 2, pp. 271–306, 2012.

[39] R. W. H. Sargent, "Optimal control," *Journal of Computational and Applied Mathematics*, vol. 124, no. 1-2, pp. 361–371, 2000.

[40] L. T. Biegler, "An overview of simultaneous strategies for dynamic optimization," *Chemical Engineering and Processing*, vol. 46, no. 11, pp. 1043–1053, 2007.

[41] G. Bashein and M. Enns, "Computation of optimal controls by a method combining quasi-linearization and quadratic programming," *International Journal of Control*, vol. 16, pp. 177–187, 1972.

[42] L. T. Biegler, "Solution of dynamic optimization problems by successive quadratic programming and orthogonal collocation," *Computers and Chemical Engineering*, vol. 8, no. 3-4, pp. 243–247, 1984.

[43] D. Tieu, W. R. Cluett, and A. Penlidis, "A comparison of collocation methods for solving dynamic optimization problems," *Computers and Chemical Engineering*, vol. 19, no. 4, pp. 375–381, 1995.

[44] A. L. Herman and B. A. Conway, "Direct optimization using collocation based on high-order Gauss-Lobatto quadrature rules," *Journal of Guidance, Control, and Dynamics*, vol. 19, no. 3, pp. 592–599, 1996.

[45] D. A. Benson, G. T. Huntington, T. P. Thorvaldsen, and A. V. Rao, "Direct trajectory optimization and costate estimation via an orthogonal collocation method," *Journal of Guidance, Control, and Dynamics*, vol. 29, no. 6, pp. 1435–1440, 2006.

[46] S. Kameswaran and L. T. Biegler, "Convergence rates for direct transcription of optimal control problems using collocation at Radau points," *Computational Optimization and Applications*, vol. 41, no. 1, pp. 81–126, 2008.

[47] D. Garg, M. Patterson, W. W. Hager, A. V. Rao, D. A. Benson, and G. T. Huntington, "A unified framework for the numerical solution of optimal control problems using pseudospectral methods," *Automatica*, vol. 46, no. 11, pp. 1843–1851, 2010.

[48] E. Hairer and G. Wanner, "Stiff differential equations solved by Radau methods," *Journal of Computational and Applied Mathematics*, vol. 111, no. 1-2, pp. 93–111, 1999.

[49] J. C. Butcher, *Numerical Methods for Ordinary Differential Equations*, John Wiley & Sons, Chichester, UK, 2003.

[50] C. L. Darby, W. W. Hager, and A. V. Rao, "Direct trajectory optimization using a variable low-order adaptive pseudospectral method," *Journal of Spacecraft and Rockets*, vol. 48, no. 3, pp. 433–445, 2011.

[51] C. L. Darby, W. W. Hager, and A. V. Rao, "An *hp*-adaptive pseudospectral method for solving optimal control problems," *Optimal Control Applications & Methods*, vol. 32, no. 4, pp. 476–502, 2011.

[52] J. F. Bonnans, J. C. Gilbert, C. Lemaréchal, and C. A. Sagastizábal, *Numerical Optimization: Theoretical and Practical Aspects*, Springer, Berlin, Germany, 2nd edition, 2006.

[53] J. Nocedal and S. J. Wright, *Numerical Optimization*, Springer Series in Operations Research and Financial Engineering, Springer, New York, Second edition, 2006.

[54] D. C. Liu and J. Nocedal, "On the limited memory BFGS method for large scale optimization," *Mathematical Programming*, vol. 45, no. 3, pp. 503–528, 1989.

[55] "The HSL mathematical software library," http://www.hsl.rl.ac.uk.

[56] "MUMPS homepage," http://graal.ens-lyon.fr/MUMPS.

[57] "PARDISO homepage," http://www.pardiso-project.org .

[58] P. E. Gill and E. Wong, "Sequential quadratic programming methods," in *Mixed Integer Nonlinear Programming*, J. Lee and S. Leyffer, Eds., vol. 154 of *The IMA Volumes in Mathematics and Its Applications*, pp. 147–224, Springer, New York, NY, USA, 2012.

[59] P. E. Gill, W. Murray, and M. A. Saunders, "SNOPT: an SQP algorithm for large-scale constrained optimization," *SIAM Review*, vol. 47, no. 1, pp. 99–131, 2005.

[60] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical Programming*, vol. 106, no. 1, pp. 25–57, 2006.

[61] "METIS—serial graph partitioning and fill-reducing matrix ordering," http://glaros.dtc.umn.edu/gkhome/metis/metis/overview.

[62] C. Büskens and D. Wassel, "The ESA NLP solver WORHP," in *Modeling and Optimization in Space Engineering*, G. Fasano and J. D. Pintér, Eds., vol. 73 of *Optimization and Its Applications*, pp. 85–110, Springer, New York, NY, USA, 2013.

[63] R. H. Byrd, J. Nocedal, and R. A. Waltz, "KNITRO: an integrated package for nonlinear optimization," in *Large-Scale Nonlinear Optimization*, vol. 83 of *Nonconvex Optimization and Its Applications*, pp. 35–59, Springer, New York, NY, USA, 2006.

[64] C. Canuto, M. Y. Hussaini, A. Quarteroni, and T. A. Zang, *Spectral Methods: Fundamentals in Single Domains*, Springer, Berlin, Germany, 2006.

[65] J.-P. Berrut and L. N. Trefethen, "Barycentric Lagrange interpolation," *SIAM Review*, vol. 46, no. 3, pp. 501–517, 2004.

[66] M. A. Patterson and A. V. Rao, "Exploiting sparsity in direct collocation pseudospectral methods for solving optimal control problems," *Journal of Spacecraft and Rockets*, vol. 49, pp. 364–377, 2012.

[67] J. R. Cloutier and C. N. D'Souza, "Unification of the direct and indirect approaches to optimal control," AIAA Meeting Papers Archive 481 AIAA-92-4529-CP, 1992.

[68] H. P. Geering, *Optimal Control with Engineering Applications*, Springer, Berlin, Germany, 2007.

[69] A. L. Schwartz, *Theory and implementation of numerical methods based on Runge-Kutta integration for solving optimal control problems [Ph.D. thesis]*, University of California, Berkeley, Calif, USA, 1996.

[70] J. Asprion, C. H. Onder, and L. Guzzella, "Including drag phases in numerical optimal control of Diesel engines," in *Proceedings of the 7th IFAC Symposium on Advances in Automotive Control*, Tokyo, Japan, 2013.

[71] S. Gros, M. Zanon, and M. Diehl, "A relaxation strategy for the optimization of airborne wind energy systems," in *European Control Conference (ECC '13)*, Zurich, Switzerland, 2013.

[72] E. G. Giakoumis and A. I. Alafouzos, "Comparative study of turbocharged diesel engine emissions during three different transient cycles," *International Journal of Energy Research*, vol. 34, no. 11, pp. 1002–1015, 2010.

[73] T. Ott, C. Onder, and L. Guzzella, "Hybrid-electric vehicle with natural gas-diesel engine," *Energies*, vol. 6, pp. 3571–3592, 2013.

[74] C. H. Bischof, H. M. Bücker, A. Vehreschild, and J. Willkomm, *Automatic Differentiation for Matlab (ADiMat)*, MATLAB-Day, Aachen, Germany, 2012.

[75] S. M. Rump, "Verification methods: rigorous results using floating-point arithmetic," *Acta Numerica*, vol. 19, pp. 287–449, 2010.

[76] M. Fink, "Automatic differentiation for Matlab," 2007, http://www.mathworks.com/matlabcentral/fileexchange/15235.

[77] G. Stewart, F. Borrelli, J. Pekar, D. Germann, D. Pachner, and D. Kihas, "Toward a systematic design for turbocharged engine control," in *Automotive Model Predictive Control*, chapter 14, pp. 211–230, Springer, Berlin, Germany, 2010.

[78] M. V. Kothare, P. J. Campo, M. Morari, and C. N. Nett, "A unified framework for the study of anti-windup designs," *Automatica*, vol. 30, no. 12, pp. 1869–1883, 1994.