Control Engineering Practice 21 (2013) 1455–1468

Contents lists available at ScienceDirect

Control Engineering Practice

journal homepage: www.elsevier.com/locate/conengprac

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

C. Guardiola [a], B. Pla [a], D. Blanco-Rodriguez [a,*], L. Eriksson [b]

[a] CMT Motores Térmicos, Universitat Politècnica de València, Camino de Vera s/n, E-46022 Valencia, Spain
[b] Vehicular Systems, Department of Electrical Engineering, Linköping University, S-581 83 Linköping, Sweden

ARTICLE INFO

Article history:
Received 23 July 2012
Accepted 21 June 2013

Keywords:
NO_x
Kalman filter
Adaptive model
Look-up tables
Diesel

ABSTRACT

No_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 benchmark model for No_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 neglects the variance information. SKF is used for the online calibration of an adaptive model for No_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_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.

1. Introduction

Legislation on emissions strengths the importance of electronic controls and embedded software in automotive engines (Ebert & Jones, 2009). Focusing on nitrogen oxides (NO_x) emissions, selective catalyst reduction system (SCR) or lean NO_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_2, while exhaust oxygen measurement is more usual.

An accurate and fast NO_x signal would permit its usage for real-time tasks but still dynamic issues must be solved. NO_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_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_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, Mazer, and Cornette (2012), and Alberer and del Re (2009) use ZrO_2 sensors for estimating NO_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_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.

* Corresponding author. Tel.: +34 963879233.
E-mail addresses: carguaga@mot.upv.es (C. Guardiola), benplamo@mot.upv.es (B. Pla), dablarod@gmail.com, dablarod@mot.upv.es (D. Blanco-Rodriguez), larer@isy.liu.se (L. Eriksson).

0967-0661/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.conengprac.2013.06.015

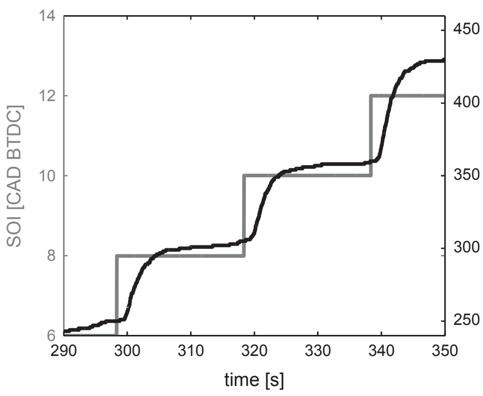

Fig. 1. NO_x sensor response to SOI steps. NO_x presents a clear delay and filtering with respect to SOI signal. Dynamic response could be fitted with a first order delayed discrete filter. SOI unit is crank angle degree before the top dead center, while NO_x is measured in ppm. Delay is in the order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

Without NO_x sensors, modeling (or virtual sensing) is possible, see e.g. Schilling, Amstutz, and Guzzella (2008) that develops a real time NO_x model using maps, Winkler-Ebner, Hirsch, Del Re, Klinger, and Mistelberger (2010) that designs a virtual NO_x 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 updating method that utilizes the KF framework but simplifies the covariance matrix and the associated updates, and it will be called Simplified KF (SKF). This approach is compared to both a full Kalman filter (KF) based update, as described in Höckerdal, Frisk, and Eriksson (2011), and a steady state Kalman filter (SSKF) based update, as described in Guardiola, Pla, Blanco-Rodriguez, and Cabrera (2013). Look-up tables updating is treated in Section 4, including a discussion on computational complexity, important when talking about real time implementation.

The algorithms are first evaluated in simulation in Section 5 and then Section 6 applies the algorithms for the NO_x estimation problem in a diesel engine, showing SKF possibilities. The application uses a Sportive Driving Mountain Profile (SDMP) cycle with sharp variations on injection and speed to identify a static $\mathbf{T_{NOx}}$ map as function of injected fuel mass m_f and speed n during one part of the cycle storing NO_x values, and then, the whole cycle NO_x emissions are predicted with such map. SKF is also applied to the New European Driving Cycle (NEDC). The only requirement is that sensor properties (delay and dynamics) must be known for estimating actual NO_x. For that, paper includes sensor dynamics in the standard state-space model. First, experimental setup and engine data used in the paper is commented.

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 ZrO_2 multilayer based NO_x sensor (Kato et al., 1996) giving a twofold measurement of NO_x (y_{NO_x}) 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 NO_x 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 cm^3
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

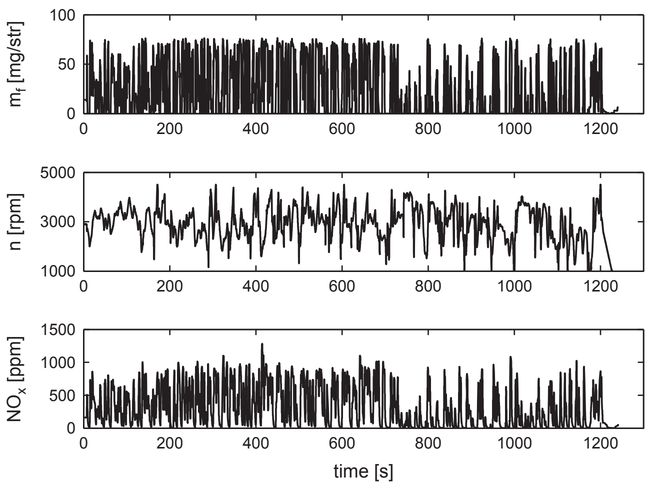

Fig. 2. SDMP test profile with sharp variations in m_f, n and NO_x.

3. A control-oriented model for NO_x estimation

Literature about NO_x 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 complex models (Schilling, Amstutz, Onder, & Guzzella, 2006) or 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 NO_x model is provided for comparison with the online methods presented hereinafter.

3.1. NO_x model used as benchmark

Thermal NO_x 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-NO_x trade-off, see e.g. Neeft, Makkee, and Moulijn (1996). Three in-cylinder variables should be necessary for NO_x reconstruction: peak temperature, oxygen concentration and residence time, but these cannot 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 NO_x model, oxygen rate in the cylinder and engine operating conditions (usually speed n and injected fuel mass m_f) seem to be suitable variables for characterizing NO_x; see e.g. Ericson, Westerberg, Andersson, and Egnell (2006), Galindo, Luján, Climent, and Guardiola (2007).

The model is set-point relative and the main equation is

$$x_m = NO_{x,0}(m_f,n)e^{-k(m_f,n)\times(r_{\mathrm{EGR}}F_r - r_{\mathrm{EGR},0}F_{r,0})}C(comb,H,T_w,\ldots) \quad (1)$$

where subindex 0 indicates nominal conditions defined by manufacturer calibration. $NO_{x,0}$ is NO_x emissions as function of nominal operating point conditions (n and m_f), and exponential is built around nominal values of inert gas $r_{\mathrm{EGR}}F_r$, 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 ZrO_2 sensor at the exhaust: see Appendix A for more details on the sub-model equations. $r_{\mathrm{EGR},0}$, $F_{r,0}$, 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 nominals, 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 NO_x diagnosis after tests, i.e. if no NO_x 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

$$y_m = \frac{1-a}{1-az^{-1}} z^{-\tau} x_m \quad (2)$$

obtaining y_m where SOI steps are performed for characterizing a and τ. Sensor response time a can be modeled as constant but delay τ has a non-linear behavior difficult to model. A detailed study on NO_x sensor characterization is made in Galindo et al. (2011). In this work, τ 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 τ variability; the interested reader can go to Trimboli, Di Cairano, Bemporad, and Kolmanovsky (2012) who model delay in F_r signal from NO_x sensor as function of air mass flow, engine operating conditions and speed.

Model is calibrated using steady-state measurements and dynamic tests minimizing $\sum_{i=1}^{n_p}(y_{NO_x} - y_m)^2$ 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_{NO_x} and y_m. The fitting is good and the model is capable of estimating y_{NO_x} 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 system and the model as 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

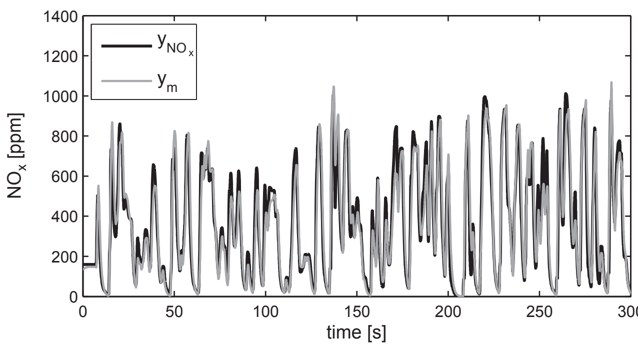

Fig. 3. Model simulation y_m and sensor signal y_{NO_x} in the SDMP test. In spite of the hypothesis taken, the results are quite good and capable of tracking NO_x emissions.

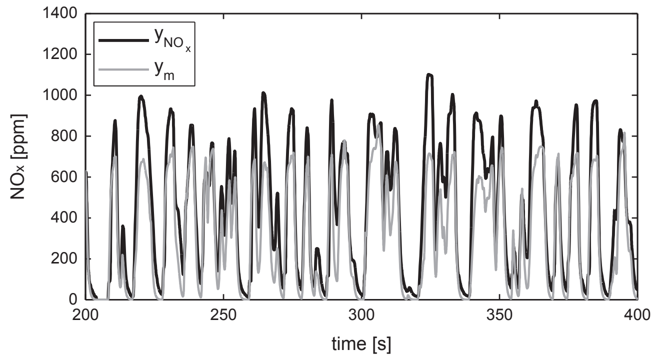

Fig. 4. Model simulation y_m and sensor signal y_{NO_x} in the SDMP test with a non-optimized calibration data set. Even though dynamics are well cached, y_m is drifted with respect to y_{NO_x}.

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 Tunestal (2008) use the subspace method for identifying NO_x 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 is 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. parameters that have not 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

$$x_k = f(x_{k-1}, u_k) + w_k \tag{3a}$$

$$y_k = h(x_k, u_k) + v_k \tag{3b}$$

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 ij of $\mathbf{F_k}$ and $\mathbf{H_k}$ are obtained

$$F_{k,ij} = \frac{\partial f_i}{\partial x_j}\Big|_{x=\hat{x}_k}; \quad H_{k,ij} = \frac{\partial h_i}{\partial x_j}\Big|_{x=\hat{x}_k}; \tag{4}$$

being $\mathbf{F}$ the linearized process matrix and $\mathbf{H}$ 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 $w_k \in \mathbb{R}^{n_x}$ and $v_k \in \mathbb{R}^{n_y}$ are assumed to be independent and both generated by Gaussian distribution with zero mean and covariance matrices $\mathbf{Q_k}$ resp. $\mathbf{R_k}$, defined by

$$E[w_k w_k^T] = \mathbf{Q_k} \tag{5a}$$

$$E[v_k v_k^T] = \mathbf{R_k} \tag{5b}$$

In applications these are often chosen to be constant, i.e. $\mathbf{Q}$ and $\mathbf{R}$, and diagonals.

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

$$\hat{x}_{k|k-1} = f(\hat{x}_{k-1}, u_k) \tag{6a}$$

$$e_k = y_k - h(\hat{x}_{k|k-1}, u_k) \tag{6b}$$

$$\hat{x}_k = \hat{x}_{k|k-1} + K_k e_k \tag{6c}$$

where K_k Kalman gain is solved by the following iterative equation:

$$\mathbf{P_{k|k-1}} = (\mathbf{F_k P_{k-1} F_k^T} + \mathbf{Q}) \tag{7a}$$

$$K_k = \mathbf{P_{k|k-1} H_k^T}(\mathbf{H_k P_{k|k-1} H_k^T} + \mathbf{R})^{-1} \tag{7b}$$

$$\mathbf{P_k} = (\mathbf{I} - \mathbf{K_k H_k})\mathbf{P_{k|k-1}} \tag{7c}$$

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

$$\mathbf{P_k} = E[x_k - \hat{x}_k][x_k - \hat{x}_k]^T \tag{8}$$

4.2. Look-up tables

A look-up table $\mathbf{T} \in \mathbb{R}^{\Pi_{i=1}^{N} n_i}$ is defined as a N-Dimensional mapping $\{\mathbf{T} : \mathbb{R}^N \to \mathbb{R}\}$ 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 $\mathbf{T}$. 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 $N=2$,

$$\mathbf{T} = [\theta_{i,j}] \tag{9}$$

where $i = 1, \ldots, n_r$ and $j = 1, \ldots, n_c$ (r stands for *row* and c for *column*). The multivariate interpolation function for generating the output y_k from input $u_k = [u_1(k)\ u_2(k)]^T$ can be expressed as

$$y_k = \text{vec}(q(u_k))^T\ \text{vec}(\mathbf{T})$$

where 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 $q_k(u_k)$ selects the 4 (2^N in the ND case) active elements $\theta_{i,j}$, $\theta_{i,j+1}$, $\theta_{i+1,j}$, $\theta_{i+1,j+1}$ (with i,j fulfilling $u_{1,k} \in [r_i, r_{i+1}]$ and $u_{2,k} \in [c_j, c_{j+1}]$) 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

$$\eta_{1,k} = \frac{u_{1,k} - r_i}{r_{i+1} - r_i}, \quad \eta_{2,k} = \frac{u_{2,k} - c_j}{c_{j+1} - c_j} \tag{10}$$

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 $u_k \to y_k$ is

$$x_k = x_{k-1} + w_k \tag{11a}$$

$$y_k = \text{vec}(q(u_k))^T x_k + v_k \tag{11b}$$

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

For convenience, the non-zero elements in $\text{vec}(q(u_k))$ are denoted $q_k^o \in \mathbb{R}^{1\times 4}$ (o stands for *observable*) and the corresponding elements in the state vector, $x_k^o \in (R)^4$. Then, following expression is obtained for the output:

$$y_k = q_k^o x_k^o + v_k \tag{12a}$$

$$q_k^o = [(1-\eta_{1,k})(1-\eta_{2,k}) \quad \eta_{1,k}(1-\eta_{2,k}) \quad (1-\eta_{1,k})\eta_{2,k} \quad \eta_{1,k}\eta_{2,k}] \tag{12b}$$

where **Q** and **R** are

$$\mathbf{Q} = \sigma_w^2 \mathbf{I}_{\mathbf{n_r}\times\mathbf{n_c}} \tag{13a}$$

$$\mathbf{R} = \sigma_v^2 \tag{13b}$$

and $\mathbf{I}_{\mathbf{n_r}\times\mathbf{n_c}}$ is the identity square matrix. If there is engineering application knowledge available, σ_w^2 might be selected individually for each table element building a vector $\Sigma_w^2 = [\sigma_{w,1}^2 \ldots \sigma_{w,n_r\times n_c}^2]$ that contains individual variances in such way $\mathbf{Q} = \mathbf{\Sigma_w^2 I}_{\mathbf{n_r}\times\mathbf{n_c}}$. In automotive, this is useful for considering order of magnitude of the parameters, i.e. absolute error will not be the same for low emissions at lower loads than highest peaks at higher ones, and on the other hand, if driving pattern exists, noise could be mapped over the table grid. This is linked with the foreseen probability that engine is running at a certain operating condition, i.e. elements aging in more usual grid areas will be slower and the opposite.

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 $\mathbf{F} = \mathbf{I}_{\mathbf{n_r}\times\mathbf{n_c}}$ is constant and $\mathbf{H_k} = \mathbf{vec}(\mathbf{q}(\mathbf{u_k}))^\mathbf{T}$. Although only the active elements x_k^o are updated at every k, all $\mathbf{P_k}$ 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 Papavassilopoulos (2011), Jørgensen, Homsen, Madsen, 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 4×4 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 K related elements). Furthermore, $\mathbf{P_k}$ 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 4×4

$$x_k^o = \mathbf{I_4} x_{k-1}^o + w_k \tag{14a}$$

$$y_k = q_k^o x_k^o + v_k \tag{14b}$$

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): σ_w^2, σ_v^2, **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 $u_{1,k}$ and $u_{2,k}$, 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^{SS}. For the 2D case

$$K^{SS} = \begin{bmatrix} k_i(\eta_1, \sigma_v^2/\sigma_w^2)\cdot k_i(\eta_2, \sigma_v^2/\sigma_w^2) \\ k_i(\eta_1, \sigma_v^2/\sigma_w^2)\cdot k_i(1-\eta_2, \sigma_v^2/\sigma_w^2) \\ k_i(1-\eta_1, \sigma_v^2/\sigma_w^2)\cdot k_i(\eta_2, \sigma_v^2/\sigma_w^2) \\ k_i(1-\eta_1, \sigma_v^2/\sigma_w^2)\cdot k_i(1-\eta_2, \sigma_v^2/\sigma_w^2) \end{bmatrix} \tag{15}$$

where k_i is computed as follows:

$$k_i(\eta, \sigma_v^2/\sigma_w^2) = \frac{0.5(1-\eta)(1+s)}{0.5(1+s)(1-2\eta+2\eta^2)+\sigma_v^2/\sigma_w^2} \tag{16a}$$

$$s = \sqrt{1+\frac{4}{(1-2\eta+2\eta^2)}\frac{\sigma_v^2}{\sigma_w^2}} \tag{16b}$$

This approach allows mapping K and this only depends on inputs (η_1 and η_2 calculated in (10)) and noise trade-off σ_v^2/σ_w^2. 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) can 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:

$$\mathbf{O} = \begin{bmatrix} q_{i_1}^o \\ q_{i_2}^o \\ q_{i_3}^o \\ q_{i_4}^o \end{bmatrix} \tag{17}$$

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 **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 $\mathbf{P_k}$ elements are predicted and updated at every k. $\mathbf{P_k}$ can be reorganized in such way that variances related with observable $\mathbf{P_k^o}$ and unobservable $\mathbf{P_k^u}$ elements are split.

$$\mathbf{P_k} = \left[\begin{array}{c|c} \mathbf{P_k^u} & \begin{matrix}\cdot\\\cdot\\\cdot\end{matrix} \\ \hline \cdot\quad\cdot\quad\cdot & \mathbf{P_k^o}\end{array}\right] \tag{18}$$

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

$$\begin{aligned}\mathbf{P_{k+1}} &= \left[\begin{array}{c|c} \mathbf{I} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{I_4}-K_k^o q_k^o\end{array}\right]\left[\begin{array}{c|c} \mathbf{P_k^u}+\mathbf{Q^u} & \begin{matrix}\cdot\\\cdot\\\cdot\end{matrix} \\ \hline \cdot\quad\cdot\quad\cdot & \mathbf{P_k^o}+\mathbf{Q^o}\end{array}\right] \\ &= \left[\begin{array}{c|c} \mathbf{P_k^u}+\mathbf{Q^u} & \begin{matrix}\cdot\\\cdot\\\cdot\end{matrix} \\ \hline \cdot\quad\cdot\quad\cdot & (\mathbf{I_4}-K_k^o q_k^o)(\mathbf{P_k^o}+\mathbf{Q^o})\end{array}\right]\end{aligned} \tag{19}$$

```
input : $\hat{x}_{k-1}$, $u_k$, $P_{k-1}$, $R$, $\mathbf{Q^o}$, $Q^u$
output: $\hat{x}_k$, $P_k$
1  while Algorithm is running do
2      if Active area changes then
3          $x^o_{k-1}$ is stored in the correct positions of $x_{k-1}$
4          $diag(\mathbf{P^o_{k-1}})$ and $P^u_{k-1}$ are stored in a global variance vector $P_{k-1}$ according to their
           correct positions
5          Redefinition of $\mathbf{P^o_{k-1}}$, $P^u_{k-1}$, $x^o_{k-1}$
6      end
7      Computation of $\eta_{1,k}$ and $\eta_{2,k}$ as in (10)
8      Definition of $q^o_k(\eta_{1,k}, \eta_{2,k})$ as in (12b)
9      $\mathbf{P^o_{k|k-1}} = \mathbf{P^o_{k-1}} + \mathbf{Q^o}$
10     $K^o_k = \mathbf{P^o_{k|k-1}} \mathbf{q^{o\,T}_k} \left(\mathbf{q^o_k} \mathbf{P^o_{k|k-1}} \mathbf{q^{o\,T}_k} + \mathbf{R}\right)^{-1}$
11     $x^o_k = x^o_{k-1} + K^o_k (y_k - q^o_k x^o_{k-1})$
12     $\mathbf{P^o_{k|k-1}} = (\mathbf{I_4} - \mathbf{K^o_k}\mathbf{q^o_k})(\mathbf{P^o_k} + \mathbf{Q^o})$
13     $P^u_k = P^u_{k-1} + Q^u$
14 end
15 Updating of $x_k$ and $P_k$ considering $x^o_k$, $diag(\mathbf{P^o_k})$ and $P^u_k$
```

where all observable matrices are 4×4, while the unobservable ones are $(n_rn_c-4)\times(n_rn_c-4)$. (19) shows how only diagonal elements of $\mathbf{P_k^u}$ are affected by adding the diagonal matrix $\mathbf{Q^u}$. However, both diagonal and non-diagonal elements of $\mathbf{P_k^o}$ are affected as in a 4×4 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 4×4 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 $\mathbf{P^u}$, non-diagonal elements are neglected building a vector $P^u = diag(\mathbf{P^u})$. 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 $\mathbf{Q^u}$ 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 $P_k^u + Q^u$ (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^u always that P_k exists, but are kept for the sake of clarity. Here, noise variance σ_w^2 is the same for all elements, allowing to pre-define diagonal matrix $\mathbf{Q^o} = \boldsymbol{\sigma_w^2}\mathbf{I_{4\times4}}$ for active system and vector $Q^u = \sigma_w^2 diag(\mathbf{I_{n_r\times n_c-4}})$ for the non-active one.
Algorithm 1. Pseudocode for the simplified Kalman filter SKF.

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 $vec(T)$ 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 $\mathbf{P^o}$ and x^o 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 (16×16), 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 n_p, 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 $(n_r n_c) \times (n_r n_c)$ is reduced to a simple 4×4. 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.

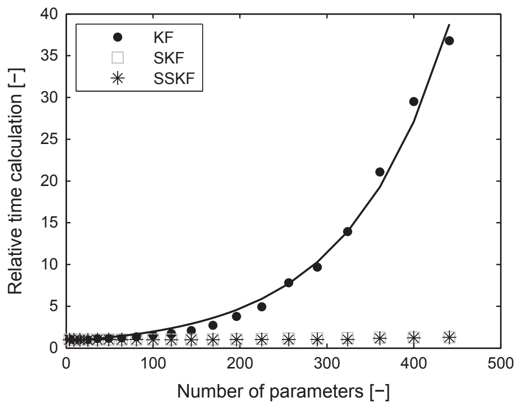

Fig. 5. Relative computation time required for the methods for updating 2D look-up tables. Times are normalized with the average time that SSKF needs for computing 1000 iterations of the filter.

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

	P	K	Rel. Time
KF	$(n_r n_c) \times (n_r n_c)$	$(n_r \times n_c) \times 1$ (by (7))	$0.82\ \exp(0.0087 n_p)$
SKF	$4 \times 4 + (n_r \times n_c) \times 1$	4×1 (by (7))	1.15
SSKF	4×4	4x1 (by (16))	1

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):

$$r = [500\ \ 1500\ \ 2500\ \ 3500] \tag{20a}$$

$$c = [0\ \ 20\ \ 40\ \ 60] \tag{20b}$$

and the true map $\mathbf{T_r}$ is defined with the surface

$$\mathbf{T_r} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} \tag{21}$$

where $x_r = vec(\mathbf{T_r})$ is the expanded state vector objective of the estimation process. No initial process knowledge is considered: x_0, x_0^o, $\mathbf{P_0}$ are initialized with zeros. $\sigma_v^2 = \sigma_w^2 = 1$ 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 $\{u_k\}_{k=1}^{1400}$ 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 σ_v^2 and σ_w^2.

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_6 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_6 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

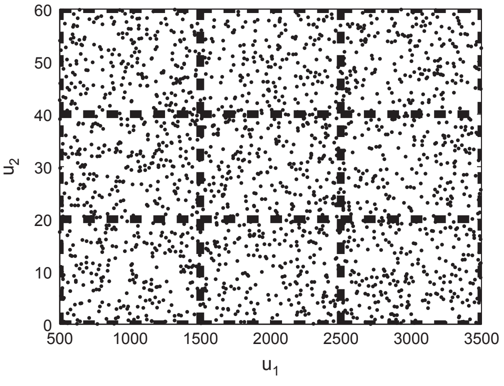

Fig. 6. Random variation of inputs u_1 and u_2 showed with dots and table grid with the dashed line.

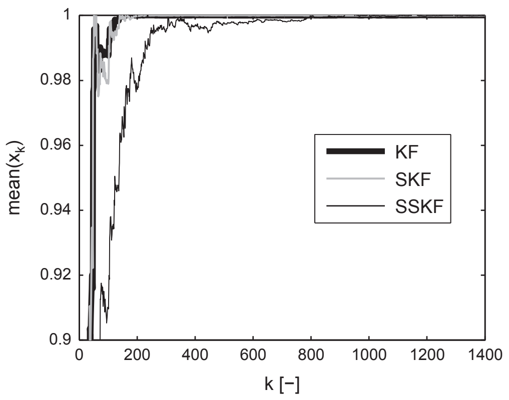

Fig. 7. Average value of table parameters for the three considered updating methods when considered a uniform distributed inputs sequence. The correct value is 1.

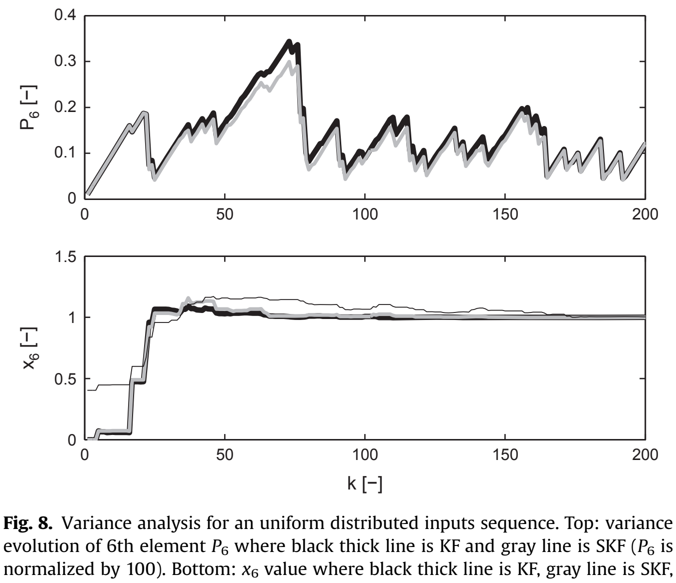

Fig. 8. Variance analysis for an uniform distributed inputs sequence. Top: variance evolution of 6th element P_6 where black thick line is KF and gray line is SKF (P_6 is normalized by 100). Bottom: x_6 value where black thick line is KF, gray line is SKF, thin black is SSKF.

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^{SS}, 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 $\mathbf{P_0}=[\mathbf{0}]$. 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

$\mathbf{T_r}$ is identified with a sequence $\{u_k\}_{k=1}^{200}$ 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 $u_1=2000\ \forall k$. 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.

$$\mathbf{T_k} = \left[\begin{array}{c|c|c|c} \cdot\ \ \cdot & \cdot & \cdot & \\ \hline x_{5,k} & x_{6,k} & x_{7,k} & x_{8,k} \\ x_{9,k} & x_{10,k} & x_{11,k} & x_{12,k} \\ \hline \cdot\ \ \cdot & \cdot & \cdot & \end{array}\right] \tag{22}$$

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 been already been covered; e.g. x_6 is closer to the convergence as 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_6, x_7 and x_8 due to no state error information is tracked, and the system and convergence becomes more dependent on data.

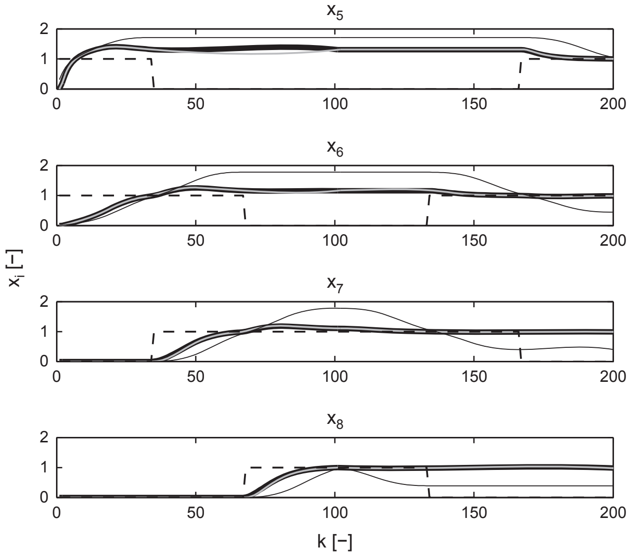

Fig. 9. States evolution for a monotonically varied inputs sequence. Black thick line is KF, gray line is SKF, thin black is SSKF and dashed line shows when the element is observable (1 is active; 0 is not active).

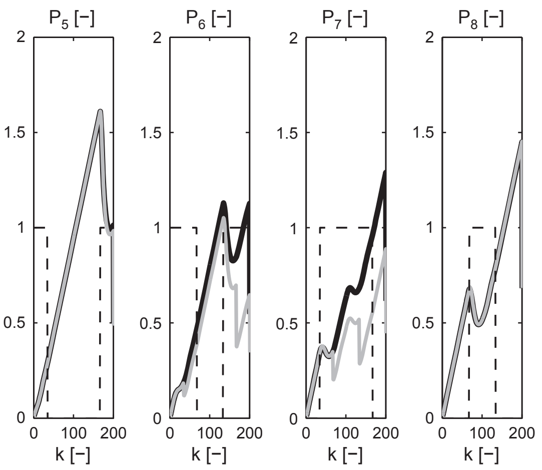

Fig. 10. P_{ii} analysis for a monotonically varied inputs sequence. Values are normalized by 100. Variances of elements 9–12 are equal to those of 5–9 because of symmetric properties. Black line is KF, gray line is SKF and dashed line shows when the element is active (1 active; 0 is not active).

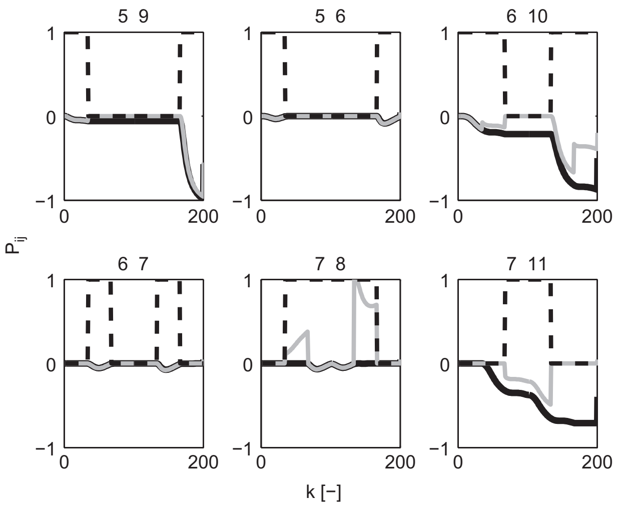

Fig. 11. P_{ij} analysis for a monotonically varied inputs sequence. Values are normalized by 100. Black line is KF, gray line is SKF and dashed line shows when the element is active (1 active; 0 is not active).

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} $\forall i \neq j$ 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

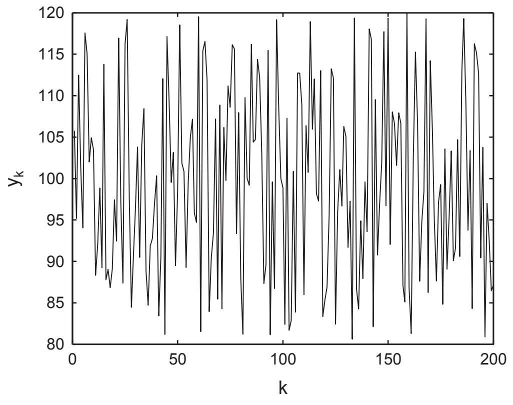

Fig. 12. Measurement y_k plot with the addition of a uniform distributed noise with zero mean and maximum amplitude of 0.2.

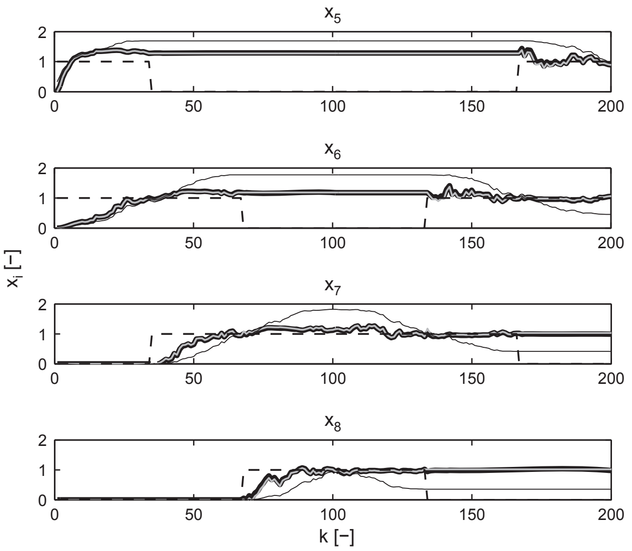

Fig. 13. States evolution for noisy measurements. Black thick line is KF, gray line is SKF, thin black is SSKF and dashed line shows when the element is active (1 is active; 0 is not active).

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. SSKF 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, a minimum condition for a correct updating, and SSKF 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 NO_x estimation

The attention is turned to the problem of online NO_x estimation. The approach consists of modeling NO_x 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 NO_x. When the engine is running again, the parameters keep evolving for correcting drift and slowly varying effects. Furthermore, the observer built for map updating can be utilized for having an actual NO_x 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 NO_x table $\mathbf{T_{NO_x}}$ 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, $\mathbf{T_{NO_x}}$ 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 NO_x 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 NO_x sensor

$$u_{1,k}=z^{-\tau}n(k) \tag{23a}$$

$$u_{2,k}=z^{-\tau}m_f(k) \tag{23b}$$

where τ 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 NO_x can be denoted as x_M (M for the used *method*) and is modeled using the adaptive map $\mathbf{T_{NOx,t}}$

$$x_M(k-\tau)|_t=\mathbf{T_{NOx,t}}(\mathbf{u_{1,k}},\mathbf{u_{2,k}}) \tag{24}$$

defining a quasi-static representation of NO_x output where t defines the time that table has been updated.

Sensor dynamics are considered as in (2)

$$y_M(k)=\frac{1-a}{1-az^{-1}}x_M(k-\tau) \tag{25}$$

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

$$x_k^w=\left[\begin{array}{c|c}\mathbf{I_{n_r n_c}} & \\ \hline (1-a)q_k & a\end{array}\right]x_{k-1}^w+w_k \tag{26a}$$

$$y_k^w=H^w x_k^w+v_k \tag{26b}$$

with

$$x_k^w=\begin{bmatrix}x\\ y_M\end{bmatrix}_k \tag{27}$$

$$H^w=[0\cdots1] \tag{28}$$

being $x_k^w\in\mathbb{R}^{n_r\times n_c+1}$ and $y_k^w\in\mathbb{R}$. Index w stands for *wide*.

The local observable system (14) is

$$x_k^{ow}=F_k^w x_{k-1}^{ow}+w_k \tag{29a}$$

$$y_k^{ow}=H^w x_k^{ow}+v_k \tag{29b}$$

$$F_k^w=\left[\begin{array}{c|c} & 0\\ \mathbf{I_4} & 0\\ & 0\\ & 0\\ \hline (1-a)q_k^o & a\end{array}\right] \tag{30}$$

$$H^w=[0\ \ 0\ \ 0\ \ 0\ \ 1] \tag{31}$$

$$x_k^{ow}=\begin{bmatrix}x_1^o\\ x_2^o\\ x_3^o\\ x_4^o\\ y_M\end{bmatrix}_k \tag{32}$$

where H^w is constant, because q_k^o and filtering parameter a are included now in the time varying process matrix $\mathbf{F_k^w}$. $\mathbf{Q^w}$ process noise matrix is also augmented and includes an extra-noise term σ_f^2 for the new state y_M

$$\mathbf{Q^w}=\left[\begin{array}{c|c}\mathbf{Q} & \mathbf{0}\\ \hline \mathbf{0} & \sigma_f^2\end{array}\right] \tag{33}$$

Even though the new local system has one more dimension (5×5), SKF hypothesis are 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. $\mathbf{T_{NO_{x,0}}}$ 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

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

while the sampling frequency is 50 Hz. Measurement y_{NO_x} is given by the NO_x sensor. A test subset of t=400 s is used for updating the adaptive map and then dynamic model with table $\mathbf{T_{NOx,400}}$ is used for predicting NO_x 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 NO_x 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

$$e_M(k) = \frac{\sum_{i=1}^{n_y} (y_{NO_x}(i) - y_M(i)|_{t=k\Delta T})}{k\Delta T} \tag{35}$$

where ΔT 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

$$\sigma_v^2 = 25^2; \quad \sigma_w^2 = 7^2; \quad \sigma_f^2 = 50^2 \tag{36}$$

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 NO_x 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

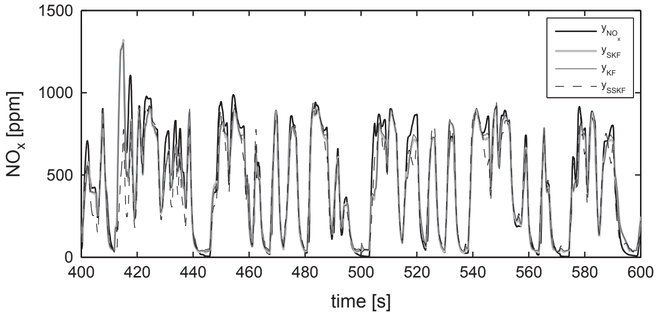

Fig. 14. NO_x prediction for the three updating methods when maps are updated during 400 s, i.e. a subset of 20,000 measurements of y_{NO_x}, and then used for the prediction. Sensor measurement y_{NO_x} is provided for comparison.

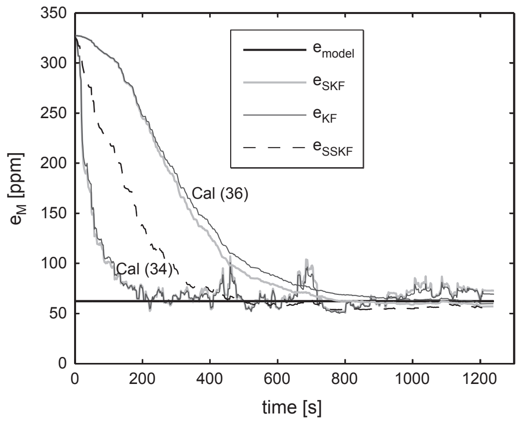

Fig. 15. Total mean absolute error e_M for the different methods and different ranges used for identification. SSKF dashed line is calculated used filter calibration (34). Lines left to the SSKF one (dashed): Using calibration (34). During first iterations, error is high, as far as table has no initial knowledge, but once that time is around 400 s, error is nearly minimum, although noise transmission is evident. Lines right to the SSKF one: Using calibration (36), noise transmission is reduced but convergence speed as well. Final filter tuning must be a trade-off between robustness and convergence speed.

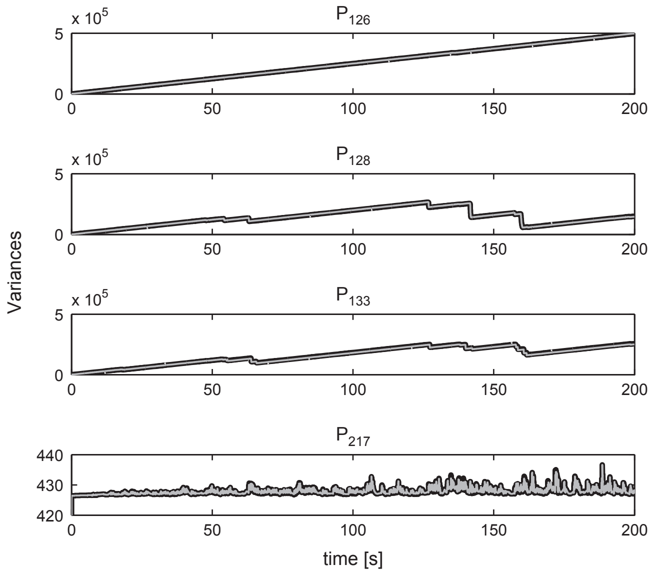

Fig. 16. Variance analysis in the SDMP updating. From top to bottom, variances of elements 126, 126, 133 and 217 are plotted. Black line is KF and gray line is SKF.

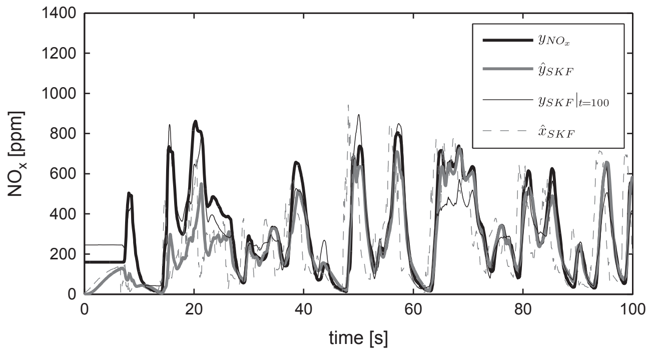

Fig. 17. NO_x estimation by three ways: y_{NO_x} sensor measurement, online observation of sensor signal by the SKF method $\hat{y_M}$, offline prediction by using the map $\mathbf{T_{NOx,100}}$ and online actual estimation $\hat{x}_{SKF}$.

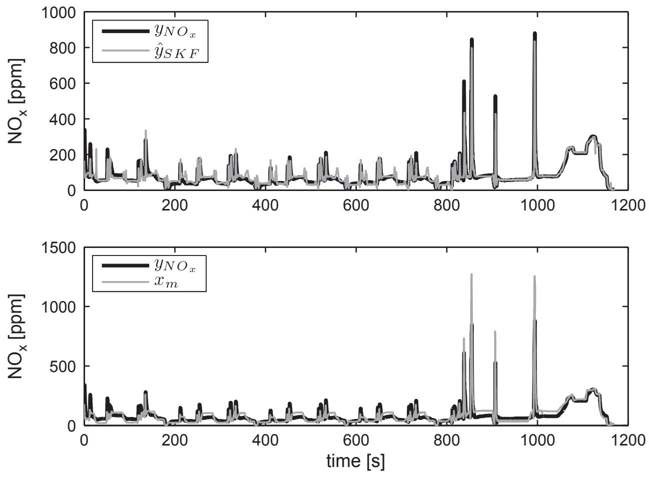

Fig. 18. NO_x estimation in the NEDC. Top plot compares $\hat{x}_{SKF}$ and y_{NO_x} while bottom ones compares y_{NO_x} measurement with x_m. Both estimations remains dynamic properties, benefited from NEDC is slow, but benchmark model exhibits some drift. Here, SKF could be directly used for recalibration of the NO_x model.

y_{NO_x} by using the adaptive map at every iteration. The adaptive map provides a perfect fitting in about 30 s (this can be tuned varying filter calibration). Alternatively, $y_{SKF}|_{t=100}$ shows the offline NO_x sensor prediction using the model $\mathbf{T_{NOx,100}}$ with acceptable results.

Table itself might predict actual NO_x emissions if interpolating the table directly applying no delay nor filtering. Furthermore, actual NO_x can be directly observed in the basis of the state-space system for learning having a compact programming and providing an adaptive estimation. For that, x_{SKF} must be included in the state vector x^w. This does not compromise the assumptions made for the SKF. Coming back to Fig. 17, $\hat{x}_{SKF}$ is the online observation of the actual NO_x using the adaptive map for high frequencies and sensor for low ones. The more that $\mathbf{T_{NOx,100}}$ fits in y_{NO_x}, the more reliable is $\hat{x}_{SKF}$. This signal should be used for online NO_x tracking. The use of the table allows solving the non-causalities for real-time estimation: note that observer is built in the basis of delayed inputs.

6.4. Results in the NEDC with the SKF and the benchmark model

SDMP is a test with sharp variations on the operating point conditions but homologation cycles such as NEDC are much slower and this could compromise the global observability; remind that higher levels of excitation are beneficial for updating. Here, air path dynamics are fast enough to follow load variations and then EGR and turbine commands are able to follow their references. Then, the table $\mathbf{T}_{NO_x}$ might directly replace $\mathbf{NO_{x,0}}$ map in (1) or update it for canceling drift. Indeed, $\mathbf{NO_{x,0}}$ could be used for initializing $\mathbf{T}_{NO_x}$ but here the initial map is the null matrix, which is a worst-case condition. NEDC cycle is running and map is updated. Calibration (36) is used. Fig. 18 shows results comparing y_{NO_x}, y_m (which exhibits drift) and y_{SKF}, which here represents NO_x prediction by using the updated map after 1 run of the cycle. Adaptive map gets a good NO_x estimation which is slightly better than NO_x benchmark model one.

SKF could be used for online adaptation and/or calibration of complex models, where a number of maps and parameters must be updated. Anyway, a deep study is required for ensuring observability, convergence and robustness properties and for getting a computationally efficient learning structure. A big number of parameters and maps should be updated and problem is non-convex. Of course that computational issue is critical and there SKF can be an effective solution. Here, SKF has been used for updating single maps with succeed.

7. Conclusions

NO_x 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 results under simulation and real data remark two key points that make that the SKF is a suitable option for online estimation:

- 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 SSKF exhibited oscillatory behavior when input data are structured.

The methods and especially the SKF are validated using a real world nonlinear model and sportive driving mountain profile SDMP for NO_x estimation, being an application of a high interest. The results are compared with the ones of a gray box NO_x 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.

Appendix A. Submodel equations for the NO_x model

For estimating EGR rate r_{EGR}, a mean value EGR flow model with sensor signals of boost pressure p_2 and mass air flow m_a as inputs is used. Intake manifold mass flow m_i is obtained from volumetric efficiency η_v (mapped with the steady-state tests), assuming constant intake temperature T_2 (the error is pretty low compared with other error sources) and using p_2 signal

$$Q_d = \frac{V_d n \eta_v}{2 \times 60} (\mathrm{m}^3/\mathrm{s})$$
$$\rho_i = \frac{p_2}{RT_2} (\mathrm{kg}/\mathrm{m}^3)$$
$$m_i = Q_d \rho_i (\mathrm{kg}/\mathrm{s}) \tag{A.1}$$

where Q_d is the volume flow at intake and ρ_i 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 m_a, r_{EGR} can be calculated

$$r_{\mathrm{EGR}} = 1 - \frac{m_a}{m_i} \tag{A.2}$$

For model calibration, measurements of oxygen at intake and exhaust from the gas analyzer are used

$$r_{\mathrm{EGR}} = \frac{O_{2,exhaust} - 0.209}{O_{2,intake} - 0.209} \tag{A.3}$$

If these measurements would be available online, an observer could be built for having a more reliable estimation. Furthermore, 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

$$F_r = 14.5 \frac{m_f}{m_a} \tag{A.4}$$

and using the oxygen output of the ZrO$_2$ 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.

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

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 $\mathbf{P^{wo}_k}$ has n extra dimensions corresponding to the n:th order discrete filter. In the paper, a first order discrete filter is used, and then $\mathbf{P^{wo}_k}$ is 5×5.

$$\mathbf{P^w_k} = \left[\begin{array}{c|c} \mathbf{P^u_k} & \\ \hline & \mathbf{P^{wo}_k} \end{array}\right] \tag{C.1}$$

$$\mathbf{P^{wo}_k} = \left[\begin{array}{c|c} \mathbf{P^o_k} & P^{ndo\,T}_k \\ \hline P^{ndo}_k & \mathbf{P^{do}_k} \end{array}\right] \tag{C.2}$$

where $\mathbf{P^{do}_k}$ is the scalar variance coupled to sensor dynamics and $P^{\mathbf{ndo}}_{\mathbf{k}}$ is the vector 1×4 with the covariances between sensor estimation and observable parameters. $\mathbf{P^w_k}$ is a positive-semidefinite matrix, so is $\mathbf{P^{wo}_k}$. K_k is reordered to contain the unobservable part $K^u_k = 0$ and the observable part K^o_k, and the latter in the parameters related K^θ_k and dynamics related part K^s_k

$$\mathbf{K^w_k} = \left[\begin{array}{c} K^u_k \\ \hline K^o_k \end{array}\right] = \left[\begin{array}{c} 0 \\ \hline K^\theta_k \\ \hline K^s_k \end{array}\right] \tag{C.3}$$

and (19) is now

$$\mathbf{P_{k+1}} = \left[\begin{array}{c|c} \mathbf{I} & \mathbf{0} \\ \hline \mathbf{0} & \mathrm{I}_5 - K^o_k H^w \end{array}\right] \left[\begin{array}{c|c} \mathbf{P^u_k} + \mathbf{Q^u} & \vdots \\ \hline \cdots & \mathbf{F^w_k P^{wo}_k (F^w_k)^T} + \mathbf{Q^{wo}} \end{array}\right] \tag{C.4}$$

$$\mathbf{P_{k+1}} = \left[\begin{array}{c|c} \mathbf{P^u_k} + \mathbf{Q^u} & \vdots \\ \hline \cdots & (\mathbf{I_5} - K^o_k H^w)(\mathbf{F^w_k P^{wo}_k (F^w_k)^T} + \mathbf{Q^{wo}}) \end{array}\right] \tag{C.5}$$

which shows that SKF partitioning is also applicable for the augmented dynamic system. $\mathbf{P_k}$ is a positive-semidefinite matrix so is P^w_k and P^{wo}_k.

References

Alberer, D., & del Re, L. (2009). Fast oxygen based transient diesel engine operation. *SAE Paper 2009-01-0622*.

Chandrasekar, J., Kim, I. S., & Bernstein, D. S. (2007). Reduced-order kalman filtering for time-varying systems. In *Proceedings of the 46th IEEE conference on decision and control*.New Orleans, LA, USA, December 12–14.

Charalampidis, A. C., & Papavassilopoulos, G. P. (2011). Computationally efficient kalman filtering for a class of nonlinear systems. *IEEE Transactions on Automatic Control*, *56*(3).

Chen, X., Wang, Y., Haskara, I., & Zhu, G. (2012). Air-to-fuel ratio control with adaptive estimation of biofuel content for diesel engine LNT regeneration. In *Proceedings of the American control conference* (pp. 4957–4962). URL ⟨www.scopus.com⟩.

Desantes, J. M., Luján, J. M., Guardiola, C., & Blanco-Rodriguez, D. (2011). Development of NO$_x$ fast estimate using NO$_x$ sensors. In *EAEC 2011 Congress*. Valencia.

Ebert, C., & Jones, C. (2009). Embedded software: *facts, figures, and future*. *IEEE Computers*, *42*(4), 42–52.

Ericson, C., Westerberg, B., Andersson, M., & Egnell, R. (2006). Modelling diesel engine combustion and NO$_x$ formation for model based control and simulation of engine and exhaust aftertreatment systems. *SAE Technical Paper 2006-01-0687*. doi:http://dx.doi.org/10.4271/2006-01-0687.

Falck, T., Dreesen, P., Brabanter, K. D., Pelckmans, K., Moor, B. D., & Suykens, J. A. K. Least-squares support vector machines for the identification of Wiener-Hammerstein systems. *Control Engineering Practice*, in press.

Galindo, J., Luján, J. M., Climent, H., & Guardiola, C. (2007). Turbocharging system design of a sequentially turbocharged diesel engine by means of a wave action model. *SAE Paper 2007-01-1564*. doi:http://dx.doi.org/10.4271/2007-01-1564.

Galindo, J., Serrano, J. R., Guardiola, C., Blanco-Rodriguez, D., & Cuadrado, I. G. (2011). An on-engine method for dynamic characterisation of NO$_x$ concentration sensors. *Experimental Thermal and Fluid Science*, *35*, 470–476.

Grünbacher, E., Kefer, P., & del Re, L. (2005). Estimation of the mean value engine torque using an extended kalman filter. *SAE Technical Paper 2005-01-0063*. doi: http://dx.doi.org/10.4271/2005-01-0063.

Guardiola, C., López, J., Martín, J., & García-Sarmiento, D. (2011). Semiempirical in-cylinder pressure based model for NO$_x$ prediction oriented to control applications. *Applied Thermal Engineering*, *31*(16), 3275–3286 doi:http://dx.doi.org/10.1016/j.applthermaleng.2011.05.048.

Guardiola, C., Pla, B., Blanco-Rodriguez, D., & Cabrera, P. (2013). A learning algorithm concept for updating look-up tables for automotive applications. *Mathematical and Computer Modelling*, *57*(April (7–8)), 1979–1989 http://dx.doi.org/10.1016/j.mcm.2012.02.001.

Hirsch, M., Alberer, D., & del Re, L. (2008). Grey-box control oriented emissions models. In *Proceedings of the 17th world congress*, *the international federation of automatic control*. Seoul, Korea, July 6–11.

Höckerdal, E., Frisk, E., & Eriksson, L. (2009). Observer design and model augmentation for bias compensation with a truck engine application. *Control Engineering Practice*, *17*(3), 408–417.

Höckerdal, E., Frisk, E., & Eriksson, L. (2011). EKF-based adaptation of look-up tables with an air mass-flow sensor application. *Control Engineering Practice*, *19*, 442–453 http://dx.doi.org/10.1016/j.conengprac.2011.01.006.

HORIBA. (August 2001). *Horiba mexa-7000degr instruction manual*.

Hsieh, M.-F., & Wang, J. (2011). Design and experimental validation of an extended kalman filter-based No_x concentration estimator in selective catalytic reduction system applications. *Control Engineering Practice*, *19*(4), 346–353 doi:http://dx.doi.org/10.1016/j.conengprac.2010.12.002.

Jørgensen, J. B., Homsen, P. G., Madsen, H., & Kristensen, M. R. (2007). A computationally efficient and robust implementation of the continuous-discrete extended kalman filter. In *Proceedings of the 2007 American control conference*.

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, *82*, 35–45.

Karlsson, M., Ekholm, K., Strandh, P., Johansson, R., & Tunestal, P. (2008). LQG control for minimization of emissions in a diesel engine. In *IEEE International conference on control applications*, *2008. CCA 2008.*(pp. 245–250), September. doi:http://dx.doi.org/10.1109/CCA.2008.4629619.

Karlsson, M., Ekholm, K., Strandh, P., Tunestål, P., & Johansson, R. (2010). Dynamic mapping of diesel engine through system identification. In *Proceedings of the American control conference*. Baltimore, MD.

Kato, N., Nakagaki, K., & Ina, N. (1996). Thick film ZrO_2 NO_x sensor. *SAE Paper 960334*.

Lavoie, G. A., Heywood, J. B., & Keck, J. C. (1970). Experimental and theoretical study of nitric oxide formation in internal combustion engines. *Combustion Science and Technology*, *1*(4), 313–326, http://dx.doi.org/10.1080/00102206908952211.

Ljung, L. (1999). *System identification: Theory for the user*. Upper Saddle River, NJ: Prentice Hall PTR.

Lughofer, E., Macian, V., Guardiola, C., & Klement, E. P. (2011). Identifying static and dynamic prediction models for NO_x emissions with evolving fuzzy systems. *Applied Soft Computing*, *11*, 2487–2500.

Manchur, T. B., & Checkel, M. D. (2005). Time resolution effects on accuracy of real-time NO_x emissions measurements. *SAE Paper 2005-01-0674*. URL ⟨http://papers.sae.org/2005-01-0674/⟩.

Neeft, J. P. A., Makkee, M., & Moulijn, J. A. (1996). Diesel particulate emission control. *Fuel Processing Technology*, *47*(1), 1–69 URL ⟨ ⟨www.scopus.com⟩, cited by (since 1996): 177*www.scopus.com*⟩.

Twigg, M. V. (2007). Progress and future challenges in controlling automotive exhaust gas emissions. *Applied Catalysis B: Environmental*, *70*(1–4), 2–15 URL ⟨www.scopus.com⟩, cited by (since 1996): 217.

Wahlström, J., & Eriksson, L. (2011). Modelling diesel engines with a variable-geometry turbocharger and exhaust gas recirculation by optimization of model parameters for capturing non-linear system dynamics. In *Proceedings of the IMechE Part D: Journal of Automobile Engineering*. (Vol. 225). doi:http://dx.doi.org/10.1177/0954407011398177.

Winkler-Ebner, B., Hirsch, M., Del Re, L., Klinger, H., & Mistelberger, W. (2010). Comparison of virtual and physical NO_x-sensors for heavy duty diesel engine application. *SAE International Journal of Engines*, *3*(1), 1124–1139, http://dx.doi.org/10.4271/2010-01-1296.

Wu, G. (2006). A table update method for adaptive knock control. *SAE Paper 2006-01-0607*. ⟨http://papers.sae.org/2006-01-0607/⟩.

Yen, G., & Michel, A. N. (1991). A learning and forgetting algorithm in associative memories: Results involving pseudo inverses. In *IEEE International symposium on circuits and systems, 1991* (Vol. 2, pp. 778–781). doi:http://dx.doi.org/10.1109/ISCAS.1991.176478.

Zhuiykov, S., & Miura, N. (2007). Development of zirconia-based potentiometric NO_x sensors for automotive and energy industries in the early 21st century: *What are the prospects for sensors? Sensors and Actuators B*, *121*, 639–651.