A computationally efficient Kalman filter based estimator for updating look-up tables applied to NOx estimation in diesel engines

C Guardiola, B Pla, D Blanco-Rodriguez and Lars Eriksson

Linköping University Post Print

N.B.: When citing this work, cite the original article.

Original Publication:

Copyright: Elsevier / International Federation of Automatic Control http://www.ifac-control.org/
Postprint available at: Linköping University Electronic Press http://urn.kb.se/resolve?urn=urn:nbn:se:liu:diva-102388
A computationally efficient Kalman filter based estimator for updating look-up tables applied to NO\textsubscript{x} estimation in diesel engines

C. Guardiola\textsuperscript{a}, B. Pla\textsuperscript{a}, D. Blanco-Rodriguez\textsuperscript{a,\*}, L. Eriksson\textsuperscript{b}

\textsuperscript{a} CMT Motores Térmicos, Universitat Politécnica de València, Camino de Vera s/n, E-46022 Valencia, Spain
\textsuperscript{b} Vehicular Systems, Department of Electrical Engineering, Linköping University, S-581 83 Linköping, Sweden

\textbf{A R T I C L E  I N F O}

Article history:
Received 23 July 2012
Accepted 21 June 2013
Available online 7 August 2013

Keywords:
NO\textsubscript{x}
Kalman filter
Adaptive model
Look-up tables
Diesel

\textbf{A B S T R A C T}

NO\textsubscript{x} estimation in diesel engines is an up-to-date problem but still some issues need to be solved. Raw sensor signals are not fast enough for real-time use while control-oriented models suffer from drift and aging. A control-oriented gray box model based on engine maps and calibrated off-line is used as a benchmark model for NO\textsubscript{x} estimation. Calibration effort is important and engine data-dependent. This motivates the use of adaptive look-up tables. In addition to, look-up tables are often used in automotive control systems and there is a need for systematic methods that can estimate or update them on-line. For that purpose, Kalman filter (KF) based methods are explored as having the interesting property of tracking estimation error in a covariance matrix. Nevertheless, when coping with large systems, the computational burden is high, in terms of time and memory, compromising its implementation in commercial electronic control units. However, look-up table estimation has a structure, that is here exploited to develop a memory and computationally efficient approximation to the KF, named Simplified Kalman filter (SKF). Convergence and robustness is evaluated in simulation and compared to both a full KF and a minimal steady-state version, that negates the variance information. SKF is used for the online calibration of an adaptive model for NO\textsubscript{x} estimation in dynamic engine cycles. Prediction results are compared with the ones of the benchmark model and of the other methods. Furthermore, actual online estimation of NO\textsubscript{x} is solved by means of the proposed adaptive structure. Results on dynamic tests with a diesel engine and the computational study demonstrate the feasibility and capabilities of the method for an implementation in engine control units.

© 2013 Elsevier Ltd. All rights reserved.

1. Introduction

Legislation on emissions strengthens the importance of electronic controls and embedded software in automotive engines (Ebert & Jones, 2009). Focusing on nitrogen oxides (NO\textsubscript{x}) emissions, selective catalyst reduction system (SCR) or lean NO\textsubscript{x} trap (LNT) requires from reliable information about exhaust gas content, the former for an appropriate urea dosing (Twigg, 2007) and the latter for a proper regeneration (Chen, Wang, Haskara, & Zhu, 2012). Commercial electronic control units (ECU) are mainly programmed with fixed maps and feedback for emissions control is often far from the exhaust, e.g. mass air flow or boost pressure sensor. This argument can be extended to soot or CO\textsubscript{2}, while exhaust oxygen measurement is more usual.

An accurate and fast NO\textsubscript{x} signal would permit its usage for real-time tasks but still dynamic issues must be solved. NO\textsubscript{x} sensors have been commonly used in test benches, e.g. gas analyzers or fast measurement systems. But until the appearance of zirconia-based (ZrO\textsubscript{2}) potentiometric sensors, there had been no choice of using these in commercial engines, because of cost, size and response limitations (Kato, Nakagaki, & Ina, 1996; Zhuiykov & Miura, 2007). However, even with the sensor there are problems using the raw output signal in real-time functions, in particular the delay from engine to sensor and the response time of the sensor; see e.g. Manchur and Checkel (2005) for an approach that addresses the effects of NO\textsubscript{x} sensor dynamics. The use of observers is an effective solution for avoiding such effects: e.g. Hsieh and Wang (2011), Desantes, Luján, Guardiola, and Blanco-Rodriguez (2011), Payri, Guardiola, Blanco-Rodriguez, Marez, and Corrente (2012), and Alberer and del Re (2009) use ZrO\textsubscript{2} sensors for estimating NO\textsubscript{x} and oxygen, or Höckerdal, Frisk, and Eriksson (2009) and Grünbacher, Kefer, and del Re (2005) apply an extended Kalman filter (EKF) to estimate other relevant engine quantities. Anyway, a fast model is needed for the observation and sensor behavior is non-linear as studied in Galindo, Serrano, Guardiola, Blanco-Rodriguez, and Cuadrado (2011). For illustrating sensor response, see Fig. 1: actual NO\textsubscript{x} is expected to respond instantaneously when performing start of injection (SOI) steps, delay and time response in the sensor are attributed to the sensor.
Without NOₓ sensors, modeling (or virtual sensing) is possible, see e.g. Schilling, Amstutz, and Guzzella (2008) that develops a real time NOₓ model using maps, Winkler-Ebner, Hirsch, Del Re, Klinger, and Mistelberger (2010) that designs a virtual NOₓ sensor for selective catalyst reduction (SCR) control and diagnosis or other examples. Problem is highly non-linear and some examples for coping with this are Takagi–Sugeno fuzzy models (Lughofer, Macian, Guardiola, & Klement, 2011; Takagi & Sugeno, 1985), Hammerstein–Wiener (Falck et al., in press), or neural networks (Yen & Michel, 1991). Nevertheless, in commercial engine ECUs the prevailing approach is to use look-up tables to model nonlinear and operating point dependent behaviors because of the simple programming although it attaches a big calibration effort. These models need to be calibrated but in general will suffer from drift problem. Section 3 presents a gray box model used as benchmark for comparison latter.

A solution for drift correction is adaptive modeling. Main algorithms for online adaptation are recursive least squares (RLS) and Kalman filter (KF). Although both copes with system drift and aging, the latter tracks estimation aging by solving the statistics of the estimation error at every iteration (by the covariance matrix). This paper analyzes the computational aspects of look-up table updating with the KF. The main contribution is a new table based update, as described in Guardiola, Pla, Blanco-Rodriguez, and Cabrera (2013). Look-up tables updating is treated in Section 3 presenting a gray box model used as benchmark for comparison latter.

2. Experimental setup and engine data

Experimental data is obtained from a twin sequential-parallel turbocharged diesel engine. This engine is a 2.2-liter 4-cylinder common rail. Engine specifications are shown in Table 1. Conventional sensor set is used (mass air flow, boost pressure, etc.) and the following extra-sensors are added: a ZrO2 multilayer based NOₓ sensor (Kato et al., 1996) giving a twofold measurement of NOₓ (yanox) and oxygen in the tailpipe and a gas analyzer (HORIBA, 2001) in a long line connecting intake manifold and tailpipe as steady-state calibration standard. These sensors are used for collecting dynamic data and some of them as inputs for the model, removing the need of observing some of the variables.

A rapid prototyping system is connected via ETK to a bypass-allowed ECU, permitting commanding and receiving signals by means of coupling a real time system via CAN. This allows easy access to injection parameters (injection pressure or start of injection SOI), boost and EGR control set points, engine calibration and possibilities to easily test new programs and routines.

The test campaign includes data for steady-state and transient tests for the benchmark model calibration. Steady-state ones cover around 300 operating points, with nominal control inputs. Operating points are fixed and signals stability is checked for storing steady-state points. Transient tests include variations on the actuators and two dynamic cycles: a designed SDMP and NEDC. SDMP performs sharp variations on injected fuel mass and speed and is shown in Fig. 2. This is a good test bench for comparing after look-up table estimation results with the ones given by the model. NEDC is also used as application. For more details about experimental setup see Lughofer et al. (2011). SOI steps are used for NOₓ sensor characterization; the interested reader can go to Galindo et al. (2011) for further information.

Table 1

| Engine technical data. For more information see Galindo et al. (2011). |
|------------------|------------------|
| Stroke (S)       | 96 mm            |
| Bore (D)         | 85 mm            |
| S/D              | 1.129            |
| Number of cylinders (z) | 4               |
| Displacement     | 2179 cm³         |
| EGR              | HP               |
| Turbocharging system | Sequential parallel (Galindo et al., 2007) |
| Valves by cylinder | 4              |
| Maximum power    | 125 kW at 4000 rpm |
| Compression ratio | 17:1            |

Fig. 1. NOₓ sensor response to SOI steps. NOₓ presents a clear delay and filtering with respect to SOI signal. Dynamic response could be fitted with a first order delayed discrete filter. SOI unit is crank angle degree before the top dead center, while NOₓ is measured in ppm. Delay is in the order of 1 s while response time is about 0.75 s.

Fig. 2. SDMP test profile with sharp variations in mₙ, n and NOₓ.
3. A control-oriented model for NOx estimation

Literature about NOx modeling in diesel engines is extensive and some examples have been already referenced in the introduction. Physical models rely on first principles but uses complex structures that require big computing times. In addition, these are not free from drift problem. On the other hand, control-oriented models are often data driven. Here, different possibilities can be found. For example, heuristic approaches can derive maps from engine data (Lughofer et al., 2011), building a model that is easy to integrate into the engine ECU, but lacking of extrapolation capabilities. This needs some filtering for catching dynamics and if implemented online, drift can be corrected with observers (Payri et al., 2012) or feedback control. Black box models rely on system identification (Karlsson, Ekholm, Strandh, Tunestål, & Johansson, 2010) but are often operating point dependent and its adaptation is not an easy task. As an intermediate solution, gray box structures can use fundamental relationships with experimental fittings (Hirsch, Alberer, & del Re, 2008). In the following a gray box NOx model is provided for comparison with the online methods presented hereinafter.

3.1. NOx model used as benchmark

Thermal NOx formation (Lavoie, Heywood, & Keck, 1970) is the dominant mechanism in diesel, benefited from peak temperatures in the cylinder and lean conditions (excess of air). These conditions maximize engine efficiency and reduce soot emissions: the well-known soot-NOx trade-off, see e.g. Neef, Makkee, and Moulijn (1996). Three in-cylinder variables should be necessary for NOx reconstruction: peak temperature, oxygen concentration and residence time, but these can not be directly measured. In-cylinder pressure may be used as input but signal treatment is still an issue (Guardiola, López, Martín, & García-Sarmiento, 2011). For a fast mean value NOx model, oxygen rate in the cylinder and engine operating conditions (usually speed n and injected fuel mass mf) seem to be suitable variables for characterizing NOx; see e.g. Ericson, Westerberg, Andersson, and Egnell (2006), Galindo, Luján, Climent, and Guardiola (2007).

The model is point relative and the main equation is

\[ x_m = NOx_{0}(m_f, n)e^{-k(m_f-n_{rev}F_r-F_{rel}P_m)C(\text{comb}, H, T_w, \ldots)} \]

where subindex 0 indicates nominal conditions defined by manufacturer calibration. NOx_{0} is NOx emissions as function of nominal operating point conditions (n and m_f), and exponential is built around nominal values of inert gas r_{EGR}F_r, which expresses the real quantity of burned gas fraction, being r_{EGR} the EGR ratio and F_r the relative fuel-to-air ratio. EGR flow is solved by a mean value engine model (MVEM) and F_r is observed from m_f, m_a and oxygen output from ZrO2 sensor at the exhaust: see Appendix A for more details on the sub-model equations. F_{rel}P_m, F_r, which define the nominal values for r_{EGR} and F_r, respectively, and k are mapped with a proper grid using n and m_f as inputs. The set-point relative structure is suitable for minimizing errors around nominal, especially when real time actions are pursued and allows defining optimal control strategies. Effects of combustion modes, humidity H, engine temperature T_w are considered with the correction factor C, i.e. C=1 when no correction. T_w and H are directly measured on engine.

The model cannot be used for stand-alone simulation because it not only requires ECU signals but also is suitable for control and diagnosis and for offline NOx diagnosis after tests, i.e. if no NOx sensor is available. Nevertheless, model can be completed with a full air-path MVEM including turbocharger and combustion

blocks; a good example of an air-path model can be found in Wahlström and Eriksson (2011).

For comparing with sensor signal, a first order discrete filter \( G(z) \) with a time delay \( \tau \in \mathbb{Z}^+ \) is applied to model output

\[ y_m = \frac{1-a}{1-ae^{-\tau}}x_m \]  

(2)

obtaining \( y_m \) where SOI steps are performed for characterizing \( a \) and \( \tau \). Sensor response time \( a \) can be modeled as constant but delay \( \tau \) has a non-linear behavior difficult to model. A detailed study on NOx sensor characterization is made in Galindo et al. (2011). In this work, \( \tau \) is constant as being sufficient for obtaining good results with the SDMP and NEDC for the model and updating methods. Anyway, a robust implementation should include \( \tau \) variability; the interested reader can go to Trimboli, Di Cairano, Bemporad, and Kolmanovsky (2012) who model delay in \( F_r \) signal from NOx sensor as function of air mass flow, engine operating conditions and speed.

Model is calibrated using steady-state measurements and dynamic tests comparing \( \sum_{i=1}^{n_p}(\text{y}_{\text{NOx}} - y_{\text{m}})^2 \) by least squares fitting where \( n_p \) is the number of points included. Model validation has been made with a set of engine cycles, including NEDC and SDMP. Fig. 3 shows results in the SDMP test comparing \( y_{\text{NOx}} \) and \( y_{\text{m}} \). The fitting is good and the model is capable of estimating \( y_{\text{NOx}} \) with a minimum error. Fig. 18 presented in Section 6.4 shows results of a non-optimized calibration for the NEDC and even though some drift appears, dynamic behavior is cached.

3.2. Motivation for updating look-up tables

Two problems can be underlined when working with control-oriented models. On one hand, the model accuracy is driven by the collection of the appropriate data and calibration of all the parameters. This is a hard and time consuming task. In fact, ECU has a big number of maps and parameters for engine and vehicle management. On the other hand, independently of how well the model has been calibrated there is inevitably a drift between the model and the surrounding conditions changes and the engine ages. Data based models are highly sensitive to the calibration data set and will have problems with aging, manufacturing discrepancies, slowly varying parameters and other non-modeled variables. To show this, an old calibration set for the model (for maps and sensor dynamics) is used for simulating the SDMP test. Results can be seen in Fig. 4, where dynamics are well cached but a clear drift exists. Here is not only aging that affects, but test or ambient conditions.

These problems are similar for other model structures and therefore learning algorithms for look-up tables can be used for calibration and/or online adaptation. In the next, the paper focuses on the online adaptation of look-up tables and develops a

![Fig. 3. Model simulation y_m and sensor signal y_NOx in the SDMP test. In spite of the hypothesis taken, the results are quite good and capable of tracking NOx emissions.](image-url)
computationally efficient method for correcting drift, aging and indeed allowing self-calibration.

4. Updating look-up tables

In the literature different approaches for online learning of look-up tables can be found. Peyton Jones and Muske (2009) use a recursive least-squares method with a forgetting factor, Wu (2006) distributes proportionally the existing error on the estimation between the active parameters and Karlsson, Ekholm, Strandh, Johansson, and Tuenastal (2008) use the subspace method for identifying NOx and soot emission models in heavy-duty diesel engines. However KF based approaches (see a complete book about KF methods in Simon, 2006) provide a systematic way for updating map parameters and it also appealing for engine applications as it can also handle aging (see Höckerdal et al., 2011 for a discussion) both for parameters and estimation, i.e. parameter drifts with respect to those have been excited for a longer time are expected to have bigger drift with respect to those have been excited recently. KF manages this issue optimally in the sense of minimizing expected estimation error.

Nevertheless, when observing large maps there are high memory requirements and computational burden. The core of the problem is that a KF implementation for updating a look-up table with 20-by-20 elements gives rise to a covariance matrix of 400-by-400 elements and thereby also a significant computational burden when solving the Riccati equations. Therefore an important discussion of the paper is how the KF can be used or modified to get an efficient updating procedure for look-up tables without an important loss of properties.

4.1 Kalman filter basics

The main equations in the KF Kalman (1960) are recalled here for presenting the nomenclature used after although the reader familiarized with control engineering could skip this subsection. In the setting the data is assumed to be generated by the following discrete time system

\[ x_k = f(x_{k-1}, u_k) + w_k \]  
\[ y_k = h(x_k, u_k) + v_k \]

where \( x_k \in \mathbb{R}^{n_x} \) represents the state vector, \( u_k \in \mathbb{R}^{n_u} \) the input vector, \( y_k \in \mathbb{R}^{n_y} \) the output vector. If \( f \) and/or \( h \) are non-linear a previous linearization step is required for the filter and then, elements \( i \) of \( F_k \) and \( H_k \) are obtained

\[ F_{k,i} = \frac{\partial f}{\partial x_k} \bigg|_{x_k = x_i} \quad H_{k,i} = \frac{\partial h}{\partial x_k} \bigg|_{x_k = x_i} \]

being \( F \) the linearized process matrix and \( H \) the linearized output matrix. From now, discussion is valid both for linear and non-linear systems and KF will be used referring to Kalman filter based methods, including non-linear Extended version and standard one.

Noises \( w_k \in \mathbb{R}^{n_w} \) and \( v_k \in \mathbb{R}^{n_v} \) are assumed to be independent and both generated by Gaussian distribution with zero mean and covariance matrices \( Q_k \) resp. \( R_k \) defined by

\[ E[w_k w_k^T] = Q_k \]  
\[ E[v_k v_k^T] = R_k \]

In applications these are often chosen to be constant, i.e. \( Q \) and \( R \), and diagonals.

Then, \( \hat{x}_k \in \mathbb{R}^{n_x} \) is the observation of the state vector \( x_k \)

\[ \hat{x}_{k|k-1} = f(\hat{x}_{k-1}, u_k) \]  
\[ e_k = y_k - h(\hat{x}_{k|k-1}, u_k) \]  
\[ \hat{x}_k = \hat{x}_{k|k-1} + K_k e_k \]

where \( K_k \) Kalman gain is solved by the following iterative equation

\[ P_{k|k-1} = (F_k P_{k-1} F_k^T + Q) \]  
\[ K_k = P_{k|k-1} H_k^T (H_k P_{k|k-1} H_k^T + R)^{-1} \]  
\[ P_k = (\mathbf{I} - K_k H_k) P_{k|k-1} \]

where matrix \( P_k \) is the covariance matrix of the state estimate error (Ljung, 1999)

\[ P_k = E[(x_k - \hat{x}_k)(x_k - \hat{x}_k)^T] \]

4.2. Look-up tables

A look-up table \( T \in \mathbb{R}^{n \times n_y} \) is defined as a N-Dimensional mapping \( T: \mathbb{R}^n \rightarrow \mathbb{R} \) defined by a grid in \( N \in \mathbb{Z}^+ \) dimensions, where each one has \( n_i \) grid points. The mapping further relies on a multivariate interpolation \( q(\cdot) \) to calculate the function value from the input using the grid for the interpolation variables and \( T \). In automotive systems, the multivariate interpolation schemes are often linear in the dimensions, and this is the case that will be considered here. Without loss of generality the presentation will use 2D tables as being the most frequently occurring dimensions, since working with other dimensions is only a matter of reducing or increasing indexes. Then, for \( N=2 \)

\[ T = [t_{ij}] \]

where \( i = 1, \ldots, n_i \) and \( j = 1, \ldots, n_j \) \( (r \text{ stands for row and } c \text{ for column}) \). The multivariate interpolation function for generating the output \( y_k \) from input \( u_k = [t_{1i}(k) t_{2j}(k)]^T \) can be expressed as

\[ y_k = \text{vec}(q(u_k))^T \text{ vec}(T) \]

where \( \text{vec}(\cdot) \) is the vectorization transformation and \( q(u_k) \) is the interpolation matrix that both selects the elements to be interpolated and contains the weights. In the 2D case \( q(u_k) \) selects the 4 \( \{2^k\} \) in the ND case) active elements \( t_{ij} \) \( t_{i,j+1} \) \( t_{i+1,j} \) \( t_{i+1,j+1} \) \( (\text{with } i,j \text{ fulfilling } u_{i,k} \in [r_i, r_{i+1}] \text{ and } u_{2,k} \in (c_j, c_{j+1}] \text{ and thus contains the following block with non-zero weights} \)

\[
\begin{bmatrix}
q(u_k)_{i,j} & q(u_k)_{i,j+1} & q(u_k)_{i+1,j} & q(u_k)_{i+1,j+1}
\end{bmatrix} =
\begin{bmatrix}
(1-\eta_{1,k})(1-\eta_{2,k}) & (1-\eta_{1,k})\eta_{2,k} & \eta_{1,k}(1-\eta_{2,k})
\eta_{1,k}\eta_{2,k}
\end{bmatrix}
\]

where

\[
\eta_{1,k} = \frac{u_{1,k} - r_i}{r_{i+1} - r_i}; \quad \eta_{2,k} = \frac{u_{2,k} - c_j}{c_{j+1} - c_j}
\]
4.3. Modeling for learning, drift or aging in tables

In the setting, $T$ models a nonlinear function and the interesting aspect is to allow the model to adapt to the system to either learn the system and/or follow the aging of the system. This is modeled in the standard way as a random walk process, where the table parameters are collected in a state vector $x_k \in \mathbb{R}^{n_r \times n_c}$. The full model $u_k \rightarrow v_k$ is

$$x_k = x_{k-1} + w_k$$

(11a)

$$y_k = \text{vec}(q(u_k))^T x_k + v_k$$

(11b)

Note that no uncertainties are allowed in the interpolation variable $u_k$.

For convenience, the non-zero elements in vec($q(u_k)$) are denoted $q_k^o \in \mathbb{R}^{1 \times d}$ ($o$ stands for observable) and the corresponding elements in the state vector, $x_k^o \in (\mathbb{R}^n)^d$. Then, following expression is obtained for the output:

$$y_k = q_k^o x_k^o + v_k$$

(12a)

$$q_k^o = [(1-\eta_1 k)(1-\eta_2 k)] \eta_1 k(1-\eta_2 k)$$

(12b)

where $Q$ and $R$ are

$$Q = \sigma^2_0 I_{n_r \times n_r}$$

(13a)

$$R = \sigma^2_w$$

(13b)

and $I_{n_r \times n_r}$ is the identity square matrix. If there is engineering application knowledge available, $\sigma^2_0$ might be selected independently for each table element building a vector $\Sigma_k^w = [\sigma^2_1; \ldots; \sigma^2_{n_r \times n_c}; \ldots; \sigma^2]$. This contains individual variances in such way that the look-up table estimation has structural properties that can be exploited to reduce the computational and memory requirements significantly in a simple way.

4.4. Kalman filter based method for updating look-up tables, KF

The EKF can now be used to observe $x_k$ when measurements of $y_k$ are given. At every iteration, $K_k$ is calculated based on (7), where $F = I_{n_r \times n_r}$ is constant and $H_k = \text{vec}(q(u_k))^T$. Although only the active elements $x_k^o$ are updated at every $k$, all $P_k$ elements enter in the equation, which leads to huge calculations and big required memory resources. This makes difficult the implementation in commercial ECUs.

There are several publications on computational aspects of Kalman filters, for example some have studied the filter optimization when different nonlinear functions are handled, e.g. Charalampidis and Papavassiliopoulos (2011), Jørgensen, Homsen, and Kristensen (2007) and Singer and Sea (1971); the latter make an interesting study on the total number of operations required for the updating phase. Chandrasekar, Kim, and Bernstein (2007) present an interesting methodology when system order is extremely large by using the finite-horizon optimization technique for obtaining reduced-order systems. But key observation here is that the look-up table estimation has structural properties that can be exploited to reduce the computational and memory requirements significantly in a simple way.

To understand the problem, covariance matrix in the KF $P$ is studied. For instance, a 2D look-up table with a size $n_r \times n_c$ needs system $F$ and covariance matrices $P$ of $(n_r, n_c) \times (n_r, n_c)$. However, only the active elements are influenced during the update and by defining a local observable system (which corresponds to the active elements), one receives a system which is $4 \times 4$ against $(n_r, n_c) \times (n_r, n_c)$. That means that non-active elements do not affect the updating until they become active (zero values for the related elements). Furthermore, $P_k$ is a symmetrical and positive-semidefinite matrix, which allows further simplifications in the KF calculations.

4.5. Local observable system and steady-state approach, SSKF

The Local Observable System is defined and analyzed and a computationally efficient approximation for the KF, from Guardiola et al. (2013), is described. Following with the 2D look-up table application, at every iteration a maximum number of 4 elements can be updated and then, if no dynamics are accounted, the general $n_r \times n_c$ system (11) can be written as $4 \times 4$

$$x_k = k_0 x_{k-1} + w_k$$

(14a)

$$y_k = q_k^o x_k^o + v_k$$

(14b)

The simplest 2D map is a table of only 4 elements, and where the one existing area is always active, i.e. system (14) is exactly (11). For the general case, (14) must be rewritten at every $k$ as elements and matrices change. Supposing that system (3) is linear time-invariant (LTI): $x_{k+1}^o$, $x_k^o$, $F$ and $H$ are constant, then KF is steady-state (Simon, 2001). Of course system (14) is not time invariant, because $q_k^o$ (and then $H$) depends on $u_{t+k}$ and $u_{t+2k}$, which are the inputs. But, if inputs are considered stationary during a certain time, sequence $\{K\}_{i=1}^\infty$ converges to a steady-state gain $K^s$. For the 2D case

$$K^s = \begin{bmatrix} k_1(\eta_1, \sigma^2_1 / \sigma^2_w) & -k_2(\eta_2, \sigma^2_2 / \sigma^2_w) \\ k_1(\eta_2, \sigma^2_1 / \sigma^2_w) & k_2(\eta_1, \sigma^2_2 / \sigma^2_w) \\ k_1(1-\eta_1, \sigma^2_1 / \sigma^2_w) & k_2(1-\eta_2, \sigma^2_2 / \sigma^2_w) \\ k_1(1-\eta_2, \sigma^2_1 / \sigma^2_w) & k_2(1-\eta_1, \sigma^2_2 / \sigma^2_w) \end{bmatrix}$$

(15)

where $k_i$ is computed as follows:

$$k_i(\eta, \sigma^2_i / \sigma^2_w) = \frac{0.5(1-\eta)(1+s)}{0.5(1+s)(1-2\eta+2\eta^2)+\sigma^2_i / \sigma^2_w}$$

(16a)

$$s = \sqrt{1 + \frac{4}{(1-2\eta+2\eta^2)\sigma^2_w}}$$

(16b)

This approach allows mapping $K$ and this only depends on inputs ($\eta_1$ and $\eta_2$ calculated in (10)) and noise trade-off $\sigma^2_1 / \sigma^2_w$. Nevertheless, as parameter aging is not considered, this makes the algorithm quite fast and light but robustness must be assessed in the applications. Algorithm pseudo-code is in Appendix B.

4.6. About system observability

The system (11) is not fully observable in one iteration. But local observability can be ensured if parameters or states in the local system (14) converge with a given data set. As system is not LTI, the ordinary observability rank condition (Ogata, 2001) is not directly applicable. $q_k^o$ depends on the input data, and it is evident that if a enough level of excitation is given, then the system could be observed, whatever the method chosen. This minimum level of excitation could be proved if 4 elements have been excited. A sufficient observability matrix for local observability may be built with the first 4 independent observations not necessarily consecutive in instants $i_1, i_2, i_3$ and $i_4$ of the elements of the involved area:

$$O = \begin{bmatrix} q_{i_1}^o \\ q_{i_2}^o \\ q_{i_3}^o \\ q_{i_4}^o \end{bmatrix}$$

(17)

and if the rank of this matrix is 4, then system (14) could be observable. But here the problem is linked with the noise tuning and
indeed full rank $\mathbf{O}$ does not lead to system full convergence (full observability does not lead to full convergence as method and model structure affect). Majority of elements are also included in other neighbor areas and because of this the minimum condition of observability of a given state depends on the number of independent measurements that affect this state, and as other elements affect this, the condition stated in Höckerdal et al. (2011) gives a general condition.

4.7. The simplified Kalman filter SKF

Despite that only the active elements affect $K_k$, all $P_k$ elements are predicted and updated at every $k$. $P_k$ can be reorganized in such way that variances related with observable $P_k^o$ and unobservable $P_k^u$ elements are split.

$$
P_k = \begin{bmatrix}
P_k^o & \cdots & \cdots & \cdots \\
\cdots & P_k^o & \cdots & \cdots \\
\cdots & \cdots & \cdots & \cdots \\
\cdots & \cdots & \cdots & P_k^o
\end{bmatrix}
$$

(18)

$K_k$, $q_k$ and $Q$ can also be split in the same way. Iterating one step ahead (7c) is written

$$
P_{k+1} = \begin{bmatrix}
1 & 0 \\
0 & 1
\end{bmatrix} \begin{bmatrix}
P_k^o + Q^o \\
(P_k^o + Q^o)^{-1}
\end{bmatrix}
$$

(19)

where all observable matrices are $4 \times 4$, while the unobservable ones are $(n, n, -4) \times (n, n, -4)$. (19) shows how only diagonal elements of $P_k^o$ are affected by adding the diagonal matrix $Q^o$. However, both diagonal and non-diagonal elements of $P_k^u$ are affected as in a $4 \times 4$ KF. Crossed relationships between observable and unobservable system, showed as dots in the matrix, are kept constant. This allows simplifying the complete system to the $4 \times 4$ active state-space system, whose resolution is highly computationally efficient.

Current work develops an approximation of the KF that requires both less memory and computations in the iterations and is named SKF. SKF builds and solves the local observable system (14) at every $k$. Some simplifications are assumed: for the non-active elements, whose covariance matrix is $P^o$, non-diagonal elements are neglected building a vector $P^o = \text{diag}(P_k^o)$. This is justified for two reasons. First, when the system is running, then leaves one area and later returns to it, related variances reflect the time that the system has been out of this area. This is a desired property since the aging is captured by the variance increase due to $Q^k$ which indeed here is defined as a diagonal matrix. Older crossed correlations are maintained over the time and whether or not they are valid is not sure. Second, simulation results show that the full EKF and the SKF have similar performances when running on synthetic data and real data from the engine application. Then, at every iteration, the local system (14) is solved with the only additional operation of $P_k^o + Q^o$ (converted to a vectorial calculation). SKF is a suboptimal filter but with a similar behavior than KF, as will be demonstrated in Sections 5 and Appendix A.

Pseudo-code for the SKF implementation is given in Algorithm 1. Note that code is to show the algorithm where some variables are not strictly needed in the computer programming, e.g. $P_k$ always that $P_k$ exists, but are kept for the sake of clarity. Here, noise variance $\sigma_w^2$ is the same for all elements, allowing to pre-define diagonal matrix $Q^o = \sigma_w^2 I_{4 \times 4}$ for active system and vector $Q^u = \sigma_v^2 \text{diag}(I_{n \times n} - 4)$ for the non-active one.

**Algorithm 1.** Pseudo-code for the simplified Kalman filter SKF.

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><strong>Input</strong>: $\hat{x}<em>{k-1}$, $u_k$, $P</em>{k-1}$, $R$, $Q^o$, $Q^u$</td>
</tr>
<tr>
<td>2</td>
<td><strong>Output</strong>: $\hat{x}_k$, $P_k$</td>
</tr>
<tr>
<td>3</td>
<td><strong>While</strong> Algorithm is running <strong>do</strong></td>
</tr>
<tr>
<td>4</td>
<td><strong>if</strong> Active area changes <strong>then</strong></td>
</tr>
<tr>
<td>5</td>
<td>$x_{k-1}^o$ is stored in the correct positions of $x_{k-1}$</td>
</tr>
<tr>
<td>6</td>
<td>diag($P_k^o$) and $P_{k-1}^u$ are stored in a global variance vector $P_{k-1}$ according to their correct positions</td>
</tr>
<tr>
<td>7</td>
<td>Redefinition of $P_{k-1}^o$, $P_{k-1}^u$, $x_{k-1}^o$</td>
</tr>
<tr>
<td>8</td>
<td><strong>end</strong></td>
</tr>
<tr>
<td>9</td>
<td>Computation of $\eta_{1,k}$ and $\eta_{2,k}$ as in (10)</td>
</tr>
<tr>
<td>10</td>
<td>Definition of $q_k^o(\eta_{1,k}, \eta_{2,k})$ as in (12b)</td>
</tr>
<tr>
<td>11</td>
<td>$P_k^{o_{k,k-1}} = P_k^o + Q^o$</td>
</tr>
<tr>
<td>12</td>
<td>$K_k = P_k^{o_{k,k-1}}Q_k^T(q_k^{o_{k,k-1}}P_k^{o_{k,k-1}}Q_k^T + R)^{-1}$</td>
</tr>
<tr>
<td>13</td>
<td>$x_k^o = x_{k-1}^o + K_k(y_k - q_k^o x_{k-1}^o)$</td>
</tr>
<tr>
<td>14</td>
<td>$P_k^o = (I_4 - K_k q_k^o) (P_k^o + Q^o)$</td>
</tr>
<tr>
<td>15</td>
<td>$P_k^u = P_{k-1}^u + Q^u$</td>
</tr>
</tbody>
</table>

where all observable matrices are $4 \times 4$, while the unobservable ones are $(n,n,-4) \times (n,n,-4)$. (19) shows how only diagonal elements of $P_k^o$ are affected by adding the diagonal matrix $Q^o$. However, both diagonal and non-diagonal elements of $P_k^u$ are affected as in a $4 \times 4$ KF. Crossed relationships between observable and unobservable system, showed as dots in the matrix, are kept constant. This allows simplifying the complete system to the $4 \times 4$ active state-space system, whose resolution is highly computationally efficient.

In the following section the memory allocation and computational complexity of the three methods, full EKF (here denoted KF), the SKF developed here and described about, as well as the SSKF is described.
4.8. Analysis of memory and computations

A computation time study is made programming code in Matlab software on a laptop computer Intel Core 2DUO T9300 2.5 GHz with Windows VISTA 64 bits. Fig. 5 shows the relative computation time used by the three methods for updating 2D look-up tables of different sizes. Only square \( n \times n \) tables have been considered with \( n \) ranging from 2 to 21, since it is the total number of elements in \( \text{vec}(T) \) that influences memory allocation and computations. The Y-axis shows the relative computation time with respect to the average time that the SSKF takes for performing 1000 iterations, i.e., a relative time of 1 is that method needs exactly the same time that SSKF. SKF needs around 1.15 times the SSKF calculation time, whatever the number of parameters. This slight difference is explained by the manipulation of variances and the need of redefining active vector and matrices \( P^k \) and \( x^k \) when active area changes (see lines 4 and 5 of Algorithm 1 and compare with Algorithm 2). However, required time for KF rapidly grows when number of parameters increases, i.e. for 256 parameters with Algorithm 2). However, required time for KF rapidly grows when number of parameters increases, i.e. for 256 parameters (16 x 16), KF needs around 7.8 times the one for SSKF. An exponential function is fitted by least squares method to the KF time as function of number of parameters \( np \), whose result is the relative time KF with respect to the one of SSKF.

Table 2 gives some numbers when comparing required memory resources and computation times for the algorithms. SKF permits reducing the requirements in terms of memory resources and the general system \( (n_l,n_h) \times (n_l,n_h) \) is reduced to a simple \( 4 \times 4 \). These results show the benefits in memory and computational when using the SSKF and SKF compared to the full KF.

5. Simulation of the updating algorithms

With these promising results the convergence and robustness of approximations need also to be studied and this is first performed using simulations. The objective of this section is comparing the abilities and performance of the algorithms against different synthetic cases where no dynamics are accounted (real system and model are static); later, Section 6 will prove the algorithms under real conditions in a diesel engine considering dynamics (sensor output will be used there as reference) and completing the study on methods capabilities.

For automotive applications, engine speed \( n \) and fuel mass injection quantity \( m_f \) usually define the engine operating point and the look-up table scheduling points selected in the simulation are inspired by these quantities. Hence \( r \) represents a grid for \( u_1(k) \) (speed in rpm) and \( c \) for \( u_2(k) \) (injected mass fuel in mg/str):

\[
\begin{array}{c}
r = \{500 \ 1500 \ 2500 \ 3500\} \\
c = [0 \ 20 \ 40 \ 60]
\end{array}
\]

and the true map \( T_r \) is defined with the surface

\[
T_r = \begin{bmatrix}
1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1
\end{bmatrix}
\]

where \( x_r = \text{vec}(T_r) \) is the expanded state vector objective of the estimation process. No initial process knowledge is considered: \( x_{0|0}, x_{0|1}, P_0 \) are initialized with zeros. \( \sigma^2 \) = 1 for all cases, unless it should be pointed.

System measurements are given and algorithms performance is evaluated. As synthetic signals are used, this allows defining error metrics. Simulations include two worst-cases and a favorable one:

- Input with random variation: a random shot of values following an uniform probability distribution. This is an ideal situation for learning as all areas are excited and parameters aging is low (excitation is high). Measurements are perfect and constant \( y_k = 1 \) \( \forall k \).
- Linear variation: varying \( u_2 \) keeping constant \( u_1 \). This variation tests a degenerated case where observability is critical along the \( u_1 \)-dimension. Measurements are perfect and constant \( y_k = 1 \) \( \forall k \).
- Measurement noise rejection: studying the effects of noise transmission between output and observation when measurement vector is noisy.

5.1. Simulation 1: input with random variation

Sequence \( \{u_1\}_{k=1}^{1400} \) following an uniform distribution over grid defined in (20) is used for exciting the system. A complete identification is possible as grid is completely covered. This situation is not so far of the reality for diesel engines, e.g. urban cycles with a lot of speeding/braking actions cause quasi-random variations in partial-low load areas of the engine map (see Section 6 for seeing how dynamics are considered). The selected grid and covered points are shown in Fig. 6.

Here, variance information is not as relevant as in other cases, because of the stochastic nature of inputs. The mean value of all states is plotted for at each time step in Fig. 7. All three methods perform well and rate of convergence can be tuned varying \( \sigma^2_{11} \) and \( \sigma^2_{22} \).

A quick view on the effect of the hypothesis for the SKF is shown for one element \( x_6 \) in top plot of Fig. 8, which shows variance \( P_{66} \) of \( x_6 \) for KF and SKF (not applicable for SSKF). This value indicates the observability of \( x_6 \): when it increases monotonically, \( x_6 \) is not observable. In some parts, an offset between \( P_{66} \) for KF and SKF appears because of the neglected covariances in SKF, but in the end, this offset is absorbed and the two methods behave similar. The bottom plot of Fig. 8 shows \( x_6 \) evolution for the

**Table 2** Memory storage and computational burden comparison between Kalman Filter KF, Simplified Kalman Filter SKF and steady-state approach SSKF.

<table>
<thead>
<tr>
<th></th>
<th>P</th>
<th>K</th>
<th>Rel. Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>KF</td>
<td>((n_l,n_h) \times (n_l,n_h))</td>
<td>((n_l,n_h) \times 1)</td>
<td>0.82 exp(0.0087(n_l))</td>
</tr>
<tr>
<td>SKF</td>
<td>(4 \times 4 )</td>
<td>(4 \times 1)</td>
<td>1.15</td>
</tr>
<tr>
<td>SSKF</td>
<td>(4 \times 4)</td>
<td>4x1</td>
<td>1</td>
</tr>
</tbody>
</table>

Fig. 5. Relative computation time required for the methods for updating 2D look-up tables. Times are normalized with the average time that SSKF needs for computing 1000 iterations of the filter.
three methods. $x_6$ is updated when the element becomes observable. Note that $x_6$ is already active in the first iteration and $K_1$, which represents the converged value of the equivalent LTI system, is non-null. Then, SSKF updates $x_6$ in $k=1$ while $K_1$ for KF and SKF is null. Anyway, this is not an advantage of this method: KF and SKF can behave similar if $P_0$ is non-null, but here for the simulations $P_0 = [0]$. When there exists no previous knowledge of the system to be learnt, it should advisable to initialize $P_0$ with a certain value for speeding up updating during first iterations.

5.2. Simulation 2: input with linear variation

$T_e$ is identified with a sequence $(u_{t})_{k=1}^{200}$ where $u_1$ varies monotonically from 0 to 60 (from $k=0$ to 100) and coming back from 60 to 0 (from $k=101$ to 200), while $u_{200} = 200 \forall k$. This condition is highly restrictive as the excitation level is not high: at each $k$ only small variations of input $u_2$ are applied and due to this full convergence condition is not fulfilled in the first running of one area, and interactions between areas and the way back are necessary to ensure the convergence of both KF and SKF. Three matrix areas are excited (see (22)); area 1: elements 5,6,9,10; area 2: 6,7,10,11; area 3: 7,8,11,12. Moreover, due to inputs nature, row 5–8 evolution is equivalent to 9–12.

$$T_k = \begin{bmatrix} x_{5,k} & x_{6,k} & x_{7,k} & x_{8,k} \\ x_{9,k} & x_{10,k} & x_{11,k} & x_{12,k} \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix} \quad (22)$$

Fig. 9 shows the evolution of 4 states $x_5$ to $x_8$. First thing to note is the similar performance of KF and SKF and the instabilities of SSKF. For instance, $x_5$ is in the area 1, and during the first run it is not capable of converging, but in one round KF and SKF methods converge, and SSKF seems to do it, at least in $k=200$. The other states have slightly different performances, because the first area has already been covered: e.g. $x_6$ is closer to the convergence area being a member of area 1 and area 2, but until the way round it does not get the full convergence for KF and SKF methods. Similar behaviors are observed for $x_7$ with these methods. In addition, $x_8$, as member of the area 3, has the advantage that areas 1 and 2 have been covered first, and the information is already available for getting the full convergence for KF and SKF in $k=100$. The full observability (in the sense of convergence) of these 8 elements is reached only when $\mathbf{O}$ in (17) accounts for the three areas when using KF and SKF. Nevertheless, SSKF estimation does not converge for $x_5$, $x_7$ and $x_8$ due to no state error information is tracked, and the system and convergence becomes more dependent on data.

![Fig. 6](image6.png)

**Fig. 6.** Random variation of inputs $u_1$ and $u_2$ showed with dots and table grid with the dashed line.

![Fig. 7](image7.png)

**Fig. 7.** Average value of table parameters for the three considered updating methods when considered a uniform distributed inputs sequence. The correct value is 1.

![Fig. 8](image8.png)

**Fig. 8.** Variance analysis for an uniform distributed inputs sequence. Top: variance evolution of 6th element $P_6$ where black thick line is KF and gray line is SKF ($P_0$ is normalized by 100). Bottom: $x_6$ value where black thick line is KF, gray line is SKF, thin black is SSKF.

![Fig. 9](image9.png)

**Fig. 9.** States evolution for a monotonically varied inputs sequence. Black thick line is KF, gray line is SKF, thin black is SSKF and dashed line shows when the element is observable (1 is active; 0 is not active).
In order to understand how KF and SKF behave is informative to study also the variance and covariance evolution (remember that this is not present for the steady state filter, SSKF). Fig. 10 shows the variance evolution for row elements, while Fig. 11 shows the evolution of a few covariances between states (corresponding to non-diagonal elements of $P$). As for the random case in 8, variance evolution for KF and SKF is similar, although some deviation exists in elements 6 and 7. This is explained for covariances or non diagonal $P_{ij}$ elements that SKF only accounted for active areas. This is shown in Fig. 11 where some crossed covariances are plotted. When the pair of elements are not active, covariances of these are not tracked, equivalent to reset them to zero in the global $P$. This is a particular difference with KF, which always tracks covariance, although area would not be active, increasing computational requirements without a justified improvement in the estimation. However, when an area is active, the SKF covariance evolves as the KF and in the limit is equivalent to the KF. The difference is larger initially but after the observations have converged the difference is negligible, as shown in Fig. 9. This pattern may be slightly modified linking the covariance values to elements pairs and not to the involved areas (excepting boundaries, all pair of elements are shared in two different areas), although final results are similar.

5.3. Simulation 3: measurement noise rejection

A uniform distributed noise with zero mean and maximum amplitude of 0.2 is applied to the measurement given in simulation 2; $y_k$ is shown in Fig. 12. A robust method must filter this noise and converges to the true values.

The resulting estimations are shown in Fig. 13, and the evolutions are quite similar to the ones of Fig. 9 although with a slight noise transmission. KF and SKF performs well, being able to also filter the noise, but SSKF is not capable of converging again having a similar response as in simulation 2. Variance and covariances are exactly the same of Figs. 10 and 11 as state-space system and input sequence does not change.

5.4. Conclusions from simulations

SKF method is demonstrated to have similar accuracy as KF, at least for the numerical cases proved in this section. Furthermore, KF and SKF solutions converge for all cases, including the simulations 2 and 3, which are specifically restrictive because of the low
level of excitation. KF, as shown in Fig. 5 and Table 2, is computationally heavy. SSKF is the fastest and lightest method as variance is not tracked and $K^SS$ formulation is derived analytically. SKF only needs around 1.15 times the SSKF time for calculation, and the memory resources required are similar, except for the track of variances. SSSK behaves well in situations where variance tracking is not critical, e.g. simulation 1, but is less robust when data is structured and the level of excitation is low, e.g. simulations 2 and 3. KF and SKF are also capable of filtering the noise, an advantage for a correct updating, and SSSK also filters the noise, but its stability depends on data. In comparison, the SKF method, derived from the KF, is demonstrated to be an efficient and accurate method for updating look-up tables, at least for the simulation conditions presented. In the next, methods are tested in a real application.

6. Online NOx estimation

The attention is turned to the problem of online NOx estimation. The approach consists of modeling NOx by using an adaptive model based on a static 2D look-up table whose parameters are estimated with the updating methods. The state-space model (11) is slightly modified for coping with sensor dynamics and inputs $u$ are treated as well.

Model parameters are estimated online while the engine is running, or re-calibrated if system ages once that an initial calibration is given without any special calibration procedure or test rig, beyond the sensor measurements. The first time that the engine is running, parameters evolve, and when the engine switches off, the stored parameters can be used for predicting NOx. When the engine is running again, the parameters keep evolving for correcting drift and slowly varying effects. Furthermore, the observer built for updating can be used for having an actual NOx estimation, i.e. avoiding filtering and delay of sensor. SKF is appropriate for online usage because of the light computational burden and the estimation capabilities. First, dynamic model for learning is presented.

6.1. Dynamic equations for learning

A NOx table $T_{NOx}$ using $m_f$ and $n$ as inputs is calibrated with the SDMP and NEDC. Off the shelf ECU maps are usually 1D or 2D and maximum sizes are around and slightly above 256 parameters (16 × 16). In the case study here, $T_{NOx}$ is 12 × 18 (216 parameters) and the first dimension accounts for engine speed $n$ and the second for the injected fuel mass, $m_f$. The second dimension has more density as NOx is more sensitive to load variations. As far as speed sensor and injection signal from ECU are fast and responses are expected fast, no dynamical treatment is made, while delays are still needed to phase the system correctly with the NOx sensor

\begin{align}
&u_{1,k} = z^{-1} n(k) \quad (23a) \\
&u_{2,k} = z^{-1} m_f(k) \quad (23b)
\end{align}

where $z$ represents the average sensor input delay obtained with the SOI procedure. This is preferred to apply the delay in the state-space model for avoiding to increase the dimension of the system.

In this basis, estimated NOx can be denoted as $\hat{X}_M$ (M for the used method) and modeled using the adaptive map $T_{NOx, \text{est}}$

\begin{equation}
\hat{x}_M(k-\tau|\tau) = T_{NOx, \text{est}}(u_{1,k}, u_{2,k}) \tag{24}
\end{equation}

defining a quasi-static representation of NOx output where $\tau$ defines the time that table has been updated.

Sensor dynamics are considered as in (2)

\begin{equation}
y_M(k) = \frac{1-\alpha}{1-\alpha z^{-1}} X_M(k-\tau) \tag{25}
\end{equation}

and the global system (11) must be augmented with one extra-dimension (if a first order filter is considered for sensor dynamics)

\begin{equation}
x_M^\omega = \left[ \begin{array}{c}
I_{m_f} \\
(1-\alpha q_k)^{-1} a
\end{array} \right] x_M^{\omega,k} + w_k \tag{26a}
\end{equation}

\begin{equation}
y_M^\omega = H^\omega x_M^\omega + v_k \tag{26b}
\end{equation}

with

\begin{equation}
x_M^\omega = \left[ \begin{array}{c}
\hat{x}_M \\
\hat{y}_M
\end{array} \right]_k \tag{27}
\end{equation}

\begin{equation}
H^\omega = [0 \ldots 1] \tag{28}
\end{equation}

being $x_M^\omega \in \mathbb{R}^{n_x \times n + 1}$ and $y_M^\omega \in \mathbb{R}$. Index $w$ stands for wide.

The local observable system (14) is

\begin{equation}
x_M^\omega = F^\omega x_M^{\omega,k} + w_k \tag{29a}
\end{equation}

\begin{equation}
y_M^\omega = H^\omega x_M^\omega + v_k \tag{29b}
\end{equation}

\begin{equation}
F^\omega = \left[ \begin{array}{ccc}
I_n & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0
\end{array} \right] \tag{30}
\end{equation}

\begin{equation}
H^\omega = \left[ \begin{array}{cccc}
1 & 0 & 0 & 0
\end{array} \right] \tag{31}
\end{equation}

\begin{equation}
x_M^\omega = \left[ \begin{array}{c}
\hat{x}_M \\
\hat{y}_M \\
\hat{v}_M
\end{array} \right]_k \tag{32}
\end{equation}

\begin{equation}
Q^\omega = \left[ \begin{array}{cccc}
Q & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & \sigma_f^2
\end{array} \right] \tag{33}
\end{equation}

Even though the new local system has one more dimension $(5 \times 5)$, SKF hypothesis is still applicable as proved in Appendix C.

6.2. Comparison of the updating algorithms in the SDMP

The SDMP cycle is used for comparing the algorithms with real engine data. $T_{NOx, \text{est}}$ is the null matrix and is updated with the cycle. Filter is calibrated by trial-and-error (see Payri et al., 2012 for a calibration method based in a Monte Carlo approach for the KF). The numerical values used are

\begin{equation}
\begin{array}{c}
\sigma_f^2 = 25^2; \\
\sigma_m^2 = 15^2; \\
\sigma_f^2 = 50^2
\end{array} \quad \begin{array}{c}
a = 0.96; \quad \tau = 0.75 \, \text{s} \\
c = [0 : 5 : 30 34 : 4 : 50 55 : 5 : 80] \\
r = [750 780 1000 1250 1500 1750 2000 : 500 : 4500]
\end{array} \tag{34}
\end{equation}

while the sampling frequency is 50 Hz. Measurement $y_{NOx}$ is given by the NOx sensor. A test subset of $t=400 \, \text{s}$ is used for updating the adaptive map and then dynamic model with table $T_{NOx,400}$ is used for predicting NOx in the whole cycle. KF, SKF and SSKF methods are used for updating the model. Results can be seen in Fig. 14. All three methods behave well and are capable of predicting NOx emissions. Dynamics assumptions are also good enough.
for having a good fitting in the SDMP. Note that indeed SSKF results are also acceptable as covariance tracking is not relevant in this cycle as in slow varying tests. These conditions are similar to the ones exposed in Section 5.1. Fitting could be optimized tuning independently the filter for each method.

It is worth comparing the convergence of the methods and for this, sample subsets of the SDMP are used in a sequence for testing the absolute mean error when the updating time is varied monotonically around the cycle

\[
e_{M}(k) = \frac{\sum_{i=2}^{n}(y_{NOx}(i) - y_{NOx}(i|k,T))}{k\Delta T}
\]

(35)

where \(\Delta T\) is the sample time and \(t\) is the time that table has been updated. Results are shown in Fig. 15. Focusing the attention in the lines generated with the filter calibration Cal (34), \(e_{M}\) tends to be lower as \(t\) grows at least during first iterations when system knowledge is poor. The horizontal line shows the benchmark model error, which is constant as model calibration is fixed. It is clear how all three methods, including SSKF, converge to an error similar to that of the benchmark model, but with the advantage that drift and aging is accounted by the online versions but not by the model. KF and SKF behavior is quite similar as expected. On the other hand, Fig. 15 also shows that KF and SKF are faster than SSKF and this is due to the filter tuning. Anyway, there exists a trade-off between convergence speed and estimation robustness, e.g. KF and SKF have a significant oscillation with respect to SSKF around the benchmark model line. This is a matter of the noise tuning and bottom plot is built with a lower gain observer with the calibration set

\[
\sigma_{v}^{2} = 25^{2}; \quad \sigma_{w}^{2} = 7^{2}; \quad \sigma_{f}^{2} = 50^{2}
\]

(36)

for KF and SKF, keeping the ones of SSKF. Results are shown in lines generated by Cal (36) in Fig. 15. Here convergence speed is lower but noise transmission and overfitting is avoided when \(t\) is large. This goes in the direction of robustness and the optimized calibration data set must solve this trade-off, considering uncertainties in the sensor behavior knowledge or in the model quality and data-set quality, among others.

With respect to the global observability, variances of 4 table parameters are compared for both KF and SKF using calibration (36). Fig. 16 shows the results. Top plot shows an element that is never observed during first 200 s and due to this, variance increases monotonically and \(x_{126}\) is still null in \(t = 200\). Second and third elements represent two elements that are active for some instants. This is clearly seen when variance decreases, while when elements are not active, variance increases again monotonically. Note that in a production system application, that will run for the life time of a vehicle, the elements of the estimation error covariance matrix need to be limited so that they do not grow too much and cause numerical problems, see e.g. Höckerdal et al. (2011) that proposes a saturation for avoiding too large variances. The bottom plot shows the variance of \(y_{M}\) that is fairly constant and is a proof of the system global observability.

KF and SKF lines are pretty similar and differences can be found only when zooming in the figure. This is another proof of that SKF behaves quite similar to KF but with a much lower computational burden involved. SSKF behavior is not bad and prove that the method can be useful when calibration data set fulfills some conditions in order to ensure robustness. Anyway, for the nature of the application here, SKF is the best solution for online updating of maps.

6.3. Online estimation

The state-space model for learning maps is useful not only for calibrating the map but also for real time observation of NOx signal. Fig. 17 shows an interesting result in line with this discussion for the SDMP and keeping calibration (36). \(\hat{y}_{SKF}\) represents the observation of
NOx emissions are estimated by using static maps function of a diesel engine operating conditions and a delayed low-pass filter for including dynamics. The solution is not new and several other authors have published about this, but here an online adaptive algorithm is presented for maps calibration and updating. Kalman filtering is used for updating because of its capability for tracking system and parameters aging. Computational issues involved when of updating look-up tables online with the Kalman filter based methods are then addressed. A simplified version of the Kalman filter SKF with similar accuracy as the standard KF, but that requires a much lower memory resources and calculation time, is developed. The local observable system serves as inspiration for analyzing the variance matrix performance and is used for comparison. SKF is based on the KF but neglects covariances of locally unobservable states. The methods are validated using a real world nonlinear model and sportive driving mountain profile SDMP for NOx estimation, being an application of a high interest. The results are compared with the ones of a gray box NOx model, obtaining similar results, which also shows the calibration capabilities of the method. Another contribution is considering sensor dynamics in the original state-space system. The obtained results as well as the simplicity of the formulation could be the key for implementing it in HiW structures and opens the possibility of its use in commercial ECUs. Future work must be in using adaptive strategies for calibrating full engine models and cope with sensor non-linearities, especially the delay.

7. Conclusions

Calculation and memory resources: The KF is a heavy method with similar accuracy than SKF, and the required computational time is rapidly growing when the system complexity increases, as well as the required memory. SSKF is the fastest method and it achieves good results with random-like data and with calculation times similar to the ones of SKF.

Robustness: KF and SKF perform quite well and do not have robustness problems, while the SKF exhibited oscillatory behavior when input data are structured.

Appendix A. Submodel equations for the NOx model

For estimating EGR rate \( \rho_{\text{EGR}} \), a mean value EGR flow model with sensor signals of boost pressure \( p_b \) and mass air flow \( m_a \) as inputs is used. Intake manifold mass flow \( m_i \) is obtained from...
volumetric efficiency \( \eta_v \) (mapped with the steady-state tests), assuming constant intake temperature \( T_i \) (the error is pretty low compared with other error sources) and using \( p_2 \) signal

\[
Q_d = \frac{V_d n_{mc}}{2 \times 60} (\text{m}^3/\text{s})
\]

\[
\rho_l = \frac{p_l}{RT} \quad (\text{kg/m}^3)
\]

\[
m_i = Q_{d, p_2} (\text{kg/s})
\]

(A.1)

where \( Q_d \) is the volume flow at intake and \( \rho_l \) is the density at intake and all variables are introduced in the international standard system.

Considering mass accumulation effects in the node of air-EGR-intake and all variables are introduced in the international standard system, assuming constant intake temperature \( T_i \) here as input and sensor as output.

There an observer could be proposed using EGR model presented there exists fast EGR sensors prototypes but because of temperature and pressure limitations, measurements are not reliable; there an observer could be proposed using EGR model presented here as input and sensor as output.

Actual relative fuel-to-air ratio \( F_r \) can be observed by using the model

\[
F_r = 14.5 \frac{m_f}{m_a}
\]

and using the oxygen output of the ZrO2 sensor for drift correction. Oxygen measurement is steady-state reliable but presents filtering and delay; \( F_r \) model is calculated from \( m_a \) and \( m_i \), which are fast signals but present drift. See Payri et al. (2012) for a complete procedure for estimating \( F_r \).

Appendix B. Algorithm for SSKF

Algorithm 2. Pseudocode for the steady-state Kalman filter for updating look-up tables SSKF

\begin{verbatim}
input :  \( \hat{x}_{k-1}, m_k \)
output:  \( \hat{x}_k \)
1 while Algorithm is running do
2   if Active area changes then
3       3.1 \( x_{o-1} \) is stored in the correct positions of \( x_{k-1} \)
4       Redefinition of new \( x_{k-1} \):
5       end
6   Computation of \( \eta_{l,k} \) and \( \eta_{l,k} \) as in (10)
7   Definition of \( q_k(\eta_{l,k},\eta_{l,k}) \) as in (12b)
8   Computing \( K^{SS} \) as in (5)
9   \( x_k = x_{k-1} + K^{SS} (y_k - q_k x_{k-1}) \)
10 end
11 Updating of \( x_k \) considering \( x_k \)
\end{verbatim}

Appendix C. Variance tracking for the dynamic system

The system (29) is the dynamic local observable system. For this system, (18) is still valid. The only modification is that the new observable part \( P_{w,k}^{\text{new}} \) has \( n \) extra dimensions corresponding to the \( n \)-th order discrete filter. In the paper, a first order discrete filter is used, and then \( P_{w,k}^{\text{new}} \) is 5 x 5.

\[
P_{w,k} = \begin{bmatrix} P_w^k & P_{w,k}^{\text{new}} \\ P_{w,k}^{\text{new}} & P_w^{\text{new}} \end{bmatrix}
\]

(C.1)

\[
P_{w,k}^{\text{new}} = \begin{bmatrix} P_{w,k}^{\text{new}} \\ P_{w,k}^{\text{new}} \end{bmatrix}
\]

(C.2)

where \( P_{w,k}^{\text{new}} \) is the scalar variance coupled to sensor dynamics and \( P_{w,k}^{\text{new}} \) is the vector \( 1 \times 4 \) with the covariances between sensor estimation and observable parameters. \( P_w^k \) is a positive-semidefinite matrix, so is \( P_{w,k}^{\text{new}} \). \( K_k \) is reordered to contain the unobservable part \( K_k^o \) and the latter in the parameters related \( K_k^e \) and dynamics related part \( K_k^d \)

\[
K_k^{w} = \begin{bmatrix} K_k^o & K_k^e \\ K_k^d & K_k^f \end{bmatrix}
\]

(C.3)

and (19) is now

\[
P_{k+1} = \begin{bmatrix} P_{k+1} & Q^m \\ Q^m & P_{k+1} \end{bmatrix}
\]

\[
F_{k+1} P_{k+1} = P_{k+1} + Q^m
\]

(C.4)

\[
P_{k+1} = \begin{bmatrix} P_{k+1} & Q^m \\ Q^m & P_{k+1} \end{bmatrix}
\]

(C.5)

which shows that SKF partitioning is also applicable for the augmented dynamic system. \( P_k \) is a positive-semidefinite matrix so is \( P_k^e \) and \( P_k^{\text{new}} \).

References


