Characteristic Parameter Identification by Recursive Ordinary Least Squares

Esten I. Grøtli, Mark A. Haring, Synne Fossøy

***Abstract*—Linear time invariant state-space models have several applications related to prediction and control. When models cannot be derived based on physical principles, these must be identified based on measurements of the input- and corresponding output vector. In this article we show that for the purpose of system identification it is useful to transform the state-space model into an autoregressive moving-average model with exogenous (ARMAX) inputs as this removes model redundancy related to the state vector. Our main result is a Kalman-like filter with forgetting factor that recursively estimates parameters of the ARMAX model based on new input and output measurements. We prove that the estimated parameters produce the smallest, weighted model errors. The recursive nature of the algorithm is advantageous computationally compared to batch processing if the data set is large, and moreover it can adapt to changes in model parameters online. The benefit of the algorithm is illustrated in a simulation of two stirred tanks in series.**

***Index Terms*—System identification, learning, LTI, ARMAX**

I. Introduction

A. Previous work

System identification is a major field of research and is concerned with finding dynamical models through observed data (see e.g. [6], [13] for some extensive information on the subject). We are here interested in the identification of discrete-time linear time invariant (LTI) state space models. Such models are extensively used for prediction and control applications. In [11] a number of the drawbacks of identification of state-space models are presented, and the authors propose to use transfer function matrices as an alternative parameterization. This is similar to our approach where the state space model is re-written on the form of an ARMAX-model. Identification of ARMA(X)-models is extensively treated in the literature, see for instance [12], [15], [7] or [10]. A recursive method for generalized least squares identification of an ARMAX-model was proposed in [14], whereas a Kalman-like recursive prediction algorithm was presented in [4] for identification of linear regression models with moving average process noise. In [5] a recursive predictor-based subspace identification algorithm was presented. Variable-rate forgetting factors of recursive least-squares algorithms have had a renewed interest lately, see e.g. [2], [3], [9]. Our approach has perhaps most similarities to the RPBSID$_{\text{PM}}$ algorithm of [5]. The method consists of three computational steps as data becomes available. The first step is to update the Markov parameters by recursive least squares. The second is to estimate the state vector, and the third and final step is to update the state-space parameters through computation of two more recursive least squares steps.

This work was supported by the industry partners Borregaard, Elkem, Hydro, Yara and by the Research Council of Norway (RCN) through the project TAPI: Towards Autonomy in Process Industries (RCN project number 294544). We also acknowledge the support by RCN to finalize the article through the project PERTINENCE: Physics-informed artificial intelligence for autonomy applications in defence (RCN project number 346105).

The authors' affiliations are with the Department of Mathematics and Cybernetics, SINTEF Digital, Trondheim, Norway

B. Contributions

The main result of this work is a Kalman-like filter with forgetting factor for estimating characteristic parameters of an ARMAX-model (Lemma 7), where we prove that the estimated parameters produce the smallest, weighted model errors. We illustrate how an ARMAX model can be used to remove model redundancy related to the state vector in LTI model (Lemma 1). Although this is well known (see e.g. [1], [8]), this is often not utilized in identification of systems on state space form. As already pointed out, our work has some similarities to [5], in that a recursive least-squares method with forgetting factor is used. However, contrary to their work we do not need the assumption of nilpotency of the state transition matrix, see [5, Assumption 2]. Another difference is that we estimate the errors to account for any correlations when estimating the parameters. That means that if the errors are strongly correlated in time (loosely speaking, if there is much process noise in comparison to measurement noise), their method is expected to perform worse on average.

C. Organisation

This article is organized as follows: Section II contains the model identification problem formulation. The non-uniqueness of the LTI model is removed by transforming it into an ARMAX model. Section II describes how a Kalman-filter like algorithm can be used to recursively estimate ARMAX-model parameters, and it is shown how the estimated parameters produces the smallest, weighted model errors. In Section IV a system consisting of two stirred tanks in series is simulated, both with constant and increasing flow, and the estimation algorithm is used to identify the parameters and output. Conclusion and further work is provided in Section V, and finally the proofs of the main results are found in appendices A and B.

II. Model identification problem formulation

Consider the following linear time-invariant state-space model:

$$\begin{aligned}\mathbf{x}_{k+1} &= \mathbf{f} + \mathbf{F}^{\mathbf{x}}\mathbf{x}_k + \mathbf{F}^{\mathbf{u}}\mathbf{u}_k + \mathbf{F}^{\mathbf{e}}\mathbf{e}_k \\ \mathbf{y}_k &= \mathbf{h} + \mathbf{H}^{\mathbf{x}}\mathbf{x}_k + \mathbf{H}^{\mathbf{u}}\mathbf{u}_k + \mathbf{H}^{\mathbf{e}}\mathbf{e}_k,\end{aligned} \tag{1}$$

with state vector $\mathbf{x}_k \in \mathbb{R}^{n_\mathbf{x}}$, input vector $\mathbf{u}_k \in \mathbb{R}^{n_\mathbf{u}}$, output vector $\mathbf{y}_k \in \mathbb{R}^{n_\mathbf{e}}$, and error vector $\mathbf{e}_k \in \mathbb{R}^{n_\mathbf{e}}$, where $k \in \mathbb{Z}$ is a counter for the time steps of the model, and where $n_\mathbf{x}, n_\mathbf{u} \in \mathbb{N}$ and $n_\mathbf{y}, n_\mathbf{e} \in \mathbb{N}$ are the corresponding dimensions. The main idea about introducing the bias $\mathbf{f}$ in the state equation and the bias $\mathbf{h}$ in the measurement equation is that, these constant terms are a result of linearizing a nonlinear function. Thus, if we want to approximate a nonlinear process with a linear model, we end up with model that is similar to the one described in (1). It is worth noticing that the process noise $\mathbf{F}^\mathbf{e}\mathbf{e}_k$, and measurement noise $\mathbf{H}^\mathbf{e}\mathbf{e}_k$ are allowed to be correlated, which is a slight generalization of the more common choice found in literature.

The only things we know about the model are its input $\mathbf{u}_k$ and its output $\mathbf{y}_k$, which are measured. All matrices and other vectors, including the state vector $\mathbf{x}_k$ and the error vector $\mathbf{e}_k$, are unknown. We aim to find a state vector $\mathbf{x}_k$, vectors $\mathbf{f}$ and $\mathbf{h}$, and matrices $\mathbf{F}^\mathbf{x}$, $\mathbf{F}^\mathbf{u}$, $\mathbf{F}^\mathbf{e}$, $\mathbf{H}^\mathbf{x}$, $\mathbf{H}^\mathbf{u}$ and $\mathbf{H}^\mathbf{e}$, such that the corresponding error vector $\mathbf{e}_k$ is small. That is, the model accurately describes the relation between the input vector $\mathbf{u}_k$ and the output vector $\mathbf{y}_k$. In turn, the obtained state-space model may be used for a variety of tasks related to prediction and control.

It is important to realize that any such model is not unique in the sense that there are multiple models (in fact, infinitely many) for which the corresponding model error is the same. This can be easily seen using a state similarity transform. Consider the similarity transform $\bar{\mathbf{x}}_k = \mathbf{T}\mathbf{x}_k + \mathbf{b}$, where $\mathbf{T}$ is any invertible, real matrix and $\mathbf{b}$ is any real vector. It follows that the state-space model in (1) can be written as

$$\begin{aligned} \bar{\mathbf{x}}_{k+1} &= \bar{\mathbf{f}} + \bar{\mathbf{F}}^\mathbf{x}\bar{\mathbf{x}}_k + \bar{\mathbf{F}}^\mathbf{u}\mathbf{u}_k + \bar{\mathbf{F}}^\mathbf{e}\mathbf{e}_k \\ \mathbf{y}_k &= \bar{\mathbf{h}} + \bar{\mathbf{H}}^\mathbf{x}\bar{\mathbf{x}}_k + \mathbf{H}^\mathbf{u}\mathbf{u}_k + \mathbf{H}^\mathbf{e}\mathbf{e}_k, \end{aligned} \tag{2}$$

with

$$\begin{aligned} &\bar{\mathbf{f}} = \mathbf{T}\mathbf{f} + \left(\mathbf{I} - \mathbf{T}\mathbf{F}^\mathbf{x}\mathbf{T}^{-1}\right)\mathbf{b}, \quad \bar{\mathbf{h}} = \mathbf{h} - \mathbf{H}^\mathbf{x}\mathbf{T}^{-1}\mathbf{b}, \\ &\bar{\mathbf{F}}^\mathbf{x} = \mathbf{T}\mathbf{F}^\mathbf{x}\mathbf{T}^{-1}, \ \ \bar{\mathbf{F}}^\mathbf{u} = \mathbf{T}\mathbf{F}^\mathbf{u}, \ \ \bar{\mathbf{F}}^\mathbf{e} = \mathbf{T}\mathbf{F}^\mathbf{e}, \ \ \bar{\mathbf{H}}^\mathbf{x} = \mathbf{H}^\mathbf{x}\mathbf{T}^{-1}. \end{aligned} \tag{3}$$

Hence, changing the model by changing the values of the matrix $\mathbf{T}$ and the vector $\mathbf{b}$ may change the values of the state vector $\bar{\mathbf{x}}_k$, the vectors $\bar{\mathbf{f}}$ and $\bar{\mathbf{h}}$, and the matrices $\bar{\mathbf{F}}^\mathbf{x}$, $\bar{\mathbf{F}}^\mathbf{u}$, $\bar{\mathbf{F}}^\mathbf{e}$ and $\bar{\mathbf{H}}^\mathbf{x}$, while the corresponding value of the error vector $\mathbf{e}_k$ remains unchanged.

To remove the model redundancy related to the state vector, we may transform the state-space model in (1) into an autoregressive moving-average model with exogenous inputs (ARMAX model).

Lemma 1. *For any state-space model in* (1)*, there exists a nonnegative integer $d \leq n_\mathbf{x}$, a real vector $\mathbf{g}$, and real matrices $\mathbf{G}^\mathbf{Y}$, $\mathbf{G}^\mathbf{U}$, $\mathbf{G}^\mathbf{u}$, $\mathbf{G}^\mathbf{E}$, and $\mathbf{G}^\mathbf{e}$ such that*

$$\mathbf{y}_k = \mathbf{g} + \mathbf{G}^\mathbf{Y}\mathbf{Y}_k + \mathbf{G}^\mathbf{U}\mathbf{U}_k + \mathbf{G}^\mathbf{u}\mathbf{u}_k + \mathbf{G}^\mathbf{E}\mathbf{E}_k + \mathbf{G}^\mathbf{e}\mathbf{e}_k \tag{4}$$

for all $k \in \mathbb{Z}$, with

$$\mathbf{Y}_k = \begin{bmatrix} \mathbf{y}_{k-d} \\ \mathbf{y}_{k-d+1} \\ \vdots \\ \mathbf{y}_{k-1} \end{bmatrix}, \quad \mathbf{U}_k = \begin{bmatrix} \mathbf{u}_{k-d} \\ \mathbf{u}_{k-d+1} \\ \vdots \\ \mathbf{u}_{k-1} \end{bmatrix}, \quad \mathbf{E}_k = \begin{bmatrix} \mathbf{e}_{k-d} \\ \mathbf{e}_{k-d+1} \\ \vdots \\ \mathbf{e}_{k-1} \end{bmatrix}. \tag{5}$$

Proof. See Appendix A. □

Remark 2. *Note that we can always obtain a state-space model of the form in* (1) *from the ARMAX model in* (4) *using the following realization:*

$$\begin{aligned} \mathbf{f} &= \begin{bmatrix} \mathbf{0} \\ \mathbf{g} \end{bmatrix}, & \mathbf{F}^\mathbf{x} &= \begin{bmatrix} \begin{bmatrix} \mathbf{0} & \mathbf{I} \end{bmatrix} \\ \mathbf{G}^\mathbf{Y} \end{bmatrix}, & \mathbf{h} &= \mathbf{0}, \\ \mathbf{H}^\mathbf{x} &= \begin{bmatrix} \mathbf{I} & \mathbf{0} \end{bmatrix}, & \mathbf{H}^\mathbf{u} &= \mathbf{G}^\mathbf{u}, & \mathbf{H}^\mathbf{e} &= \mathbf{G}^\mathbf{e}, \end{aligned} \tag{6}$$

and

$$\mathbf{F}^\mathbf{u} = \begin{bmatrix} \mathbf{M}_1^\mathbf{u} \\ \mathbf{M}_2^\mathbf{u} \\ \vdots \\ \mathbf{M}_d^\mathbf{u} \end{bmatrix}, \quad \mathbf{F}^\mathbf{e} = \begin{bmatrix} \mathbf{M}_1^\mathbf{e} \\ \mathbf{M}_2^\mathbf{e} \\ \vdots \\ \mathbf{M}_d^\mathbf{e} \end{bmatrix}, \tag{7}$$

where $\mathbf{M}_i^\mathbf{u}$ and $\mathbf{M}_i^\mathbf{e}$ for $i \in \{1, 2, \ldots, d\}$ can be obtained from the equations in (30), (32), *and* (36) *of Appendix A.*

Remark 3. *Note that if we are not interested in estimating the state $\mathbf{x}_k$ in* (1)*, then the relation between the input $\mathbf{u}_k$ and the output $\mathbf{y}_k$ can be perfectly described even when setting $\mathbf{h} = \mathbf{0}$, as seen from Remark 2. Strictly speaking, we do not need either of the bias terms. We can also use the same model without bias terms if we increase the degree d of the model by one. However, by increasing the degree, we will introduce many more parameters than strictly necessary, namely, the $\mathbf{G}^\mathbf{Y}$ increases with $n_\mathbf{y} \times n_\mathbf{y}$ parameters, the matrix $\mathbf{G}^\mathbf{U}$ increases with $n_\mathbf{y} \times n_\mathbf{u}$ parameters, while the matrix $\mathbf{G}^\mathbf{E}$ increases with $n_\mathbf{y} \times n_\mathbf{e}$ parameters. If we would allow a bias, we would only have $n_\mathbf{y}$ parameters instead.*

The constant d, the vector $\mathbf{g}$, and the matrices $\mathbf{G}^\mathbf{Y}$, $\mathbf{G}^\mathbf{U}$, $\mathbf{G}^\mathbf{u}$, $\mathbf{G}^\mathbf{E}$, and $\mathbf{G}^\mathbf{e}$ can be regarded as model parameters of the ARMAX model in (4). We made the following assumption related to our knowledge of the parameters d, $\mathbf{G}^\mathbf{E}$, $\mathbf{G}^\mathbf{e}$.

Assumption 4. *The constant d and the matrices $\mathbf{G}^\mathbf{E}$ and $\mathbf{G}^\mathbf{e}$ of the ARMAX model in* (4) *are known. Moreover, the values of $\mathbf{G}^\mathbf{E}$ and $\mathbf{G}^\mathbf{e}$ are such that*

$$\operatorname{rank}\begin{bmatrix} \mathbf{G}^\mathbf{E} & \mathbf{G}^\mathbf{e} \end{bmatrix} = n_\mathbf{y}. \tag{8}$$

All other model parameters are estimated using the method presented in the next section.

Remark 5. *In practice, the constant d and the matrices $\mathbf{G}^\mathbf{E}$ and $\mathbf{G}^\mathbf{e}$ are often unknown. In that case, d, $\mathbf{G}^\mathbf{E}$, $\mathbf{G}^\mathbf{e}$ can be regarded as tuning parameters, where d is a measure for the complexity of the ARMAX model (i.e., a larger value of d results in a more complex model), and where $\mathbf{G}^\mathbf{E}$ and $\mathbf{G}^\mathbf{e}$ are tuning matrices related to the error characteristics of the model (e.g., similar to the noise covariance matrices* *of a Kalman filter). For example, we may choose $\mathbf{G}^{\mathbf{E}}$ to be a zero matrix and $\mathbf{G}^{\mathbf{e}}$ to be a diagonal matrix with the standard deviation of the measurement noise of each of the output variables on its diagonal. However, note that often more accurate estimates of all other model parameters can be obtained (with the method in Section III) if more suitable values for $\mathbf{G}^{\mathbf{E}}$ and $\mathbf{G}^{\mathbf{e}}$ are chosen.*

Remark 6. *The rank condition in* (8) *of Assumption 4 implies that all outputs of the ARMAX model in* (4) *are influenced by model errors. As a result, the model is feasible for all possible values of the input vector $\mathbf{u}_k$ and the output vector $\mathbf{y}_k$.*

III. Recursive parameter estimation with forgetting

With d, $\mathbf{G}^{\mathbf{E}}$, $\mathbf{G}^{\mathbf{e}}$ being known (see Assumption 4), let us introduce the following vector that contains all remaining model parameters:

$$\boldsymbol{\xi} = \begin{bmatrix} \mathbf{g} \\ \mathrm{vec}(\mathbf{G}^{\mathbf{Y}}) \\ \mathrm{vec}(\mathbf{G}^{\mathbf{U}}) \\ \mathrm{vec}(\mathbf{G}^{\mathbf{u}}) \end{bmatrix}. \tag{9}$$

Using this parameter vector, it follows that the ARMAX model in (4) can be written as

$$\mathbf{y}_k = \mathbf{G}_k^{\boldsymbol{\xi}} \boldsymbol{\xi} + \mathbf{G}^{\mathbf{E}} \mathbf{E}_k + \mathbf{G}^{\mathbf{e}} \mathbf{e}_k, \tag{10}$$

with

$$\mathbf{G}_k^{\boldsymbol{\xi}} = \begin{bmatrix} 1 & \mathbf{Y}_k^T & \mathbf{U}_k^T & \mathbf{u}_k^T \end{bmatrix} \otimes \mathbf{I}. \tag{11}$$

Now, let us define the following "state" vector that contains both model parameters and error variables:

$$\boldsymbol{\zeta}_k = \begin{bmatrix} \boldsymbol{\xi} \\ \mathbf{E}_k \end{bmatrix}. \tag{12}$$

Note that

$$\mathbf{E}_{k+1} = \mathbf{A}^{\mathbf{E}} \mathbf{E}_k + \mathbf{A}^{\mathbf{e}} \mathbf{e}_k, \tag{13}$$

with

$$\mathbf{A}^{\mathbf{E}} = \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}, \quad \mathbf{A}^{\mathbf{e}} = \begin{bmatrix} \mathbf{0} \\ \mathbf{I} \end{bmatrix}. \tag{14}$$

Therefore, we have that

$$\begin{aligned} \boldsymbol{\zeta}_{k+1} &= \mathbf{Q}^{\boldsymbol{\zeta}} \boldsymbol{\zeta}_k + \mathbf{Q}^{\mathbf{e}} \mathbf{e}_k \\ \mathbf{y}_k &= \mathbf{G}_k^{\boldsymbol{\zeta}} \boldsymbol{\zeta}_k + \mathbf{G}^{\mathbf{e}} \mathbf{e}_k, \end{aligned} \tag{15}$$

with

$$\mathbf{Q}^{\boldsymbol{\zeta}} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}^{\mathbf{E}} \end{bmatrix}, \quad \mathbf{Q}^{\mathbf{e}} = \begin{bmatrix} \mathbf{0} \\ \mathbf{A}^{\mathbf{e}} \end{bmatrix}, \quad \mathbf{G}_k^{\boldsymbol{\zeta}} = \begin{bmatrix} \mathbf{G}_k^{\boldsymbol{\xi}} & \mathbf{G}^{\mathbf{E}} \end{bmatrix}. \tag{16}$$

Note that the model in (15) is a state-space model with known matrices (i.e., the matrices contain known input and output data). The only thing unknown is the state vector $\boldsymbol{\zeta}_k$. We will estimate the state vector using a Kalman-like filter with forgetting factor. Note that estimating the state vector $\boldsymbol{\zeta}_k$ in (12) implies estimating the all parameters of the vector $\boldsymbol{\xi}$ in (9). The estimates produced by the Kalman-like filter corresponds to the solutions of the following sequence of optimization problems:

$$\begin{aligned} \min_{\{\boldsymbol{\zeta}_r\}_{r=0}^{k}, \{\mathbf{e}_r\}_{r=0}^{k-1}} & \left\{ \sum_{r=0}^{k-1} \lambda^{k-r} \|\mathbf{e}_r\|^2 \right. \\ & \left. + \lambda^k (\boldsymbol{\zeta}_0 - \hat{\boldsymbol{\zeta}}_0)^T \mathbf{P}_0^{-1} (\boldsymbol{\zeta}_0 - \hat{\boldsymbol{\zeta}}_0) \right\} \\ \text{s.t.} & \begin{cases} \boldsymbol{\zeta}_{r+1} = \mathbf{Q}^{\boldsymbol{\zeta}} \boldsymbol{\zeta}_r + \mathbf{Q}^{\mathbf{e}} \mathbf{e}_r, \\ \mathbf{y}_r = \mathbf{G}_r^{\boldsymbol{\zeta}} \boldsymbol{\zeta}_r + \mathbf{G}^{\mathbf{e}} \mathbf{e}_r, \quad \forall r \in \{0, 1, \dots, k-1\} \end{cases} \end{aligned} \tag{17}$$

for all $k \in \mathbb{N}$. Here, the constant $\lambda \in (0, 1]$ is the forgetting factor, $\hat{\boldsymbol{\zeta}}_0$ is a real vector, and $\mathbf{P}_0$ is a symmetric, positive definite matrix. Note that, by minimizing the optimization problems in (17), we find the state vector $\boldsymbol{\zeta}_k$ the produces the smallest, weighted model errors $\mathbf{e}_k$.

Lemma 7. *The minimizers $\hat{\boldsymbol{\zeta}}_k$ of the sequence of optimization problems in* (17) *corresponding to $\boldsymbol{\zeta}_k$ can be written as the output of the following recursive, Kalman-like equations:*

$$\hat{\boldsymbol{\zeta}}_{k+1} = \mathbf{Q}^{\boldsymbol{\zeta}} \hat{\boldsymbol{\zeta}}_k + \mathbf{K}_k (\mathbf{y}_k - \mathbf{G}_k^{\boldsymbol{\zeta}} \hat{\boldsymbol{\zeta}}_k) \tag{18}$$

and

$$\begin{aligned} \mathbf{P}_{k+1} = \frac{1}{\lambda} \bigg(& (\mathbf{Q}^{\boldsymbol{\zeta}} - \mathbf{K}_k \mathbf{G}_k^{\boldsymbol{\zeta}}) \mathbf{P}_k (\mathbf{Q}^{\boldsymbol{\zeta}} - \mathbf{K}_k \mathbf{G}_k^{\boldsymbol{\zeta}})^T \\ & + (\mathbf{Q}^{\mathbf{e}} - \mathbf{K}_k \mathbf{G}^{\mathbf{e}})(\mathbf{Q}^{\mathbf{e}} - \mathbf{K}_k \mathbf{G}^{\mathbf{e}})^T \bigg), \end{aligned} \tag{19}$$

with gain matrix

$$\begin{aligned} \mathbf{K}_k = & \Big(\mathbf{Q}^{\boldsymbol{\zeta}} \mathbf{P}_k (\mathbf{G}_k^{\boldsymbol{\zeta}})^T + \mathbf{Q}^{\mathbf{e}} (\mathbf{G}^{\mathbf{e}})^T \Big) \\ & \times \Big(\mathbf{G}_k^{\boldsymbol{\zeta}} \mathbf{P}_k (\mathbf{G}_k^{\boldsymbol{\zeta}})^T + \mathbf{G}^{\mathbf{e}} (\mathbf{G}^{\mathbf{e}})^T \Big)^{-1}, \end{aligned} \tag{20}$$

for all $k \geq 0$, where $\hat{\boldsymbol{\zeta}}_0$ and $\mathbf{P}_0$ are the initial conditions of the recursion equations.

Proof. A sketch of the proof is given in Appendix B. □

We use the minimizer $\hat{\boldsymbol{\zeta}}_k$ as an estimate of $\boldsymbol{\zeta}_k$ and, therefore, as an estimate for the unknown model parameters of the ARMAX model in (4).

Remark 8. *A forgetting factor close to one is standard practice in recursive estimation methods. Lower forgetting factors are typically used when the system exhibits time-varying behavior or nonlinear dynamics that a fixed linear model cannot capture well. In such cases, older data becomes less relevant, and faster adaptation to recent changes is beneficial.*

IV. Simulation

A. System description - stirred tanks in series

Here, two stirred tanks in series are described by a model of the form (1) where $\mathbf{F}^{\mathbf{x}} = \mathrm{expm}(\mathbf{F}^{\mathbf{x},\mathbf{C}} T)$ and $\mathbf{F}^{\mathbf{u}} = (\mathbf{F}^{\mathbf{x},\mathbf{C}})^{-1} (\mathbf{F}^{\mathbf{x}} - \mathbf{I}) \mathbf{F}^{\mathbf{u},\mathbf{C}}$, with

$$\mathbf{F}^{\mathbf{x},\mathbf{C}} = \begin{bmatrix} -\frac{F}{V_1} & 0 \\ \frac{F}{V_2} & -\frac{F}{V_2} \end{bmatrix}, \quad \mathbf{F}^{\mathbf{u},\mathbf{C}} = \begin{bmatrix} \frac{F}{V_1} \\ 0 \end{bmatrix}, \tag{21}$$

the state vector $\mathbf{x}_k$ having elements being the concentration of the first and second tank, respectively; the input being the concentration into the first tank, and the output being the concentration of the second tank. Furthermore, F is the flow rate, $V_1 = 150\,\mathrm{m}^3$ and $V_2 = 100\,\mathrm{m}^3$ the volumes of the first and second tank respectively, and $T = 10\,\mathrm{s}$ the sampling rate. Hence, $\mathbf{f} = \mathbf{0}$, $\mathbf{h} = \mathbf{0}$, $\mathbf{H}^{\mathbf{x}} = [0, 1]$ and $\mathbf{H}^{\mathbf{u}} = 0$. We use

$$\mathbf{F}^{\mathbf{e}} = \begin{bmatrix} 0.01 & 0 \\ 0 & 0 \end{bmatrix}, \tag{22}$$

and $\mathbf{H}^{\mathbf{e}} = [0, 0.005]$. $\mathbf{e}_k$ are chosen as white, independent, standard, Gaussian variables. The system is simulated for $3000\,\mathrm{s}$. We differentiate between two different cases: When flow rate is constant $F = 1\,\mathrm{m}^3\,\mathrm{s}^{-1}$, and when flow rate is increasing $F(t) = 1 + 0.005\frac{t}{T}\mathrm{m}^3\,\mathrm{s}^{-1}$.

B. Results

In Figure 1 we have plotted $\|\hat{\boldsymbol{\xi}}(t) - \boldsymbol{\xi}\|$ for different tuning parameters using the recursive estimator provided in Lemma 7. In blue and red, the true values, $\mathbf{G}^{\mathbf{E}} = \begin{bmatrix} 0.0009 & 0.0042 & 0 & -0.009239 \end{bmatrix}$ and $\mathbf{G}^{\mathbf{e}} = \begin{bmatrix} 0 & 0.005 \end{bmatrix}$ are used with forgetting factor $\lambda = 1.0$ and $\lambda = 0.9$, respectively. In both cases the parameter estimation error vanishes quickly. In yellow and purple $\mathbf{G}^{\mathbf{E}} = 0.01 \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix}$ and $\mathbf{G}^{\mathbf{e}} = \begin{bmatrix} 1 & 1 \end{bmatrix}$, while the forgetting factor is $\lambda = 1.0$ and $\lambda = 0.9$, respectively. The chosen values for $\mathbf{G}^{\mathbf{E}}$ and $\mathbf{G}^{\mathbf{e}}$ in yellow and purple are solely for illustrative purposes, demonstrating an example where values different from the true ones are used.

In Figure 2 the corresponding estimated outputs are plotted and compared to the true output, given by black circles. We see that the estimated outputs matches the true output really well, although there is some deviation for the case when $\mathbf{G}^{\mathbf{E}}$ and $\mathbf{G}^{\mathbf{e}}$ are unknown, and no forgetting factor is used.

Figure 3 is similar to Figure 2 for the case when the flow rate F is increasing. For this case we only consider $\mathbf{G}^{\mathbf{E}}$ and $\mathbf{G}^{\mathbf{e}}$ unknown. We have also compared with other methods using Normalized Root Mean Square Error, expressed as a percentage:

$$v_{\text{fit}} = \max\left(1 - \frac{\|\mathbf{y} - \hat{\mathbf{y}}\|}{\|\mathbf{y} - \overline{\mathbf{y}}\|}, 0\right) \cdot 100\%, \tag{23}$$

where $\overline{\mathbf{y}}$ is the arithmetic mean of $\mathbf{y}$. It is calculated for our method (with four different set of parameters), as well as for $\hat{\mathbf{y}} = \hat{\mathbf{y}}(t)^{\text{CVA}}$, $\hat{\mathbf{y}} = \hat{\mathbf{y}}(t)^{\text{SSARX}}$, $\hat{\mathbf{y}} = \hat{\mathbf{y}}(t)^{\text{MOESP}}$, with these being the estimated outputs using MATLAB's *n4sid*[1] function with 'N4Weight' set to 'CVA', 'SSARX' and 'MOESP', respectively. The method calculates a state-space model on innovation form, and we predict the output based on the estimated steady-state Kalman gain. For the $\hat{\mathbf{y}} = \hat{\mathbf{y}}(t)^{\text{RPBSID}}$, see [5, Algorithm 2], we initialized with the parameters in Table I. We emphasize that these methods all aim to estimate the noise-characteristics, where as this is considered a potential future improvement of our proposed algorithm. We trained 'CVA', 'SSARX' and 'MOESP' with input and output data from the interval 0 s-2000 s, then we compared the fit on the intervals INT1: 50 s-1000 s, INT2: 1000 s-2000 s, and INT3: 2000 s-3000 s. The motivation is to have one interval that covers the transient, but excluding the first couple of samples to allow recursive methods to have sufficient data to provide output; one interval where all methods are trained; and finally one interval where the batch methods have not seen the data before.

[1]*n4sid* documentation: https://se.mathworks.com/help/ident/ref/n4sid.html

TABLE I
PARAMETERS OF RPBSID$_{\text{PM}}$ ALGORITHM OF [5, ALGORITHM 2]

Parameter	Value
λ_1 λ_2, λ_3	0.9
ρ_1, ρ_2, ρ_3	0.2
p	5
f	5
n	2
$\mathbf{S}$	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$
$\boldsymbol{\Theta}_x$	$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$
$\boldsymbol{\Theta}_y$	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$

The results can be seen in Table II and Table IV, for F constant and F increasing, respectively. For F constant, we have also calculated the Normalized Estimated Mean Square (NEES), which can be seen in Table III. Good performance is indicated with values close to 1. The NEES is defined as

$$\text{NEES} = \frac{1}{(k_N - k_0 + 1)n_{\mathbf{y}}} \sum_{k=k_0}^{k_N} (\mathbf{y}_k - \hat{\mathbf{y}}_k)^T (\boldsymbol{\Sigma}_k^{\mathbf{y}})^{-1} (\mathbf{y}_k - \hat{\mathbf{y}}_k), \tag{24}$$

where $\boldsymbol{\Sigma}_k^{\mathbf{y}}$ is the output covariance matrix at time step k, and k_0 and k_N denote the initial and final time steps of the evaluation interval, respectively.

It is most interesting to see how our method for $\mathbf{G}^{\mathbf{E}}$ and $\mathbf{G}^{\mathbf{e}}$ unknown, compares to other methods. For the chosen initial conditions it consistently performs better than 'RPBSID'. We note however, that 'RPBSID' has many parameters, which can potentially be chosen in a way to improve performance. When F is constant we see from Table II, that the batch methods generally performs better than with unknown $\mathbf{G}^{\mathbf{E}}$ and $\mathbf{G}^{\mathbf{e}}$.

When F is increasing, we see from Table IV, that our recursive method with forgetting factor, performs better in the portion of the data set not seen by the batch methods, INT3. The results suggest that our method could substantially improve the real-time prediction of linear systems subject to parametric changes.

V. CONCLUSIONS AND FURTHER WORK

In this work we have presented a method for recursively estimating some characteristic parameters of an LTI system. To circumvent model redundancy related to the state vector, we first transform the state-space model into an ARMAX model. We then present a Kalman-like filter with forgetting factor to estimate unknown parameters of this model. It is shown that the algorithm estimates the ARMAX parameters

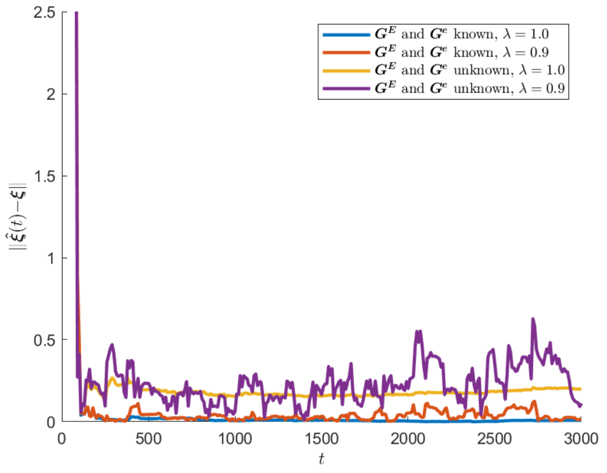

Fig. 1. Convergence of estimated parameters $\hat{\boldsymbol{\xi}}$ to true parameters $\boldsymbol{\xi}$ when F is constant.

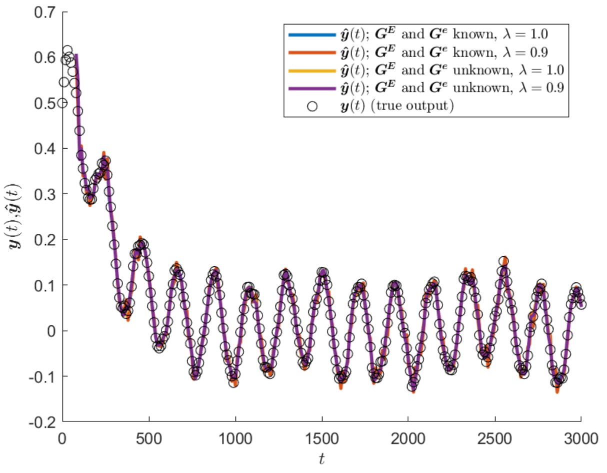

Fig. 2. Comparison of outputs when F is constant.

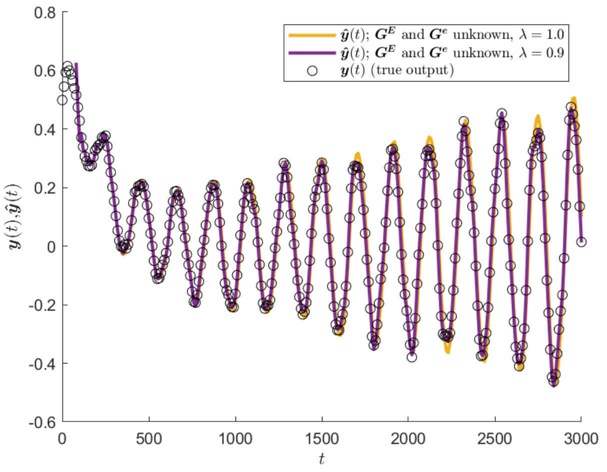

Fig. 3. Comparison of outputs when F is increasing.

TABLE II
A COMPARISON OF v_{FIT} DIFFERENT METHODS. F IS CONSTANT.

Method	INT1	INT2	INT3
$\hat{\boldsymbol{y}}(t)$; $\boldsymbol{G^E}$ and $\boldsymbol{G^e}$ known, $\lambda = 1.0$	91.3632	91.7682	91.2554
$\hat{\boldsymbol{y}}(t)$; $\boldsymbol{G^E}$ and $\boldsymbol{G^e}$ known, $\lambda = 0.9$	91.175	90.386	89.4944
$\hat{\boldsymbol{y}}(t)$; $\boldsymbol{G^E}$ and $\boldsymbol{G^e}$ unknown, $\lambda = 1.0$	91.4137	86.1063	84.7825
$\hat{\boldsymbol{y}}(t)$; $\boldsymbol{G^E}$ and $\boldsymbol{G^e}$ unknown, $\lambda = 0.9$	90.1553	82.2643	81.0943
$\hat{\boldsymbol{y}}^{\text{RPBSID}}(t)$	54.9948	59.9665	60.4047
$\hat{\boldsymbol{y}}^{\text{CVA}}(t)$	94.8391	91.731	90.7878
$\hat{\boldsymbol{y}}^{\text{SSARX}}(t)$	94.9902	91.8268	90.9086
$\hat{\boldsymbol{y}}^{\text{MOESP}}(t)$	95.9556	91.7311	90.7877

TABLE III
A COMPARISON OF NEES FOR DIFFERENT METHODS. F IS CONSTANT.

Method	INT1-3
$\hat{\boldsymbol{y}}(t)$; $\boldsymbol{G^E}$ and $\boldsymbol{G^e}$ known, $\lambda = 1.0$	0.92876
$\hat{\boldsymbol{y}}(t)$; $\boldsymbol{G^E}$ and $\boldsymbol{G^e}$ known, $\lambda = 0.9$	1.2306
$\hat{\boldsymbol{y}}(t)$; $\boldsymbol{G^E}$ and $\boldsymbol{G^e}$ unknown, $\lambda = 1.0$	2.5314
$\hat{\boldsymbol{y}}(t)$; $\boldsymbol{G^E}$ and $\boldsymbol{G^e}$ unknown, $\lambda = 0.9$	3.8854
$\hat{\boldsymbol{y}}^{\text{RPBSID}}(t)$	33.142
$\hat{\boldsymbol{y}}^{\text{CVA}}(t)$	0.90433
$\hat{\boldsymbol{y}}^{\text{SSARX}}(t)$	0.90468
$\hat{\boldsymbol{y}}^{\text{MOESP}}(t)$	0.89648

that produces the smallest, weighted model errors. We also show how the estimated parameters can be used in realizing a state-space model. Finally, we employ the algorithm on simulated data from two stirred tanks in series, and compare with other relevant methods for system identification. The results of the work will be further explored in the future, e.g. how to simultaneously estimate the parameters which are assumed known here, and how to design control signals for simultaneous identification and reference tracking.

ACKNOWLEDGEMENT

We would like to express our sincere gratitude to Prof. Lennart Ljung for his insightful questions regarding the problem formulation, which greatly contributed to the clarity and depth of this research.

APPENDIX A
PROOF OF LEMMA 1

Let us first introduce the following vectors and matrices:

$$\mathbf{m}_i = \begin{cases} \mathbf{h}, & \text{if } i = 0, \\ \mathbf{H^x}\left(\mathbf{F^x}\right)^{i-1}\mathbf{f}, & \text{if } i > 0, \end{cases} \tag{25}$$

$$\mathbf{M}_i^{\mathbf{x}} = \mathbf{H^x}\left(\mathbf{F^x}\right)^{i}, \tag{26}$$

$$\mathbf{M}_i^{\mathbf{u}} = \begin{cases} \mathbf{H^u}, & \text{if } i = 0, \\ \mathbf{H^x}\left(\mathbf{F^x}\right)^{i-1}\mathbf{F^u}, & \text{if } i > 0, \end{cases} \tag{27}$$

and

$$\mathbf{M}_i^{\mathbf{e}} = \begin{cases} \mathbf{H^e}, & \text{if } i = 0, \\ \mathbf{H^x}\left(\mathbf{F^x}\right)^{i-1}\mathbf{F^e}, & \text{if } i > 0, \end{cases} \tag{28}$$

for all $i \in \mathbb{N}$. From the equations of the state-space model in (1), it follows that the output $\mathbf{y}_k$ can be written as

$$\mathbf{y}_k = \mathbf{r} + \mathbf{R^x}\mathbf{x}_{k-d} + \mathbf{R^U}\mathbf{U}_k + \mathbf{R^u}\mathbf{u}_k + \mathbf{R^E}\mathbf{E}_k + \mathbf{R^e}\mathbf{e}_k, \tag{29}$$

TABLE IV
A COMPARISON OF v_{FIT} DIFFERENT METHODS. F IS INCREASING.

Method	INT1	INT2	INT3
$\hat{\boldsymbol{y}}(t)$; $\boldsymbol{G}^{\boldsymbol{E}}$ and $\boldsymbol{G}^{\boldsymbol{e}}$ unknown, $\lambda = 1.0$	92.3262	93.0369	93.1457
$\hat{\boldsymbol{y}}(t)$; $\boldsymbol{G}^{\boldsymbol{E}}$ and $\boldsymbol{G}^{\boldsymbol{e}}$ unknown, $\lambda = 0.9$	91.4133	93.1198	95.0029
$\hat{\boldsymbol{y}}^{\text{RPBSID}}(t)$	54.6494	60.185	62.2116
$\hat{\boldsymbol{y}}^{\text{CVA}}(t)$	95.2272	89.9633	85.8529
$\hat{\boldsymbol{y}}^{\text{SSARX}}(t)$	95.3193	90.3366	86.2945
$\hat{\boldsymbol{y}}^{\text{MOESP}}(t)$	95.382	89.9586	85.8467

with

$$\begin{aligned} \mathbf{r} &= \sum_{i=0}^{d} \mathbf{m}_i, & \mathbf{R}^{\mathbf{x}} &= \mathbf{M}^{\mathbf{x}}_d, \\ \mathbf{R}^{\mathbf{U}} &= \begin{bmatrix} \mathbf{M}^{\mathbf{u}}_d & \mathbf{M}^{\mathbf{u}}_{d-1} & \dots & \mathbf{M}^{\mathbf{u}}_1 \end{bmatrix}, & \mathbf{R}^{\mathbf{u}} &= \mathbf{M}^{\mathbf{u}}_0, \\ \mathbf{R}^{\mathbf{E}} &= \begin{bmatrix} \mathbf{M}^{\mathbf{e}}_d & \mathbf{M}^{\mathbf{e}}_{d-1} & \dots & \mathbf{M}^{\mathbf{e}}_1 \end{bmatrix}, & \mathbf{R}^{\mathbf{e}} &= \mathbf{M}^{\mathbf{e}}_0 \end{aligned} \tag{30}$$

for any nonnegative integer d. Similarly, from the same equations, it follows that

$$\mathbf{Y}_k = \mathbf{v} + \mathbf{V}^{\mathbf{x}}\mathbf{x}_{k-d} + \mathbf{V}^{\mathbf{U}}\mathbf{U}_k + \mathbf{V}^{\mathbf{E}}\mathbf{E}_k, \tag{31}$$

with

$$\begin{aligned} \mathbf{v} &= \begin{bmatrix} \mathbf{m}_0 \\ \sum_{i=0}^{1} \mathbf{m}_i \\ \vdots \\ \sum_{i=0}^{d-1} \mathbf{m}_i \end{bmatrix}, & \mathbf{V}^{\mathbf{U}} &= \begin{bmatrix} \mathbf{M}^{\mathbf{u}}_0 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{M}^{\mathbf{u}}_1 & \mathbf{M}^{\mathbf{u}}_0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \mathbf{0} \\ \mathbf{M}^{\mathbf{u}}_{d-1} & \cdots & \mathbf{M}^{\mathbf{u}}_1 & \mathbf{M}^{\mathbf{u}}_0 \end{bmatrix}, \\ \mathbf{V}^{\mathbf{x}} &= \begin{bmatrix} \mathbf{M}^{\mathbf{x}}_0 \\ \mathbf{M}^{\mathbf{x}}_1 \\ \vdots \\ \mathbf{M}^{\mathbf{x}}_{d-1} \end{bmatrix}, & \mathbf{V}^{\mathbf{E}} &= \begin{bmatrix} \mathbf{M}^{\mathbf{e}}_0 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{M}^{\mathbf{e}}_1 & \mathbf{M}^{\mathbf{e}}_0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \mathbf{0} \\ \mathbf{M}^{\mathbf{e}}_{d-1} & \cdots & \mathbf{M}^{\mathbf{e}}_1 & \mathbf{M}^{\mathbf{e}}_0 \end{bmatrix} \end{aligned} \tag{32}$$

for any nonnegative integer d.

Lemma 9. *There exists an integer $d \leq n_{\mathbf{x}}$, such that*

$$\mathbf{R}^{\mathbf{x}} = \mathbf{R}^{\mathbf{x}} \left(\mathbf{V}^{\mathbf{x}}\right)^{+} \mathbf{V}^{\mathbf{x}}. \tag{33}$$

Proof. Let the characteristic polynomial of $\mathbf{F}^{\mathbf{x}}$ be given by

$$\sum_{i=0}^{n_{\mathbf{x}}} \alpha_{n_{\mathbf{x}}-i} \left(\mathbf{F}^{\mathbf{x}}\right)^i = \mathbf{0}, \tag{34}$$

with $\alpha_0 = 1$ and real coefficients α_i for $i \in \{1, 2, \dots, n_{\mathbf{x}}\}$. From (26), (30), and (34), it follows that

$$\mathbf{R}^{\mathbf{x}} = -\sum_{i=0}^{n_{\mathbf{x}}-1} \alpha_{n_{\mathbf{x}}-i}\mathbf{M}^{\mathbf{x}}_i. \tag{35}$$

Because $\mathbf{V}^{\mathbf{x}}$ contains $\mathbf{M}^{\mathbf{x}}_i$ for all $i \in \{0, 1, \dots, n_{\mathbf{x}} - 1\}$ if $d = n_{\mathbf{x}}$, it follows that each row of $\mathbf{R}^{\mathbf{x}}$ can be written as a linear combination of the rows of $\mathbf{V}^{\mathbf{x}}$, which implies that (33) is satisfied for $d = n_{\mathbf{x}}$. Because there may be smaller values of d for which (33) also holds, we have that $d \leq n_{\mathbf{x}}$. □

Let d be chosen such that the condition in (33) of Lemma 9 holds. By combining (29), (31), and (33), we obtain (4), with

$$\begin{aligned} \mathbf{g} &= \mathbf{r} - \mathbf{R}^{\mathbf{x}} \left(\mathbf{V}^{\mathbf{x}}\right)^{+} \mathbf{v}, & \mathbf{G}^{\mathbf{Y}} &= \mathbf{R}^{\mathbf{x}} \left(\mathbf{V}^{\mathbf{x}}\right)^{+}, \\ \mathbf{G}^{\mathbf{U}} &= \mathbf{R}^{\mathbf{U}} - \mathbf{R}^{\mathbf{x}} \left(\mathbf{V}^{\mathbf{x}}\right)^{+} \mathbf{V}^{\mathbf{U}}, & \mathbf{G}^{\mathbf{u}} &= \mathbf{R}^{\mathbf{u}}, \\ \mathbf{G}^{\mathbf{E}} &= \mathbf{R}^{\mathbf{E}} - \mathbf{R}^{\mathbf{x}} \left(\mathbf{V}^{\mathbf{x}}\right)^{+} \mathbf{V}^{\mathbf{E}}, & \mathbf{G}^{\mathbf{e}} &= \mathbf{R}^{\mathbf{e}}. \end{aligned} \tag{36}$$

APPENDIX B
SKETCH OF THE PROOF OF LEMMA 7

The proof of the lemma is inductive and starts at $k = 0$, working up to any value of $k \geq 0$. For $k = 0$, the optimization problem in (17) can be written as

$$\min_{\boldsymbol{\zeta}_0} (\boldsymbol{\zeta}_0 - \hat{\boldsymbol{\zeta}}_0)^T \mathbf{P}_0^{-1} (\boldsymbol{\zeta}_0 - \hat{\boldsymbol{\zeta}}_0). \tag{37}$$

It is easy to see that the minizer of the optimization problem is given by $\hat{\boldsymbol{\zeta}}_0$. For $k = 1$, the optimization problem is given by

$$\begin{aligned} &\min_{\boldsymbol{\zeta}_0, \boldsymbol{\zeta}_1, \mathbf{e}_0} \left\{ \lambda \|\mathbf{e}_0\|^2 + \lambda (\boldsymbol{\zeta}_0 - \hat{\boldsymbol{\zeta}}_0)^T \mathbf{P}_0^{-1} (\boldsymbol{\zeta}_0 - \hat{\boldsymbol{\zeta}}_0) \right\} \\ &\text{s.t.} \begin{cases} \boldsymbol{\zeta}_1 = \mathbf{Q}^{\boldsymbol{\zeta}} \boldsymbol{\zeta}_0 + \mathbf{Q}^{\mathbf{e}} \mathbf{e}_0, \\ \mathbf{y}_0 = \mathbf{G}^{\boldsymbol{\zeta}}_0 \boldsymbol{\zeta}_0 + \mathbf{G}^{\mathbf{e}} \mathbf{e}_0. \end{cases} \end{aligned} \tag{38}$$

We can rewrite this optimization problem as an unconstrained optimization problem using Lagrangian multipliers:

$$\begin{aligned} \min_{\boldsymbol{\zeta}_0, \boldsymbol{\zeta}_1, \mathbf{e}_0, \boldsymbol{\mu}_0, \boldsymbol{\rho}_0} &\Big\{ \lambda \|\mathbf{e}_0\|^2 + \lambda (\boldsymbol{\zeta}_0 - \hat{\boldsymbol{\zeta}}_0)^T \mathbf{P}_0^{-1} (\boldsymbol{\zeta}_0 - \hat{\boldsymbol{\zeta}}_0) \\ &+ 2\boldsymbol{\mu}_0^T \left(\boldsymbol{\zeta}_1 - \mathbf{Q}^{\boldsymbol{\zeta}} \boldsymbol{\zeta}_0 - \mathbf{Q}^{\mathbf{e}} \mathbf{e}_0 \right) \\ &+ 2\boldsymbol{\rho}_0^T \left(\mathbf{y}_0 - \mathbf{G}^{\boldsymbol{\zeta}}_0 \boldsymbol{\zeta}_0 - \mathbf{G}^{\mathbf{e}} \mathbf{e}_0 \right) \Big\}, \end{aligned} \tag{39}$$

where $\boldsymbol{\mu}_0$ and $\boldsymbol{\rho}_0$ are vectors of Lagrangian multipliers. We (partially) minimize with respect to $\boldsymbol{\zeta}_0$, $\mathbf{e}_0$, $\boldsymbol{\mu}_0$, and $\boldsymbol{\rho}_0$. Therefore, we set the derivatives of the optimization cost function with respect to these variables to zero, which leads to the following set of equations:

$$\begin{bmatrix} \lambda \mathbf{P}_0^{-1} & \mathbf{0} & \left(\mathbf{Q}^{\boldsymbol{\zeta}}\right)^T & \left(\mathbf{G}^{\boldsymbol{\zeta}}_0\right)^T \\ \mathbf{0} & \lambda \mathbf{I} & \left(\mathbf{Q}^{\mathbf{e}}\right)^T & \left(\mathbf{G}^{\mathbf{e}}_0\right)^T \\ \mathbf{Q}^{\boldsymbol{\zeta}} & \mathbf{Q}^{\mathbf{e}} & \mathbf{0} & \mathbf{0} \\ \mathbf{G}^{\boldsymbol{\zeta}}_0 & \mathbf{G}^{\mathbf{e}}_0 & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\zeta}_0 \\ \mathbf{e}_0 \\ \boldsymbol{\mu}_0 \\ \boldsymbol{\rho}_0 \end{bmatrix} = \begin{bmatrix} \lambda \mathbf{P}_0^{-1} \hat{\boldsymbol{\zeta}}_0 \\ \mathbf{0} \\ \boldsymbol{\zeta}_1 \\ \mathbf{y}_0 \end{bmatrix}. \tag{40}$$

A solution of this set of equations is given by

$$\begin{aligned} \boldsymbol{\zeta}_0 &= \hat{\boldsymbol{\zeta}}_0 + \frac{1}{\lambda} \mathbf{P}_0 (\mathbf{Q}^{\boldsymbol{\zeta}} - \mathbf{K}_0 \mathbf{G}^{\boldsymbol{\zeta}}_0)^T \mathbf{P}_1^{-1} (\boldsymbol{\zeta}_1 - \hat{\boldsymbol{\zeta}}_1) \\ &\quad + \mathbf{P}_0 (\mathbf{G}^{\boldsymbol{\zeta}}_0)^T \left(\mathbf{G}^{\boldsymbol{\zeta}}_0 \mathbf{P}_0 (\mathbf{G}^{\boldsymbol{\zeta}}_0)^T + \mathbf{G}^{\mathbf{e}} (\mathbf{G}^{\mathbf{e}})^T \right)^{-1} (\mathbf{y}_0 - \mathbf{G}^{\boldsymbol{\zeta}}_0 \hat{\boldsymbol{\zeta}}_0), \\ \mathbf{e}_0 &= \frac{1}{\lambda} (\mathbf{Q}^{\mathbf{e}} - \mathbf{K}_0 \mathbf{G}^{\mathbf{e}}_0)^T \mathbf{P}_1^{-1} (\boldsymbol{\zeta}_1 - \hat{\boldsymbol{\zeta}}_1) \\ &\quad + (\mathbf{G}^{\mathbf{e}})^T \left(\mathbf{G}^{\boldsymbol{\zeta}}_0 \mathbf{P}_0 (\mathbf{G}^{\boldsymbol{\zeta}}_0)^T + \mathbf{G}^{\mathbf{e}} (\mathbf{G}^{\mathbf{e}})^T \right)^{-1} (\mathbf{y}_0 - \mathbf{G}^{\boldsymbol{\zeta}}_0 \hat{\boldsymbol{\zeta}}_0), \\ \boldsymbol{\mu}_0 &= -\mathbf{P}_1^{-1} (\boldsymbol{\zeta}_1 - \hat{\boldsymbol{\zeta}}_1), \\ \boldsymbol{\rho}_0 &= \mathbf{K}_0^T \mathbf{P}_1^{-1} (\boldsymbol{\zeta}_1 - \hat{\boldsymbol{\zeta}}_1) \\ &\quad - \lambda \left(\mathbf{G}^{\boldsymbol{\zeta}}_0 \mathbf{P}_0 (\mathbf{G}^{\boldsymbol{\zeta}}_0)^T + \mathbf{G}^{\mathbf{e}} (\mathbf{G}^{\mathbf{e}})^T \right)^{-1} (\mathbf{y}_0 - \mathbf{G}^{\boldsymbol{\zeta}}_0 \hat{\boldsymbol{\zeta}}_0), \end{aligned} \tag{41}$$

where $\hat{\boldsymbol{\zeta}}_1$, $\mathbf{P}_1$, and $\mathbf{K}_0$ are defined in (18), (19), and (20), respectively. Note that the invertibility of the matrices $\mathbf{P}_1$ and $\mathbf{G}_0^{\boldsymbol{\zeta}}\mathbf{P}_0(\mathbf{G}_0^{\boldsymbol{\zeta}})^T + \mathbf{G}^{\mathbf{e}}(\mathbf{G}^{\mathbf{e}})^T$ follows from the rank condition in (8) of Assumption 4 and the invertibility of the matrix $\mathbf{P}_0$. Substituting this solution in (38) yields

$$\min_{\boldsymbol{\zeta}_1}\Big\{(\boldsymbol{\zeta}_1 - \hat{\boldsymbol{\zeta}}_1)^T\mathbf{P}_1^{-1}(\boldsymbol{\zeta}_1 - \hat{\boldsymbol{\zeta}}_1) + \lambda(\mathbf{y}_0 - \mathbf{G}_0^{\boldsymbol{\zeta}}\hat{\boldsymbol{\zeta}}_0)^T \times \left(\mathbf{G}_0^{\boldsymbol{\zeta}}\mathbf{P}_0(\mathbf{G}_0^{\boldsymbol{\zeta}})^T + \mathbf{G}^{\mathbf{e}}(\mathbf{G}^{\mathbf{e}})^T\right)^{-1}(\mathbf{y}_0 - \mathbf{G}_0^{\boldsymbol{\zeta}}\hat{\boldsymbol{\zeta}}_0)\Big\}. \tag{42}$$

The corresponding minimizer is given by $\hat{\boldsymbol{\zeta}}_1$. Similarly, for any $k \geq 0$, by using a Lagrangian formulation and (partially) minimizing with respect to $\boldsymbol{\zeta}_0$, $\mathbf{e}_0$, $\boldsymbol{\mu}_0$, and $\boldsymbol{\rho}_0$, then minimizing with respect to $\boldsymbol{\zeta}_1$, $\mathbf{e}_1$, $\boldsymbol{\mu}_1$, and $\boldsymbol{\rho}_1$, subsequently minimizing with respect to $\boldsymbol{\zeta}_2$, $\mathbf{e}_2$, $\boldsymbol{\mu}_2$, and $\boldsymbol{\rho}_2$, etc., we can write any optimization problem in (17) as

$$\min_{\boldsymbol{\zeta}_k}\Big\{(\boldsymbol{\zeta}_k - \hat{\boldsymbol{\zeta}}_k)^T\mathbf{P}_k^{-1}(\boldsymbol{\zeta}_k - \hat{\boldsymbol{\zeta}}_k) + \sum_{r=0}^{k-1}\lambda^{k-r}(\mathbf{y}_r - \mathbf{G}_r^{\boldsymbol{\zeta}}\hat{\boldsymbol{\zeta}}_r)^T \times \left(\mathbf{G}_r^{\boldsymbol{\zeta}}\mathbf{P}_r(\mathbf{G}_r^{\boldsymbol{\zeta}})^T + \mathbf{G}^{\mathbf{e}}(\mathbf{G}^{\mathbf{e}})^T\right)^{-1}(\mathbf{y}_r - \mathbf{G}_r^{\boldsymbol{\zeta}}\hat{\boldsymbol{\zeta}}_r)\Big\}, \tag{43}$$

where $\hat{\boldsymbol{\zeta}}_k$ and $\mathbf{P}_k$ are obtained using the recursive equations in (18) and (19). Hence, for any $k \geq 0$, the minimizer for $\boldsymbol{\zeta}_k$ is given by $\hat{\boldsymbol{\zeta}}_k$.

References

[1] M Aoki. *State Space Modeling of Time Series*. Berling: Springer, 1990.

[2] Michelangelo Bin. Generalized recursive least squares: Stability, robustness, and excitation. *Systems & Control Letters*, 161:105144, March 2022.

[3] Adam L. Bruce, Ankit Goel, and Dennis S. Bernstein. Convergence and consistency of recursive least squares with variable-rate forgetting. *Automatica*, 119:109052, September 2020.

[4] Lei Guo, Lige Xia, and John B. Moore. Robust recursive identification of multidimensional linear regression models. *International Journal of Control*, 48(3):961–979, September 1988.

[5] Ivo Houtzager, Jan Willem Van Wingerden, and Michel Verhaegen. Recursive predictor-based subspace identification with application to the real-time closed-loop tracking of flutter. *IEEE Transactions on Control Systems Technology*, 20(4):934–949, 2012. Publisher: IEEE.

[6] Lennart Ljung. *System identification: theory for the user*. Prentice Hall information and system sciences series. Prentice Hall PTR, Upper Saddle River, NJ, 2nd ed edition, 1999.

[7] Guillaume Mercere, Ivan Markovsky, and Jose A. Ramos. Innovation-based subspace identification in open- and closed-loop. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 2951–2956, Las Vegas, NV, USA, December 2016. IEEE.

[8] T. Nicolai, M. Haring, E. I. Grøtli, J. T. Gravdahl, and J. Reger. Realizing LTI model by identifying chacteristic paramters using least squares optimization. In *Proc. of 2023 European Control Conference*, 2023.

[9] Hyo-Sang Shin and Hae-In Lee. A New Exponential Forgetting Algorithm for Recursive Least-Squares Parameter Estimation, April 2020. arXiv:2004.03910 [cs, eess].

[10] Henrik Spliid. A Fast Estimation Method for the Vector Autoregressive Moving Average Model with Exogenous Variables. *Journal of the American Statistical Association*, 78(384):843–849, December 1983.

[11] P. Stoica and M. Jansson. MIMO system identification: state-space and subspace approximations versus transfer function and instrumental variables. *IEEE Transactions on Signal Processing*, 48(11):3087–3099, November 2000.

[12] P. Stoica, T. Soderstrom, and B. Friedlander. Optimal instrumental variable estimates of the AR parameters of an ARMA process. *IEEE Transactions on Automatic Control*, 30(11):1066–1074, November 1985.

[13] Peter Van Overschee and Bart De Moor. *Subspace Identification for Linear Systems*. 1996. Publication Title: Subspace Identification for Linear Systems Issue: November 2014.

[14] I.N. Vuchkov and L.N. Boyadjieva. A New Algorithm of the Recursive Generalized Least-squares Method. *IFAC Proceedings Volumes*, 21(9):797–802, August 1988.

[15] Harm H.M. Weerts, Miguel Galrinho, Giulio Bottegal, Håkan Hjalmarsson, and Paul M.J. Van Den Hof. A sequential least squares algorithm for ARMAX dynamic network identification. *IFAC-PapersOnLine*, 51(15):844–849, 2018.