2024 European Control Conference (ECC)
June 25-28, 2024. Stockholm, Sweden

Convergence in Delayed Recursive Identification of Nonlinear Systems

Torbjörn Wigren

Abstract— **Early detection of delay attacks on feedback control systems can be achieved by recursive identification of delay and dynamics. The paper contributes with an analysis of the convergence of a multiple-model based algorithm for joint recursive identification of fractional delay and continuous time nonlinear state space dynamics. It is proved that the true parameter vector is in the set of global convergence points, while reasons are given why a standard local stability analysis fails. A numerical example illustrates these results.**

I. Introduction

Delay plays the role as a main enemy of feedback in a number of applications, among them feedback control systems running over wireless interfaces [1]. Delay injection can also be used by adversaries to launch attacks on feedback control systems as noted e.g. in [10]. In case the dynamics is known such attacks can be detected by estimation of the delay [11]. In case the dynamics is unknown, recursive identification of the delay together with the system dynamics that would otherwise impair the delay identification is needed. This application of recursive identification to delay attack detection [26] has given rise to questions regarding the convergence properties of the algorithm of [23]. The main contribution of the paper provides answers to these questions.

Recursive identification of delay and system dynamics is well established for linear systems, see e.g. [3] that showed that output error methodology has performance advantages for these systems. If needed, fractional delays can be handled by exploiting zero order hold (ZoH) sampling zeros [2]. The above methods remain applicable for the Hammerstein and Wiener models that contain cascaded linear dynamic and static nonlinear blocks [17], [20]. For general nonlinear models simultaneous recursive identification of delay and dynamics becomes more intricate. Integer delays can be handled by delay line chains augmented to nonlinear discrete time models like the NARMAX model [6]. However, a majority of the systematic methods for nonlinear controller design are formulated in a continuous time setting, c.f. [5], [7], [9], and [15]. It therefore makes sense to consider joint recursive identification of delay and nonlinear dynamics based on continuous time models combined with Euler sampling schemes. Recursive sequential Monte-Carlo (SMC) methods like the bootstrap particle filter then constitute interesting alternatives, see [21]. Many SMC methods are not recursive though, due to inherent smoothing steps, [21]. The algorithm of [23] is however based on multiple nonlinear time shifted continuous time state space models, that are interpolated to allow recursive fractional delay identification after Euler sampling. Note that linear state space models with delay is an important special case of [23], cf. [26].

Torbjö Wigren is with the Department of Information Technology, Uppsala University, Box 337, SE-75105 Uppsala, Sweden. torbjorn.wigren @it.uu.se.

The convergence analysis that forms the main contribution of the paper exploits an analysis of associated ordinary differential equations (ODEs), as developed in [12], [13]. Previous work, like the theses [4] and [19] supervised by the author, analysed the convergence for single model nonlinear dynamics without delay, while [25] generalized [4] and [19] to include a nonlinear output. Because of the new delay model and a subtle approximation in the gradient computation of [23], the convergence results of the present paper differ considerably from [4], [19] and [25]. As before it is proved that the true parameter vector is in the set of stationary points, towards which [23] converges globally. However, the analysis of local convergence along the path of [4], [19] and [25] fails. Therefore, a numerical example of a delay attack on a nonlinear automotive cruise control system, [16], is used to show that the stability of the true parameter vector is not an issue in general. This application is further treated in [27]. Another difference as compared to [4], [19] and [25] is that the scaling of [22] is included in the analysis.

The nonlinear model and the algorithm of [23] are reviewed in Sections II and III, to enable a self contained analysis of local and global convergence in Section IV. The convergence properties are illustrated numerically in Section V. Conclusions appear in Section VI.

II. Model and Gradient

A. Fractional time delay interpolation

The fractional delay model is based on interpolation between adjacent states of multiple state and gradient ODEs, each time shifted one sampling period. The following assumptions relating to the delay are therefore introduced, cf. [23]:

- D1) The plant is time invariant and exponentially stable.
- D2) The delay T is constant.

Because of D1, the initial conditions can be disregarded when the time invariance of D1 and D2 is exploited. Secondly, time invariance implies that it does not matter if the single delay appears at the input or at the output. In [23] the delay is selected to affect the output signal model.

The inputs $\mathbf{u}_m(t)$, state vectors $\hat{\mathbf{x}}_m(t,\boldsymbol{\theta}_S)$, state gradients $\boldsymbol{\Psi}_{S,m}(t,\boldsymbol{\theta}_S)$ and delay gradients $\boldsymbol{\psi}_{T,m}(t,\theta_T,\boldsymbol{\theta}_S)$, $m = 0,...,M$, needed for identification of the delay are

$$\mathbf{u}_m(t) = \left(\mathbf{u}_{1,0}^T(t-mT_S) \;\; ... \;\; \mathbf{u}_{K,0}^T(t-mT_S)\right)^T,$$

$$k = 1, ..., K, \tag{1}$$

$$\mathbf{u}_{k,0}(t - mT_S) = \left(u_{k,0}(t - mT_S) \quad ... \quad u_{k,0}^{(n_k)}(t - mT_S) \right)^T, \tag{2}$$

$$\hat{\mathbf{x}}_m(t, \boldsymbol{\theta}_S) = \left(\hat{x}_{1,0}(t - mT_S, \boldsymbol{\theta}_S) \quad ... \quad \hat{x}_{n,0}(t - mT_S, \boldsymbol{\theta}_S) \right)^T, \tag{3}$$

$$\boldsymbol{\Psi}_{S,m}(t, \boldsymbol{\theta}_S) = \boldsymbol{\Psi}_{S,0}(t - mT_S, \boldsymbol{\theta}_S), \tag{4}$$

$$\boldsymbol{\psi}_{T,m}(t, \theta_T, \boldsymbol{\theta}_S) = \boldsymbol{\psi}_{T,0}(t - mT_S, \boldsymbol{\theta}_S). \tag{5}$$

The subscript $_m$ thus denotes the index of the $m : th$ multiple model. The dimensions of the input (disregarding derivatives), state, output and parameter vectors are K, n, ℓ and $d+1$. The superscript $^{()}$ denotes differentiation multiple times and T_S is the sampling period. $\boldsymbol{\theta}_S$ is the parameter vector of the ODE defined in section II.B, and θ_T is the delay parameter. The total parameter vector, that is common for all multiple models, is

$$\boldsymbol{\theta} = \begin{pmatrix} \theta_T & \boldsymbol{\theta}_S^T \end{pmatrix}^T. \tag{6}$$

To briefly outline the computation of the states and gradients, note that because of D1 and D2, the state vectors (3) and the corresponding state gradients (4) can be generated by solving the state and state gradient ODEs reviewed in subsections II.B and II.C, for integer time delay $m = 0$. The remaining states and state gradients are then obtained from previously stored $\hat{\mathbf{x}}_0(t, \boldsymbol{\theta}_S)$ and $\boldsymbol{\Psi}_0(t, \boldsymbol{\theta}_S)$. Asymptotically, when $\mu(t)/t \to 0$ in the algorithm (31) this holds exactly because of D1 and D2, which reduces the computational complexity with a factor of M.

The delay parameter θ_T is related to the state and output signals via a linear interpolation between the states and gradients that are adjacent in the time delay domain, thereby driving the adaptation of (31) exploiting (29). θ_T is partitioned in integer and fractional delay as follows

$$\theta_T = T = mT_S + T_f, \quad m \in [0, M-1]. \tag{7}$$

Here mT_S is the integer part of the delay while m is the number of sampling periods. The maximum delay is MT_S. The fractional delay is denoted T_f, and it fulfils

$$0 \leq T_f \leq T_S. \tag{8}$$

The linearly interpolated state vector, in between the $m : th$ and $m+1 : th$ state vectors of (3), becomes

$$\hat{\mathbf{x}}(t - \theta_T, \boldsymbol{\theta}_S) = \left(1 - \frac{T_f}{T_S}\right) \hat{\mathbf{x}}_m(t, \boldsymbol{\theta}_S) + \frac{T_f}{T_S} \hat{\mathbf{x}}_{m+1}(t, \boldsymbol{\theta}_S). \tag{9}$$

Note that interpolated quantities are not marked with any subscript. The restriction (7) will keep the estimate interior to the delay range of the multiple models. A differentiation of (9) with respect to $\boldsymbol{\theta}_S$ results in the gradient interpolation

$$\boldsymbol{\Psi}_S(t - \theta_T, \boldsymbol{\theta}_S) = \left(1 - \frac{T_f}{T_S}\right) \boldsymbol{\Psi}_{S,m}(t, \boldsymbol{\theta}_S) + \frac{T_f}{T_S} \boldsymbol{\Psi}_{S,m+1}(t, \boldsymbol{\theta}_S). \tag{10}$$

A similar treatment to derive the gradient $\boldsymbol{\psi}_{T,m}(t, \theta_T, \boldsymbol{\theta}_S)$ of (5) appears complicated since it is not clear how the gradient should be formed with respect to the integer numbering of the multiple models. A direct differentiation of (9) would also destroy the interpolation structure of (9) via T_f. For these reasons, $\boldsymbol{\psi}_{T,m}(t, \theta_T, \boldsymbol{\theta}_S)$ is *approximated* here and in [23], with the same interpolation as in (9), i.e.

$$\boldsymbol{\psi}_T(t - \theta_T, \boldsymbol{\theta}_S) = \left(1 - \frac{T_f}{T_S}\right) \boldsymbol{\psi}_{T,m}(t, \boldsymbol{\theta}_S) + \frac{T_f}{T_S} \boldsymbol{\psi}_{T,m+1}(t, \boldsymbol{\theta}_S), \tag{11}$$

The algorithm of [23] is therefore an *approximate* recursive prediction error method (RPEM). The interpolated predicted output signal is finally defined by the output matrix $\mathbf{C}$ as

$$\hat{\mathbf{y}}(t, \theta_T, \boldsymbol{\theta}_S) = \hat{\mathbf{y}}(t - \theta_T, \boldsymbol{\theta}_S) = \mathbf{C}\hat{\mathbf{x}}(t - \theta_T, \boldsymbol{\theta}_S). \tag{12}$$

B. Multiple state space models

To avoid overparameterization, the state coordinates of the ODE model are defined in a canonical manner in [23], with one parameterized nonlinear right hand side state component that is processed by a chain of integrators. Since it is enough to generate the first of the $M+1$ states and store time shifted copies, the state space model of main interest is

$$\dot{\hat{\mathbf{x}}}_0(t, \boldsymbol{\theta}_S) = \begin{pmatrix} \dot{\hat{x}}_{1,0}(t, \boldsymbol{\theta}_S) \\ \vdots \\ \dot{\hat{x}}_{n-1,0}(t, \boldsymbol{\theta}_S) \\ \dot{\hat{x}}_{n,0}(t, \boldsymbol{\theta}_S) \end{pmatrix} = \begin{pmatrix} \hat{x}_{2,0}(t, \boldsymbol{\theta}_S) \\ \vdots \\ \hat{x}_{n,0}(t, \boldsymbol{\theta}_S) \\ f\left(\hat{\mathbf{x}}_0(t, \boldsymbol{\theta}_S), \mathbf{u}_0(t), \boldsymbol{\theta}_S\right) \end{pmatrix} \tag{13}$$

$$\hat{\mathbf{y}}_0(t, \theta_T, \boldsymbol{\theta}_S) = \mathbf{C}\hat{\mathbf{x}}_0(t - \theta_T, \boldsymbol{\theta}_S), \tag{14}$$

where $\hat{\mathbf{y}}_0(t, \theta_T, \boldsymbol{\theta}_S)$ is the model output for $m = 0$.

The Stone-Weierstrass theorem, [18], proves that multipolynomials have universal approximation properties. For this reason the parameterization of (13) is selected to be polynomial, and given by

$$f(\hat{\mathbf{x}}_0(t, \boldsymbol{\theta}_S), \mathbf{u}_0(t), \boldsymbol{\theta}_S) = \boldsymbol{\varphi}^T(\hat{\mathbf{x}}_0(t, \boldsymbol{\theta}_S), \mathbf{u}_0(t))\boldsymbol{\theta}_S, \tag{15}$$

$$\boldsymbol{\theta}_S^T = \Big(\theta_{S,0...0} \quad ... \quad \theta_{S,0...I_{u_K^{(n_K)}}} \quad ... \quad \theta_{S,0...I_{u_K^{(n_K-1)}} I_{u_K^{(n_K)}}} \quad ... \quad \theta_{S,I_{x_1}...I_{u_K^{(n_K)}}} \Big), \tag{16}$$

$$\boldsymbol{\varphi}^T(\hat{\mathbf{x}}_0(t, \boldsymbol{\theta}_S), \mathbf{u}_0(t)) = \Big(1 \quad ... \quad \left(u_{K,0}^{(n_K)}(t)\right)^{I_{u_K^{(n_K)}}} \quad ... \quad \Big((\hat{x}_{1,0}(t, \boldsymbol{\theta}_S))^{I_{x_1}} \quad ... \quad (\hat{x}_{n,0}(t, \boldsymbol{\theta}_S))^{I_{x_n}} \, (u_{1,0}(t))^{I_{u_1}}$$

$$\ldots \quad \left(u_{K,0}^{(n_K)}(t)\right)^{I_{u_K^{(n_K)}}}\Bigg)\Bigg). \tag{17}$$

Here I_m denotes a maximum degree and the regression vector component 1 corresponds to $\theta_{S,0\ldots0}$. The vectors (16) and (17) are filled from left to right when the indices vary, where the rightmost index varies the fastest.

The notation is admittedly complicated, however the use of more mathematically compact alternatives like Kronecker powers and lexical ordering [8] would limit accessability for the wide engineering audience.

C. Gradients

The continuous time gradient $\boldsymbol{\psi}(t,\theta_T,\boldsymbol{\theta}_S)$ follows from the interpolated output (12), the interpolated state and delay gradients (10) and (11), and from (6). The gradient is

$$\begin{aligned}\boldsymbol{\psi}(t,\theta_T,\boldsymbol{\theta}_S) &= \boldsymbol{\psi}(t,\boldsymbol{\theta}) = \begin{pmatrix}\boldsymbol{\psi}_T^T(t,\boldsymbol{\theta}) & \boldsymbol{\psi}_S^T(t,\boldsymbol{\theta})\end{pmatrix}^T \\ &= \begin{pmatrix}\boldsymbol{\psi}_T^T(t-\theta_T,\boldsymbol{\theta}_S) & \boldsymbol{\psi}_S^T(t-\theta_T,\boldsymbol{\theta}_S)\end{pmatrix}^T \\ &= \begin{pmatrix}\frac{\partial\hat{\mathbf{y}}(t-\theta_T,\boldsymbol{\theta}_S)}{\partial\theta_T} & \frac{\partial\hat{\mathbf{y}}(t-\theta_T,\boldsymbol{\theta}_S)}{\partial\boldsymbol{\theta}_S}\end{pmatrix}^T \\ &= \begin{pmatrix}\frac{\partial\hat{\mathbf{y}}(t,\boldsymbol{\theta})}{\partial\theta_T} & \frac{\partial\hat{\mathbf{y}}(t,\boldsymbol{\theta})}{\partial\boldsymbol{\theta}_S}\end{pmatrix}^T = \begin{pmatrix}\frac{\partial\hat{\mathbf{y}}(t,\boldsymbol{\theta})}{\partial\boldsymbol{\theta}}\end{pmatrix}^T. \end{aligned} \tag{18}$$

A differentiation of (12) gives

$$\boldsymbol{\psi}_S(t,\boldsymbol{\theta}) = \mathbf{C}\left(\frac{\partial\hat{\mathbf{x}}(t,\theta)}{\partial\boldsymbol{\theta}_S}\right)^T = \mathbf{C}\boldsymbol{\Psi}_S(t,\boldsymbol{\theta}). \tag{19}$$

$\boldsymbol{\Psi}_S(t,\boldsymbol{\theta})$ is the interpolated matrix state space gradient.

The next step is to compute $\boldsymbol{\Psi}_{S,0}(t,\boldsymbol{\theta})$. The remaining matrix state space gradients are generated from $\boldsymbol{\Psi}_{S,m}(t,\boldsymbol{\theta})$ by shifting previously stored values. The computation of $\boldsymbol{\Psi}_{S,0}(t,\boldsymbol{\theta})$ is done by a differentiation of the ODE (13), after which the resulting matrix ODE is integrated using the discretization of subsection III.B. It follows from (13) that

$$\dot{\boldsymbol{\Psi}}_{S,0}(t-\theta_T,\boldsymbol{\theta}_S) = \begin{pmatrix}\frac{\partial\hat{x}_{2,0}(t-\theta_T,\boldsymbol{\theta}_S)}{\partial\boldsymbol{\theta}_S}\\ \vdots \\ \frac{\partial\hat{x}_{n,0}(t-\theta_T,\boldsymbol{\theta}_S)}{\partial\boldsymbol{\theta}_S} \\ \frac{\partial f(\hat{\mathbf{x}}_0(t-\theta_T,\boldsymbol{\theta}_S),\mathbf{u}_0(t-\theta_T),\boldsymbol{\theta}_S)}{\partial\boldsymbol{\theta}_S}\end{pmatrix}. \tag{20}$$

The bottom row components of the right hand side of (20) need to be computed for the parametrization of $f(\cdot,\cdot,\cdot)$ given by (15), (16) and (17). The result is

$$\begin{aligned}&\frac{\partial f\left(\hat{\mathbf{x}}_0(t-\theta_T,\boldsymbol{\theta}_S),\mathbf{u}_0(t-\theta_T),\boldsymbol{\theta}_S\right)}{\partial\boldsymbol{\theta}_S} \\ &= \boldsymbol{\varphi}^T(\hat{\mathbf{x}}_0(t-\theta_T,\boldsymbol{\theta}_S),\mathbf{u}_0(t-\theta_T)) + \boldsymbol{\theta}_S^T \\ &\times \begin{pmatrix}\left(\frac{\partial\boldsymbol{\varphi}(\hat{\mathbf{x}}_0(t-\theta_T,\boldsymbol{\theta}_S),\mathbf{u}_0(t-\theta_T))}{\partial\hat{x}_{1,0}(t-\theta_T,\boldsymbol{\theta}_S)}\right)^T \\ \vdots \\ \left(\frac{\partial\boldsymbol{\varphi}(\hat{\mathbf{x}}_0(t-\theta_T,\boldsymbol{\theta}_S),\mathbf{u}_0(t-\theta_T))}{\partial\hat{x}_{n,0}(t-\theta_T,\boldsymbol{\theta}_S)}\right)^T\end{pmatrix}^T \frac{\partial\hat{\mathbf{x}}_0(t-\theta_T,\boldsymbol{\theta}_S)}{\partial\boldsymbol{\theta}_S},\end{aligned} \tag{21}$$

$$\begin{aligned}&\left(\frac{\partial\boldsymbol{\varphi}(\hat{\mathbf{x}}_0(t-\theta_T,\boldsymbol{\theta}_S),\mathbf{u}_0(t-\theta_T))}{\partial\hat{x}_{i,0}(t-\theta_T,\boldsymbol{\theta}_S)}\right)^T = \Big(\mathbf{0}^T \quad 1 \quad u_{K,0}^{(n_K)}(t) \\ &2\hat{x}_{i,0}(t-\theta_T,\boldsymbol{\theta}_S) \quad 2\hat{x}_{i,0}(t-\theta_T,\boldsymbol{\theta}_S)u_{K,0}^{(n_K)}(t) \quad \ldots\Big).\end{aligned} \tag{22}$$

Finally the gradient component $\boldsymbol{\psi}_{T,0}$ of (11) is computed. It is the key to enable recursive identification of the delay. An introduction of $\bar{\tau} = t-\theta_T$, and then using (14) leads to

$$\begin{aligned}\boldsymbol{\psi}_{T,0}(t-\theta_T,\boldsymbol{\theta}_S) &= \frac{\partial\hat{\mathbf{y}}_0(t-\theta_T,\boldsymbol{\theta}_S)}{\partial\theta_T} = \frac{\partial\hat{\mathbf{y}}_0(\bar{\tau},\boldsymbol{\theta}_S)}{\partial\bar{\tau}}\frac{\partial\bar{\tau}}{\theta_T} \\ &= \dot{\hat{\mathbf{y}}}_0(t-\theta_T,\boldsymbol{\theta}_S)(-1) = -\mathbf{C}\dot{\hat{\mathbf{x}}}_0(t-\theta_T,\boldsymbol{\theta}_S),\end{aligned} \tag{23}$$

where $\dot{\hat{\mathbf{x}}}_0(t-\theta_T,\boldsymbol{\theta}_S)$ is given by (13) and in (31).

III. Recursive Identification Algorithm

A. Projection

As discussed in [12], [13], [14], a projection algorithm is needed to keep the running parameter estimate in the model set $D_{\mathcal{M}}^s$. In [23] the set of linearized asymptotically stable models that have a delay in the interval of (7) is used. Hence

$$\begin{aligned}D_{\mathcal{M}}^s = \Big\{&\begin{pmatrix}\theta_T^s & (\boldsymbol{\theta}^s)^T\end{pmatrix}^T \\ &| \ |eig(\mathbf{S}^s(\boldsymbol{\theta}^s))| < 1-\kappa, m^s \in [0, M-1]\Big\}\end{aligned} \tag{24}$$

$$\begin{aligned}&\mathbf{S}^s(\boldsymbol{\theta}^s) \\ &= \mathbf{I}_n + \alpha T_S \begin{pmatrix}0 & 1 & 0 & \ldots & 0 \\ 0 & 0 & 1 & \ddots & 0 \\ \vdots & \vdots & \ddots & \ddots & 0 \\ 0 & 0 & \ldots & 0 & 1 \\ & & (\boldsymbol{\theta}^s)^T\ \frac{d\boldsymbol{\varphi}^s(\mathbf{x}_0^s(t,\boldsymbol{\theta}^s),\mathbf{u}_0)}{d\mathbf{x}_0^s(t,\boldsymbol{\theta}^s)} & & \end{pmatrix}.\end{aligned} \tag{25}$$

where $\kappa > 0$ is small. The projection algorithm of [23] prevents whole update steps of (31) that end up outside $D_{\mathcal{M}}^s$. The superscript ^{s} is used to indicate the scaling of the sampling period introduced by (26) in the next subsection.

B. Scaling and discretization

The scaling of the sampling period is applied to improve the numerical properties when discretization is performed, see Theorem 1 and 3 of [22]. This scaling is only applied for the discretization of the ODEs, and it is given by

$$T_S^s = \alpha T_S, \tag{26}$$

The parameter vector $\boldsymbol{\theta}_S$ is changed to $\boldsymbol{\theta}_S^s$ because of the scaling. However, Theorem 2 of [22] states a linear relation between $\boldsymbol{\theta}_S$ and $\boldsymbol{\theta}_S^s$ that is easily used to recover $\boldsymbol{\theta}_S$.

Running estimates are now introduced, marked by $\hat{}$ and an added time dependence. To define the discretization, stored states are first shifted one time step according to

$$\begin{aligned}&\hat{\mathbf{x}}_m^s(t+T_s,\hat{\boldsymbol{\theta}}_S^s(t)) \\ &= \hat{\mathbf{x}}_{m-1}^s(t,\hat{\boldsymbol{\theta}}_S^s(t-T_S)), \quad m = 1,\ldots,M.\end{aligned} \tag{27}$$

Thereafter $\hat{\mathbf{x}}_0^s(t+T_s, \hat{\boldsymbol{\theta}}_S^s(t))$ is generated by the Euler forward integration method applied to (13), using (15)-(17):

$$\begin{pmatrix} \hat{x}_{1,0}^s(t+T_S, \hat{\boldsymbol{\theta}}_S^s(t)) \\ \vdots \\ \hat{x}_{n-1,0}^s(t+T_S, \hat{\boldsymbol{\theta}}_S^s(t)) \\ \hat{x}_{n,0}^s(t+T_S, \hat{\boldsymbol{\theta}}_S^s(t)) \end{pmatrix} = \begin{pmatrix} \hat{x}_{1,0}^s(t, \hat{\boldsymbol{\theta}}_S^s(t)) \\ \vdots \\ \hat{x}_{n-1,0}^s(t, \hat{\boldsymbol{\theta}}_S^s(t)) \\ \hat{x}_{n,0}^s(t, \hat{\boldsymbol{\theta}}_S^s(t)) \end{pmatrix}$$

$$+\alpha T_S \begin{pmatrix} \hat{x}_{2,0}^s(t, \hat{\boldsymbol{\theta}}_S^s(t)) \\ \vdots \\ \hat{x}_{n,0}^s(t, \hat{\boldsymbol{\theta}}_S^s(t)) \\ (\boldsymbol{\varphi}^s(\hat{\mathbf{x}}_0^s(t, \hat{\boldsymbol{\theta}}_S^s(t)), \mathbf{u}_0(t))^T \hat{\boldsymbol{\theta}}_S^s(t) \end{pmatrix}. \quad (28)$$

The discretization and scaling is identical for the gradient equations (20)-(22) and (23), see (31) for all details.

C. Search method and algorithm

The scaled algorithm of [23] follows by application of the Gauss-Newton algorithm of [14], for minimization of

$$V(\boldsymbol{\theta}^s, \boldsymbol{\Lambda}^s) = \frac{1}{2} \lim_{t\to\infty} E[(\boldsymbol{\varepsilon}^s(t, \boldsymbol{\theta}^s))^T (\boldsymbol{\Lambda}^s(t, \boldsymbol{\theta}^s))^{-1} \boldsymbol{\varepsilon}^s(t, \boldsymbol{\theta}^s) + \ln(\det(\boldsymbol{\Lambda}^s(t, \boldsymbol{\theta}^s)))]. \quad (29)$$

$\boldsymbol{\Lambda}^s(t, \boldsymbol{\theta}^s)$ is the covariance matrix of the prediction error

$$\boldsymbol{\varepsilon}^s(t, \boldsymbol{\theta}^s) = \mathbf{y}(t) - \hat{\mathbf{y}}^s(t, \theta_T, \boldsymbol{\theta}_S^s), \quad (30)$$

where $\mathbf{y}(t)$ is the measured output signal. $\boldsymbol{\Lambda}^s(t, \boldsymbol{\theta}^s)) > \mathbf{0}$ provided that an assumption like S3 below holds.

The approximate RPEM is then given by

$$\mu(t) = \frac{t}{t + \mu_1 T_S} \bar{\mu}(t)$$

$$\bar{\mu}(t + T_S) = \mu_0 \bar{\mu}(t) + 1 - \mu_0$$

$$\boldsymbol{\varepsilon}^s(t) = \mathbf{y}(t) - \hat{\mathbf{y}}^s(t)$$

$$\boldsymbol{\Lambda}^s(t) = [\boldsymbol{\Lambda}^s(t - T_S) + \frac{\mu(t)}{t} \left(\boldsymbol{\varepsilon}^s(t)(\boldsymbol{\varepsilon}^s(t))^T - \boldsymbol{\Lambda}^s(t - T_S)\right)\Big]_{D_{\mathcal{M}}^s}$$

$$\mathbf{R}^s(t) = [\mathbf{R}^s(t - T_S) + \frac{\mu(t)}{t} \left(\boldsymbol{\psi}^s(t)(\boldsymbol{\Lambda}^s(t))^{-1}(\boldsymbol{\psi}^s(t))^T - \mathbf{R}^s(t - T_S)\right)\Big]_{D_{\mathcal{M}}^s}$$

$$\begin{pmatrix} \hat{\theta}_T^s(t) \\ \hat{\boldsymbol{\theta}}_S^s(t) \end{pmatrix} = \Big[\begin{pmatrix} \hat{\theta}_T^s(t - T_S) \\ \hat{\boldsymbol{\theta}}_S^s(t - T_S) \end{pmatrix} + \frac{\mu(t)}{t} (\mathbf{R}^s(t))^{-1} \boldsymbol{\psi}^s(t) (\boldsymbol{\Lambda}^s(t))^{-1} \boldsymbol{\varepsilon}^s(t) \Big]_{D_{\mathcal{M}}^s}$$

$$\hat{m}^s(t) = \left\lfloor \frac{\hat{\theta}_T^s(t)}{T_S} \right\rfloor$$

$$\hat{T}_f^s(t) = \hat{\theta}_T^s(t) - T_S \hat{m}^s(t)$$

$$\hat{\mathbf{x}}_m^s(t + T_s) = \hat{\mathbf{x}}_{m-1}^s(t), \quad m = 1, ..., M$$

$$\boldsymbol{\varphi}^s(t) = \big(1 \quad ... \big((\hat{x}_{1,0}^s(t))^{I_{x_1}} \quad ... \quad (\hat{x}_{n,0}^s(t))^{I_{x_n}} u_1^{I_{u_1}}(t) \quad ... \quad \left(u_K^{(n_K)}(t)\right)^{I_{u_K^{(n_K)}}}\big)\big)^T$$

$$\hat{\mathbf{x}}_0^s(t + T_s) = \left(\begin{pmatrix} \hat{x}_{1,0}^s(t) \\ \vdots \\ \hat{x}_{n-1,0}^s(t) \\ \hat{x}_{n,0}^s(t) \end{pmatrix} + \alpha T_S \begin{pmatrix} \hat{x}_{2,0}^s(t) \\ \vdots \\ \hat{x}_{n,0}^s(t) \\ (\boldsymbol{\varphi}^s(t))^T \hat{\boldsymbol{\theta}}_S^s(t) \end{pmatrix} \right)_{\text{sat}}$$

$$\hat{\mathbf{x}}^s(t + T_S - T_S \hat{m}^s(t) - \hat{T}_f^s(t)) = \left(1 - \frac{\hat{T}_f^s(t)}{T_S}\right) \hat{\mathbf{x}}_{\hat{m}^s(t)}^s(t + T_S) + \frac{\hat{T}_f^s(t)}{T_S} \hat{\mathbf{x}}_{\hat{m}^s(t)+1}^s(t + T_S)$$

$$\hat{\mathbf{y}}^s(t + T_S) = \mathbf{C} \hat{\mathbf{x}}^s(t + T_S - T_S \hat{m}^s(t) - \hat{T}_f^s(t))$$

$$\boldsymbol{\Psi}_{S,m}^s(t + T_s) = \boldsymbol{\Psi}_{S,m-1}^s(t), \quad m = 1, ..., M$$

$$\frac{\partial \boldsymbol{\varphi}^s(t)}{\partial \hat{x}_{i,0}^s} = \left(\mathbf{0}^T \quad ... \quad 2\hat{x}_{i,0}^s(t) u_K^{(n_K)}(t) \quad ...\right)^T,$$

$$\frac{\partial \boldsymbol{\varphi}^s(t)}{\partial \hat{\mathbf{x}}_0^s} = \begin{pmatrix} \frac{\partial \boldsymbol{\varphi}^s(t)}{\partial \hat{x}_{1,0}^s} & ... & \frac{\partial \boldsymbol{\varphi}^s(t)}{\partial \hat{x}_{i,0}^s} & ... & \frac{\partial \boldsymbol{\varphi}^s(t)}{\partial \hat{x}_{n,0}^s} \end{pmatrix},$$

$$\boldsymbol{\Psi}_{S,0}^s(t + T_S) = \left(\begin{pmatrix} \frac{\partial \hat{x}_{1,0}^s(t)}{\partial \boldsymbol{\theta}_S^s} \\ \vdots \\ \frac{\partial \hat{x}_{n-1,0}^s(t)}{\partial \boldsymbol{\theta}_S^s} \\ \frac{\partial \hat{x}_{n,0}^s(t)}{\partial \boldsymbol{\theta}_S^s} \end{pmatrix} + \alpha T_S \begin{pmatrix} \frac{\partial \hat{x}_{2,0}^s(t)}{\partial \boldsymbol{\theta}_S^s} \\ \vdots \\ \frac{\partial \hat{x}_{n,0}^s(t)}{\partial \boldsymbol{\theta}_S^s} \\ (\boldsymbol{\varphi}^s(t))^T + (\hat{\boldsymbol{\theta}}_S^s(t))^T \left(\frac{\partial \boldsymbol{\varphi}^s(t)}{\partial \hat{\mathbf{x}}_0^s}\right) \frac{\partial \hat{\mathbf{x}}_0^s(t)}{\partial \boldsymbol{\theta}_S^s} \end{pmatrix} \right)_{\text{sat}}$$

$$\boldsymbol{\Psi}_S^s(t + T_S - T_S \hat{m}^s(t) - \hat{T}_f^s(t)) = \left(1 - \frac{\hat{T}_f^s(t)}{T_S}\right) \boldsymbol{\Psi}_{S,\hat{m}^s(t)}^s(t + T_S)$$

$$+\frac{\hat{T}_f^s(t)}{T_S}\boldsymbol{\Psi}^s_{S,\hat{m}^s(t)+1}(t+T_S)$$

$$\boldsymbol{\psi}^s_S(t+T_S-T_S\hat{m}^s(t)-\hat{T}^s_f(t))$$

$$=\mathbf{C}\boldsymbol{\Psi}^s_S(t+T_S-T_S\hat{m}(t)-\hat{T}_f(t))$$

$$\boldsymbol{\psi}^s_{T,m}(t+T_s)=\boldsymbol{\psi}^s_{T,m-1}(t),\quad m=1,...,M$$

$$\boldsymbol{\psi}^s_{T,0}(t+T_S)=-\mathbf{C}\begin{pmatrix}\hat{x}^s_{2,0}(t)\\ \vdots\\ \hat{x}^s_{n,0}(t)\\ (\boldsymbol{\varphi}^s(t))^T\hat{\boldsymbol{\theta}}^s_S(t)\end{pmatrix}$$

$$\boldsymbol{\psi}_T(t+T_S-T_S\hat{m}^s(t)-\hat{T}^s_f(t))$$

$$=\left(1-\frac{\hat{T}^s_f(t)}{T_S}\right)\boldsymbol{\psi}^s_{T,\hat{m}^s(t)}(t+T_S)$$

$$+\frac{\hat{T}^s_f(t)}{T_S}\boldsymbol{\psi}^s_{T,\hat{m}^s(t)+1}(t+T_S)$$

$$\boldsymbol{\psi}^s(t+T_S)=\begin{pmatrix}\boldsymbol{\psi}^s_T(t+T_S-T_S\hat{m}(t)-\hat{T}_f(t))\\ \boldsymbol{\psi}^s_S(t+T_S-T_S\hat{m}(t)-\hat{T}_f(t))\end{pmatrix}. \quad (31)$$

In (31), $\mu(t)/t$ denotes the gain sequence, μ_0 and μ_1 are constants for tuning, $\mathbf{R}^s(t)$ denotes the Hessian and $\lfloor\cdot\rfloor$ is the floor operator. The variables $\bar{\mu}(t)$, $\boldsymbol{\Lambda}^s(t)$, $\mathbf{R}^s(t)$, $\hat{\theta}^s_T(t)$, $\hat{\boldsymbol{\theta}}^s_S(t)$, $\hat{\mathbf{x}}^s_{0,m}(t)$, $\boldsymbol{\Psi}^s_{S,m}(t)$ and $\boldsymbol{\psi}^s_{T,m}(t)$ need initialization, which is discussed in [4], [19], [22], [23], and [24]. A saturation, marked by $_{\rm sat}$, has been added above to ensure boundedness in the convergence analysis. The saturation can be arbitrarily large and does not affect (31) in practice.

IV. Convergence

It is first noted that due to the shared dynamic model, parts of the analysis are similar to that of [4], [19] and [25].

A. Convergence analysis with associated ODEs

The convergence of (31) is analysed with averaging theory, using the method with associated ODEs developed by Ljung, [12], [13]. The technical report [12] defines a general recursive algorithm for identification of nonlinear dynamics, and provides conditions needed for the existence of a (vector) ODE associated with the algorithm, with the right hand side representing the average updating direction of the algorithm for a fixed parameter vector. It is then proved that global convergence of the general algorithm follows if the ODE is globally Lyapunov stable, while local convergence cannot hold in case the linearized associated ODE is not stable.

B. Regularity conditions

In addition to D1 and D2, the following regularity conditions are needed for the forthcoming analysis:

- M1: $D^s_{\mathcal{M}}$ is a compact subset of $\mathcal{R}^{d+1+(d+1)^2+\ell^2}$, such that the extended parameter vector $\boldsymbol{\theta}^s_e=\left((\boldsymbol{\theta}^s)^T\ (\mathrm{vec}(\mathbf{R}^s))^T\ (\mathrm{vec}(\boldsymbol{\Lambda}^s))^T\right)^T\in D^s_{\mathcal{M}}$ implies that the state dynamics and the state gradient dynamics of (31), as well as their derivatives with respect to $\boldsymbol{\theta}^s$ are exponentially stable and bounded.
- M2: $D^s_{\mathcal{M}}$ is a compact subset of $\mathcal{R}^{d+1+(d+1)^2+\ell^2}$, such that the extended parameter vector $\boldsymbol{\theta}^s_e\in D^s_{\mathcal{M}}$ implies that $\mathbf{R}^{\mathbf{s}}(t)\geq\delta_{\mathbf{R}}\mathbf{I}_{d+1}$, $\forall t$, some $\delta_{\mathbf{R}}>0$.
- M3: $D^s_{\mathcal{M}}$ is a compact subset of $\mathcal{R}^{d+1+(d+1)^2+\ell^2}$, such that the extended parameter vector $\boldsymbol{\theta}^s_e\in D^s_{\mathcal{M}}$ implies that $\boldsymbol{\Lambda}^s(t)\geq\delta_{\boldsymbol{\Lambda}}\mathbf{I}_\ell$, $\forall t$, some $\delta_{\boldsymbol{\Lambda}}>0$.
- M4: $\mathbf{u}_m(t)=(u_{1,0}(t-mT_S)\ ...\ u_{K,0}(t-mT_S))^T$, $m=0,..,M$, $k=1,...,K$, i.e. no input signal derivatives appear.
- M5: $\mathbf{u}_0(t)=\mathbf{C}_{\mathbf{u}}\mathbf{x}_{\bar{\mathbf{u}},0}(t)$, where $\mathbf{x}_{\bar{\mathbf{u}},0}(t)$ is generated from the independent and identically distributed (i.i.d) bounded vector of random variables $\{\bar{\mathbf{u}}(t)\}$, by asymptotically stable linear filtering.
- G1: $\lim_{t\to\infty}\mu(t)=\mu>0$, and $\mu(t)$ is non-increasing.
- A1: The data sequence $\{\mathbf{z}(t)\}=\{(\mathbf{y}^T(t)\ \ \mathbf{u}_0^T(t)\}$ is strictly stationary, and $\|\mathbf{z}(t)\|\leq C<\infty$, with probability one (w.p.1), $\forall t$.
- A2: The following limits exist for fixed $\boldsymbol{\theta}^s_e\in D^s_{\mathcal{M}}$:

$$\lim_{t\to\infty}E[\boldsymbol{\psi}^s(t,\boldsymbol{\theta}^s)(\boldsymbol{\Lambda}^s)^{-1}\boldsymbol{\varepsilon}^s(t,\boldsymbol{\theta}^s)]=\mathbf{f}(\boldsymbol{\theta}^s,\boldsymbol{\Lambda}^s)$$

$$\lim_{t\to\infty}E[\boldsymbol{\psi}^s(t,\boldsymbol{\theta}^s)(\boldsymbol{\Lambda}^s)^{-1}(\boldsymbol{\psi}^s(t,\boldsymbol{\theta}^s))^T]$$

$$=\mathbf{G}(\boldsymbol{\theta}^s,\boldsymbol{\Lambda}^s)$$

$$\lim_{t\to\infty}E[\boldsymbol{\varepsilon}^s(t,\boldsymbol{\theta}^s)(\boldsymbol{\varepsilon}^s(t,\boldsymbol{\theta}^s))^T]\quad=\quad\mathbf{J}(\boldsymbol{\theta}^s)$$

- S1: For each t, $\bar{s}$, $t\geq\bar{s}$, there exists a random vector $\mathbf{z}^0_{\bar{s}}(t)$ that belongs to the σ-algebra generated by $\mathbf{z}^t$ but independent of $\mathbf{z}^{\bar{s}}$ (for $\bar{s}=t$ take $\mathbf{z}^0_{\bar{s}}(t)=\mathbf{0}$), such that $E[\|\mathbf{z}(t)-\mathbf{z}^0_{\bar{s}}(t)\|^4]<C\lambda^{t-\bar{s}}$, $C<\infty$, $|\lambda|<1$.
- S2: The system can be described by $\mathbf{y}(t)=\mathbf{C}_{\mathbf{y}}\mathbf{x}_{\bar{\mathbf{y}}}(t-T_{\mathbf{y}})+\mathbf{w}(t)$, where $\mathbf{x}_{\bar{\mathbf{y}}}(t)$ is generated by sampling of the states of a continuously differentiable, bounded ODE that meets D1 and D2, and where the disturbance $\mathbf{w}(t)=\mathbf{C}_{\bar{\mathbf{w}}}\mathbf{x}_{\bar{\mathbf{w}}}(t)$, where $\mathbf{x}_{\bar{\mathbf{w}}}$ is generated from a sequence $\{\bar{\mathbf{w}}(t)\}$ of bounded i.i.d. random vectors, independent of $\{\mathbf{u}_0(t)\}$, by asymptotically stable linear filtering.

The regularity conditions are numbered as in [14], and $\mathrm{vec}(\mathbf{H})$ denotes a vector stacking the columns of the matrix $\mathbf{H}$. The conditions M1-M5 mean that the model is restricted to be exponentially stable with signals generated by exponentially stable filtering. In addition the system is assumed to have these properties as stated by A1, S1 and S2. This defines a stochastic framework that together with G1 and A2

imply existence of the associated ODEs and their relation to (31). The analysis holds for a correlated $\mathbf{w}(t)$ by S2.

C. Tools for global convergence analysis

The following result for (31) now holds:

Theorem 1: Assume that D1, D2, M1-M5, G1, A1, A2, S1 and S2 hold for (31). Also assume that there exists a twice differentiable positive function $V(\boldsymbol{\theta}^s, \mathbf{R}^s, \boldsymbol{\Lambda}^s)$ such that

$$\frac{d}{d\tau} V(\boldsymbol{\theta}_D^s(\tau), \mathbf{R}_D^s(\tau), \boldsymbol{\Lambda}_D^s(\tau)) \leq 0,$$

$$\left((\boldsymbol{\theta}_D^s)^T \ \ (\mathrm{vec}(\mathbf{R}_D^s))^T \ \ (\mathrm{vec}(\boldsymbol{\Lambda}_D^s))^T \right)^T \in D_{\mathcal{M}}^s \setminus \partial D_{\mathcal{M}}^s,$$

when evaluated along solutions of the associated ODEs

$$\begin{array}{rcl} \frac{d}{d\tau}\boldsymbol{\theta}_D^s(\tau) & = & \mu(\mathbf{R}_D^s(\tau))^{-1}\mathbf{f}(\boldsymbol{\theta}_D^s(\tau), \boldsymbol{\Lambda}_D^s(\tau)) \\ \frac{d}{d\tau}\mathbf{R}_D^s(\tau) & = & \mu\left(\mathbf{G}(\boldsymbol{\theta}_D^s(\tau), \boldsymbol{\Lambda}_D^s(\tau)) - \mathbf{R}_D^s(\tau)\right) \\ \frac{d}{d\tau}\boldsymbol{\Lambda}_D^s(\tau) & = & \mu\left(\mathbf{J}(\boldsymbol{\theta}_D^s(\tau)) - \boldsymbol{\Lambda}_D^s(\tau)\right) \end{array} .$$

Let

$$D_{\mathcal{C}}^s = \left\{ \left((\boldsymbol{\theta}_D^s)^T \ \ (\mathrm{vec}(\mathbf{R}_D^s))^T \ \ (\mathrm{vec}(\boldsymbol{\Lambda}_D^s)) \right)^T \right.$$

$$\left. \in D_{\mathcal{M}}^s \setminus \partial D_{\mathcal{M}}^s | \frac{d}{d\tau} V(\boldsymbol{\theta}_D^s(\tau), \mathbf{R}_D^s(\tau), \boldsymbol{\Lambda}_D^s(\tau)) = 0 \right\}.$$

Then either

$$\left(\hat{\boldsymbol{\theta}}^s(t) \ \ (\mathrm{vec}\left(\mathbf{R}^s(t)\right))^T \ \ (\mathrm{vec}(\boldsymbol{\Lambda}^s(t)))^T \right)^T \to D_{\mathcal{C}}^s$$

w.p.1 as $t \to \infty$,

or

$$\left(\hat{\boldsymbol{\theta}}^s(t) \ \ (\mathrm{vec}\left(\mathbf{R}^s(t)\right))^T \ \ (\mathrm{vec}(\boldsymbol{\Lambda}^s(t)))^T \right)^T \to \partial D_{\mathcal{M}}^s. \quad \square$$

Proof: Appears in the Appendix. $\square$

D. Global convergence to a true parameter vector

The positive criterion function (29) is now selected as the Lyapunov function of Theorem 1, as in [14]. Insertion of $\boldsymbol{\theta}_D^s(\tau)$, $\mathbf{R}_D^s(\tau)$ and $\boldsymbol{\Lambda}_D^s(\tau)$ in (29), followed by differentiation along the lines of [14] Section 4.4, leads to

$$\frac{d}{d\tau} V(\boldsymbol{\theta}_D^s(\tau), \mathbf{R}_D^s(\tau), \boldsymbol{\Lambda}_D^s(\tau))$$

$$= -\mu \mathbf{f}^T(\boldsymbol{\theta}_D^s(\tau), \boldsymbol{\Lambda}_D^s(\tau))(\mathbf{R}_D^s(\tau))^{-1}\mathbf{f}(\boldsymbol{\theta}_D^s(\tau), \boldsymbol{\Lambda}_D^s(\tau))$$

$$-\frac{1}{2}\mathrm{tr}\left((\boldsymbol{\Lambda}_D^s(\tau))^{-\frac{1}{2}} \left(\mu\left(\mathbf{J}(\boldsymbol{\theta}_D^s(\tau)) - \boldsymbol{\Lambda}_D^s(\tau)\right)\right)(\boldsymbol{\Lambda}_D^s(\tau))^{-1} \right.$$

$$\left. \times \left(\mu\left(\mathbf{J}(\boldsymbol{\theta}_D^s(\tau)) - \boldsymbol{\Lambda}_D^s(\tau)\right)\right)(\boldsymbol{\Lambda}_D^s(\tau))^{-\frac{1}{2}} \right) \leq 0, \qquad (32)$$

with equality only if

$$\mathbf{f}(\boldsymbol{\theta}_D^s, \boldsymbol{\Lambda}_D^s) = \mathbf{0}, \qquad (33)$$

$$\mathbf{J}(\boldsymbol{\theta}_D^s) - \boldsymbol{\Lambda}_D^s = \mathbf{0}, \qquad (34)$$

where the positive definiteness stated by M2 and M3 are required to obtain (32). The equations (33) and (34) define the set of stationary points of the ODEs of Theorem 1.

To proceed and study the case where the system is in the model set, the following assumption relating to the true parameter vector $\boldsymbol{\theta}_e^{s,\star}$ is required

S3: There is a parameter vector $\boldsymbol{\theta}_e^{s,\star} = \left((\boldsymbol{\theta}^{s,\star})^T \ (\mathrm{vec}\left(\mathbf{R}^{s,\star}\right))^T \ (\mathrm{vec}\left(\boldsymbol{\Lambda}^{s,\star}\right))^T \right)^T \in D_{\mathcal{M}}^s \setminus \partial D_{\mathcal{M}}^s$, with $\boldsymbol{\theta}^{s,\star} = \left(\boldsymbol{\theta}_T^{s,\star} \ \ \left(\boldsymbol{\theta}_S^{s,\star}\right)^T \right)^T$, such that the data sequence $\{\mathbf{z}(t)\}$ fulfils $\mathbf{y}(t) = \hat{\mathbf{y}}^s(t - \theta_T^{s,\star}, \boldsymbol{\theta}_S^{s,\star}) + \boldsymbol{\varepsilon}^s(t, \boldsymbol{\theta}^{s,\star})$, where $\boldsymbol{\varepsilon}^s(t, \boldsymbol{\theta}^{s,\star})$ is independent of $\mathbf{u}_0(t)$, with zero mean and covariance $\boldsymbol{\Sigma} > \mathbf{0}$.

The system is thus formally defined in the scaled discrete time domain, with the state and output defined by interpolation between multiple models. The main reason is that a definition in continuous time would introduce modeling errors invalidating S3, since there is no exact transformation between continuous and discrete time. A similar assumptions would be needed for any other model structure.

To analyse if the true parameter vector is a stationary point of the ODEs of Theorem 1, (33) and (34) are investigated for $\boldsymbol{\theta}_e^{s,\star}$. First it can be observed that S3 implies that $\boldsymbol{\varepsilon}^s(t, \boldsymbol{\theta}^{s,\star})$ and $\boldsymbol{\psi}(t, \boldsymbol{\theta}^{s,\star})$ are independent , since $\boldsymbol{\psi}(t, \boldsymbol{\theta}^{s,\star})$ is generated only from $\mathbf{u}_0(t)$. Assumptions A2 and S3 therefore give

$$\mathbf{f}(\boldsymbol{\theta}^{s,\star}, \boldsymbol{\Lambda}^{s,\star}) = \lim_{t\to\infty} E\left[\boldsymbol{\psi}^s(t, \boldsymbol{\theta}^{s,\star})(\boldsymbol{\Lambda}^{\mathbf{s},\star})^{-1}\boldsymbol{\varepsilon}^s(t, \boldsymbol{\theta}^{s,\star})\right]_{|\boldsymbol{\theta}_e^{s,\star}}$$

$$= \lim_{t\to\infty} E\left[\boldsymbol{\psi}^s(t, \boldsymbol{\theta}^{s,\star})\right]_{|\boldsymbol{\theta}_e^{s,\star}} (\boldsymbol{\Lambda}^{\mathbf{s},\star})^{-1} \lim_{t\to\infty} E\left[\boldsymbol{\varepsilon}^s(t, \boldsymbol{\theta}^{s,\star})\right]_{|\boldsymbol{\theta}_e^{s,\star}}$$

$$= \lim_{t\to\infty} E\left[\boldsymbol{\psi}^s(t, \boldsymbol{\theta}^{s,\star})\right]_{|\boldsymbol{\theta}_e^{s,\star}} (\boldsymbol{\Lambda}^{\mathbf{s},\star})^{-1}\mathbf{0} = \mathbf{0}. \qquad (35)$$

$$\mathbf{J}(\boldsymbol{\theta}^{s,\star}) = \lim_{t\to\infty} E[\boldsymbol{\varepsilon}^s(t, \boldsymbol{\theta}^{s,\star})(\boldsymbol{\varepsilon}^s(t, \boldsymbol{\theta}^{s,\star}))^T] = \boldsymbol{\Sigma}. \qquad (36)$$

This implies

Theorem 2: Assume that D1, D2, M1-M5, G1, A1, A2, S1, S2 and S3 hold. Then the true parameter vector $\boldsymbol{\theta}_e^{s,\star} = \left(\boldsymbol{\theta}_T^{s,\star} \ \ \left(\boldsymbol{\theta}_S^{s,\star}\right)^T \ \ (\mathrm{vec}\left(\mathbf{R}^{s,\star}\right))^T \ \ (\mathrm{vec}\left(\boldsymbol{\Sigma}\right))^T \right)^T \in D_{\mathcal{C}}^s$. $\square$

The convergence to $D_{\mathcal{C}}^s$ or $\partial D_{\mathcal{M}}^s$ is global by Theorem 1. Note however that $D_{\mathcal{C}}^s$ may contain more points than $\boldsymbol{\theta}_e^{s,\star}$.

E. No proof of local convergence

Perhaps surprising, no way to prove local convergence has been found. This can be explained as follows.

For RPEMs local convergence to the true parameter vector in the sense of [12], [13] can be proved by first establishing positive definiteness and thereby invertibility of $\mathbf{G}\left(\boldsymbol{\theta}^{s,\star}, \boldsymbol{\Lambda}^{s,\star}\right)$. For RPEMs it also holds that the linearized right hand side $\mathbf{f}\left(\boldsymbol{\theta}^{s,\star}, \boldsymbol{\Lambda}^{s,\star}\right)$ of A2 equals $-\mathbf{G}\left(\boldsymbol{\theta}^{s,\star}, \boldsymbol{\Lambda}^{s,\star}\right)$. Therefore the linearized right hand side of the first associated ODE of Theorem 1 becomes a scaled negative identity matrix and $\boldsymbol{\theta}_e^{s,\star}$ is asymptotically stable which implies local convergence, cf. [12], [13], [25].

The fact that (11) represents a gradient *approximation* prevents the above proof, since $\mathbf{G}\left(\boldsymbol{\theta}^{s,\star}, \boldsymbol{\Lambda}_S^{s,\star}\right)$ will no longer equal the negative linearized right hand side. It is also unclear how the needed differentiation of (9) should be carried out.

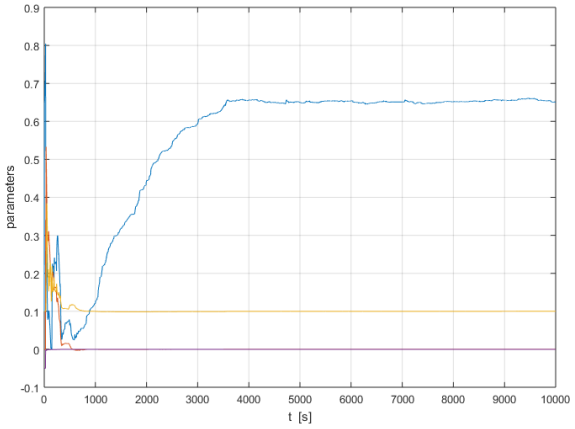

Fig. 1. Scaled parameters of (31). The blue curve represents $\hat{\theta}_T^s(t)$.

V. Numerical illustration of local convergence

To illustrate Theorem 2, and to address local convergence, (31) was applied to a nonlinear automotive cruise control system subject to delay attack [27]. The commanded force $mu_1(t)$ is counteracted by air resistance $-mk_0x_1^2(t)$, [16]. Here m is the mass, $u_1(t)$ is the accelerator control signal, $x_1(t)$ is the velocity and k_0 is the air resistance coefficient. When a delay attack occurs *before* the driver closes the loop, Newton's second law motivates the discrete time model

$$\hat{x}_1^s(t+T_S) = \hat{x}_1^s(t)$$
$$+\alpha T_S\left(\hat{\boldsymbol{\theta}}_{S,00}(t)1 + \hat{\boldsymbol{\theta}}_{S,01}(t)u_1(t) + \hat{\boldsymbol{\theta}}_{S,20}(\hat{x}_1^s(t))^2\right),$$
$$\hat{y}_1^s(t) = \hat{x}_1^s(t-\hat{\theta}_T^s), \qquad (37)$$

with $\hat{x}_1^s(t-\hat{\theta}_T^s)$ obtained from (9). The true parameters were

$$\boldsymbol{\theta}^{s,\star} = (0.65000 \quad 0.00000 \quad 1.00000 \quad -0.00120)^T, \qquad (38)$$

i.e. the delay attack used 0.65 s. Data was generated with a sampling period of 0.10 s using (37), (38), adding output measurement noise with standard deviation 0.1 m/s. The control signal $u_1(t) \in [-0.7, 1.5]$ m/s^2 switching to a new uniformly distributed random value every 10 s, resulting in $x_1(t) \in [1, 30]$ m/s. The implementation [24] of (31) was then applied with $\kappa = 0.9995$, $\bar{\mu}(0) = 1$, $\mu_1 = 300$, $\mu_0 = 0.9995$, $\hat{x}(0) = 10.0$, $\Lambda^s(0) = 0.1$, $\mathbf{R}^s(0) = \text{blockdiag}(0.1, \mathbf{I}_3)$, and $\hat{\boldsymbol{\theta}}^s(0) = (0.10000\ 0.00000\ 0.10000\ -0.05000)^T$. Re-scaling by Theorem 2 of [22] then gave

$$\hat{\boldsymbol{\theta}}(10^4) = (0.65050 \quad 0.00017 \quad 0.99979 \quad -0.00120)^T, \qquad (39)$$

at the end of run. Fig.1 and (39) then validate Theorem 2.

VI. Conclusions

In case there is a true parameter vector, i.e. a system in the model set, then the true parameter vector was proved to be in the set of global convergence points of (31). It was also explained why a proof of local convergence remains challenging. A numerical example was therefore used to assess the local convergence. A local stability analysis of (31), and refined algorithms remain open for future research.

References

[1] "Service requirements for next generation new services and markets, rev. 16.4.0", 3GPP, TS 22.261, 2018.

[2] K. J. Åström, P. Hagander and J. Sternby, "Zeros of sampled systems", *Automatica*, vol. 20, no. 1, pp. 31-38, 1984.

[3] S. Björklund and L. Ljung, "A review of time-delay estimation techniques", In *Proc. 42nd IEEE Conference on Decision and Control*, Maui, Hawaii, USA, pp. 2502-2507, 2003.

[4] L. Brus, "Nonlinear identification and control with solar energy applications", Ph.D. dissertation, Department of Information Technology, Uppsala University, Uppsala, Sweden, April 25, 2008.

[5] A. E. Bryson and Y.-C. Ho, *Applied Optimal Control - Optimization, Estimation and Control*. New York, NY: Taylor and Francis, 1975.

[6] S. Chen and S. A. Billings, "Representation of nonlinear systems: the NARMAX model", *Int. J. Control*, vol. 49, pp. 1013-1032, 1989.

[7] E. Fridman, *Introduction to Time-Delay Systems: Analysis and Control*. Birkhäuser, 2014.

[8] R. Fröberg, *An Introduction to Gröbner bases.* Chichester U. K.: Wiley, 1997.

[9] H. K. Khalil, *Nonlinear Systems*. Upper Saddle River, NJ: Prentice Hall, 2002.

[10] E. Korkmaz, A. Dolgikh, M. Davis and V. Skormin, "ICS security testbed with delay attack case study", In *Military Communications Conference*, pp. 283-288, 2016.

[11] V. Léchappé, J. De Léon, E. Moulay, F. Plestan and A. Glumineau, "Delay and state observation for SISO nonlinear systems with input delay", *Int. J. Robust and Nonlinear Contr.*, pp. 2356-2368, no. 6, vol. 28, 2018.

[12] L. Ljung, "Theorems for the asymptotic analysis of recursive stochastic algorithms", Report 7522, Department of Automatic Control, Lund Institute of Technology, Lund, Sweden, 1975.

[13] L. Ljung, "Analysis of recursive stochastic algorithms", *IEEE Trans. Automat. Contr.*, vol. AC-22, pp. 551-575, 1977.

[14] L. Ljung and T. Söderström, *Theory and Practice of Recursive Identification*. Cambridge, MA: MIT Press, 1983.

[15] H. Nijmeijer and A. J. van der Schaft, *Nonlinear Dynamical Control Systems.* New York, NY: Springer, 1990.

[16] P. Nilsson, O. Hussien, A. Balkan, Y. Chen, A. D. Ames, J. W. Grizzle, N. Ozay, H. Peng and P. Tabuada, "Correct-by-construction adaptive cruise control: two approaches", *IEEE Trans. Contr. Systems Tech.*, vol. 24, no. 4, pp. 1294-1307, 2016.

[17] P. Stoica and T. Söderström, "Instrumental variable methods for Hammerstein systems", *Int. J. Contr.*, vol 35, pp. 459-476, 1982.

[18] M. H. Stone, "The generalized Weierstrass approximation theorem", *Math. Mag.*, vol. 21, pp. 148-157, 1948.

[19] S. Tayamon, "Nonlinear system identification and control applied to selective catalytic reduction systems", Ph.D dissertation, department of Information Technology, Uppsala University, Uppsala, Sweden, September, 2014.

[20] D. Westwick and M. Verhaegen, "Identifying MIMO Wiener systems using subspace model identification methods", *Signal Processing*, vol. 52, pp. 235–258, 1996.

[21] A. Wigren, M. Wågberg, F. Lindsten, A. G. Wills and T. B. Schön, "Nonlinear system identification: learning while respecting physical models using a sequential Monte Carlo method", *IEEE Contr. Systems Mag.*, vol. 42, pp. 75-102, 2022.

[22] T. Wigren, "Scaling of the sampling period in nonlinear system identification", In *Proc. American Control Conference*, Portland, Oregon, pp. 5058-5065, 2005.

[23] T. Wigren, "Networked and delayed recursive identification of nonlinear systems", In *Proc 56th IEEE Conference on Decision and Control*, Melbourne, Victoria, Australia, pp. 5851-5858, 2017.

[24] T. Wigren, "MATLAB software for nonlinear and delayed recursive identification - revision 2", Uppsala University, Uppsala, Sweden, Technical Reports from the Department of Information Technology, 2022-002, 2022. Available: https://www.it.uu.se/research/publications/reports/2022-002/RecursiveNonlinearNetworkedIdentificationSW-r3.zip

[25] T. Wigren, "Recursive identification of a nonlinear state space model", *Int. J. Adaptive Contr. Signal Processing*, vol. 37, no. 2, pp. 447-473, 2023. Available: https://onlinelibrary.wiley.com/doi/10.1002/acs.3531

[26] T. Wigren and A. Teixeira, "On-line identification of delay attacks in networked servo control", In *Prep. IFAC World Congress*, Yokohama, Japan, pp. 1041-1047, 2023.

[27] T. Wigren and A. Teixeira, "Delay attack and detection in feedback linearized control systems", In *Proc. European Control Conference*, Stockholm, Sweden, 2024.

APPENDIX

A. Outline of the proof of Theorem 1

To prove Theorem 1, it is first needed to re-write (31) as the general algorithm of [12], by mapping all variables of (31) to those of [12]. Thereafter the conditions D1, D2, M1-M5, G1, A1, A2, S1 and S2 need to be proved to imply those of the theorems of [12]. This proves Theorem 1. Towards this end, the detailed proof of convergence of the related *open access* paper [25] is used as a freely downloadable reference, to obtain a proof within the 8 page limitation of this paper.

B. Proof of Theorem 1

Appendix A of [25] reviews the general algorithm and the results proved for it in [12]. The parts that are relevant here appear in Appendix B of [25]. Variables of the general algorithm of [12] are denoted with a $\check{}$, exactly as in [25].

1) Algorithm mapping: The projection algorithm is first included in the general algorithm with an indicator function, as in equation (B1) of [25]. The dimension of the indicator function is extended here to include the delay range of (24).

The mapping of variables between (31) and the general algorithm of [12] are $\check{\mathbf{x}}(t) = \boldsymbol{\theta}_e^s(t)$, $\check{\boldsymbol{\varphi}}(t) = \boldsymbol{\eta}^s(t)$, $\gamma(t) = \frac{1}{t}$, and $\check{\mathbf{e}}(t+1) = \bar{\mathbf{z}}(t+T_s) = (\bar{\mathbf{u}}^T(t+T_S) \quad \bar{\mathbf{w}}^T(t+T_S))$, as in Lemma 4 and (B5) of [25].

The combined state vector $\boldsymbol{\eta}^s(t)$ of (31) is constructed next. Due to the shifting of (27), the state $\hat{\mathbf{x}}_0^s(t)$ is first included in the vector $\boldsymbol{\eta}^s(t)$, thereby also *including the scaling* that is not in [23], [25]. Referring to D2, the shifted states $\hat{\mathbf{x}}_m^s(t)$, $m = 1, ..., M$ are then included in $\boldsymbol{\eta}^s(t)$ by application of multiple delay lines in the general state model in [12], [25]. This procedure is repeated for the matrix gradients $\boldsymbol{\Psi}_{S,m}^s(t)$ after vectorization of the matrix states. Using M4, M5 and S2 of the present paper, $\boldsymbol{\eta}^s(t)$ is then augmented with the input state, noise state, output state, input signal and noise signal, to define the counterpart to (B4) of [25], which includes $(\hat{\mathbf{x}}_m^s(t))^T$, $(\mathrm{vec}(\boldsymbol{\Psi}_{S,m}(t)))^T$, $m = 0, ..., M$, $\mathbf{x}_{\bar{y}}^T(t)$, $\mathbf{x}_{\bar{\mathbf{u}}}^T(t)$, $\mathbf{x}_{\bar{w}}^T(t)$, $\bar{\mathbf{u}}^T(t)$, and $\bar{\mathbf{w}}^T(t)$.

The remaining parts of the construction of the right hand sides of $\mathbf{Q}(t, \check{\mathbf{x}}(t-1), \check{\boldsymbol{\varphi}}(t-1))$ and $\mathbf{g}(t; \check{\boldsymbol{\varphi}}(t-1), \check{\mathbf{x}}(t-1), \check{\mathbf{e}}(t))$ of the general algorithm are, except for the delay lines, analogous to the one of the proof of Lemma 4 of [25]. This follows since the output, states and gradients of (31) are generated by polynomial functions, both here and in [25], cf. (11), (12), (15), (16), (17), (21), (22) and (23).

This proves that Lemma 4 of [25] holds for (31).

2) Verification of regularity conditions: The regularity conditions R1-R11 of [25] on the general algorithm can now be verified for (31).

R1 follows from the saturation in the *scaled* state and state gradient recursions of (31), and since D1, M4, M5, S1 and S2 imply boundedness of the remaining states of $\boldsymbol{\eta}(t)$, [25].

R2 follows by the same arguments as in [25], since the states and gradients of the present paper and in [25] are both generated by polynomials, therefore the differentiability properties are identical. Since $\boldsymbol{\eta}^s(t)$ of the present paper is extended as compared to [25], the difference in the verification is that there is a need for $M+1$ state bounds and $M+1$ state gradient bounds in the present paper, where the steps of each bounding computation is the same as in [25]. The multiple bounding leads to the results $\left\| \frac{\partial \mathbf{Q}}{\partial \check{\mathbf{x}}} \right\| \leq C < \infty$ and $\left\| \frac{\partial \mathbf{Q}}{\partial \check{\boldsymbol{\varphi}}} \right\| \leq C < \infty$, that verify R2.

R3 follows by the same arguments as in [25], since the states and gradients of the present paper and in [25] are both generated by polynomials, and since all signals are bounded in $D_{\mathcal{M}}^s \setminus \partial D_{\mathcal{M}}^s$, by M1, M4, M5 and A2.

R4 is verified by repeated application of the mean value theorem. As in [25], it is first proved that $\mathbf{g}(t; \check{\boldsymbol{\varphi}}(t-1), \check{\mathbf{x}}(t-1), \check{\mathbf{e}}(t))$ is continuously differentiable with respect to $\check{\boldsymbol{\varphi}}(t-1)$ and $\check{\mathbf{x}}(t-1)$. It is first noted that $\boldsymbol{\eta}^s(t)$ is formed only from the state components listed in Appendix B.1 and not the predicted or system outputs, as seen by comparison with the right hand side of $\mathbf{g}(t; \check{\boldsymbol{\varphi}}(t-1), \check{\mathbf{x}}(t-1), \check{\mathbf{e}}(t))$ of Lemma 4 of [25]. Continuous differentiability therefore follows since the right hand sides of the state and matrix ODEs are polynomial, and since the remaining states are formed by asymptotically stable linear filtering, referring to M4, M5 and S2. As in [25] the corners of the saturation can be disregarded. The state equation of the general algorithm is then iterated formally for a *fixed* $\bar{\check{\mathbf{x}}}$ exploiting the mean value theorem, to obtain a closed form expression as in (B33) of [25]. This result is bounded, using the projection algorithm, M1 and M5, exactly as in (B34)-(B36) of [25]. This proves R4.

R5 is proved by again noting that $\boldsymbol{\eta}^s(t)$ is generated by iteration using the right hand side of $\mathbf{g}(t; \check{\boldsymbol{\varphi}}(t-1), \check{\mathbf{x}}(t-1), \check{\mathbf{e}}(t))$ of Lemma 4 of [25]. R5 therefore follows as in Appendix 4A of [14], cf. [25].

R6 treats the existence and details of the average updating directions, exploiting the fact that the dependence of the projection algorithm disappears for interior points of $D_{\mathcal{M}}^s$. This follows since the saturation, G1, M1, M2 and M3 imply an updating rate of (31) that tends to 0 when $t \to \infty$. For fixed $\theta_e^s \in D_{\mathcal{M}}^s \setminus \partial D_{\mathcal{M}}^s$ the indicator function will then always equal 1, $t > t_0$, $t_0 < \infty$ and R6 follows, see [25].

R7 follows as in [25] since also here $\{\bar{\mathbf{u}}(t) \quad \bar{\mathbf{w}}(t)\}$ is a set of i.i.d. random variables.

R8-R11 follow by G1 and evaluation of $\gamma(t) = 1/t$.

This proves that Lemma 5 of [25] holds for (31).

3) The boundedness condition: The boundedness condition is proved exactly as in [25] since all components of $\boldsymbol{\eta}^s(t)$ are bounded also in the present paper, cf. R1 above.

4) Concluding the proof of Theorem 1: The validity of Lemma 4 of [25] proved in Appendix B.1 for (31), shows that (31) can be written as the general algorithm of [12]. Lemma 5 of [25] proved in Appendix B.2 for (31) proves that D1, D2, M1-M5, G1, A1, A2, S1 and S2 imply that Lemma 2 of [25] holds for (31). As proved in Appendix B.3 the boundedness condition of Lemma 2 of [25] is fulfilled for (31). A comparison of the general algorithm of [12], [25] and R6, with (31) and A2 proves Theorem 1. □