<http://www.diva-portal.org>

Postprint

This is the accepted version of a paper presented at *IEEE 56th Annual Conference on Decision and Control (CDC), DEC 12-15, 2017, Melbourne, AUSTRALIA.*

Citation for the original published paper:

Fang, M., Galrinho, M., Hjalmarsson, H. (2017)
Recursive Identification Based on Weighted Null-Space Fitting
In: *2017 IEEE 56TH ANNUAL CONFERENCE ON DECISION AND CONTROL (CDC)*
IEEE
IEEE Conference on Decision and Control
<https://doi.org/10.1109/CDC.2017.8264345>

N.B. When citing this work, cite the original published paper.

Permanent link to this version:

<http://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-223849>

Recursive Identification Based on Weighted Null-Space Fitting

Mengyuan Fang, Miguel Galrinho and Håkan Hjalmarsson

Abstract—Algorithms for online system identification consist in updating the estimated model while data are being collected. A standard method for online identification of structured models is the recursive prediction error method (PEM). The problem is that PEM does not have an exact recursive formulation, and the need to rely on approximations makes recursive PEM prone to convergence problems. In this paper, we propose a recursive implementation of weighted null-space fitting, an asymptotically efficient method for identification of structured models. Consisting only of (weighted) least-squares steps, the recursive version of the algorithm has the same convergence and statistical properties of the off-line version. We illustrate these properties with a simulation study, where the proposed algorithm always attains the performance of the off-line version, while recursive PEM often fails to converge.

I. INTRODUCTION

The topic of online estimation has been extensively studied in signal processing and system identification (e.g., [1]–[6]). Online estimation is important in applications that require online decisions. In such applications, the user cannot afford to separate the estimation procedure into a data-collection phase and an estimation phase. Rather, as data containing information about the signal or system are being collected over time, the estimate should be updated as new data samples are measured, constantly providing the user with an estimate based on the currently available information. In order to compute the estimate update before new data samples are collected, online estimation algorithms must comply with certain constraints on computational speed and memory allocation. Such algorithms are called *recursive*.

In this paper, we consider the problem of recursive system identification of structured models. In system identification, the prediction error method (PEM) is a standard choice because of its optimal asymptotic properties [7]. However, PEM does not have an exact recursive formulation, because the predictor and its gradient cannot be computed at the current parameter estimate. To circumvent this, recursive PEM relies on approximations [8].

Although it can be shown that PEM and recursive PEM have similar convergence properties—that is, the estimate will either converge to a global minimizer of the cost function or to the boundary of the set where the parameter vector

should lie—a projection mechanism to keep the estimate within this set is fundamental in the recursive version, as the estimate may easily diverge from the set when the sample size is small [8]. However, the effects of such projection mechanism are difficult to analyze [9].

Standard instrumental variable methods have an exact recursive formulation [7], but do not have optimal statistical properties even off-line. The refined instrumental variable method, which yields asymptotically efficient estimates in open loop, can easily be implemented recursively, but the recursive algorithm is less robust and less accurate than the off-line version for finite data [10]. When applied to Box-Jenkins models, it is stated in [8] that recursive PEM corresponds to the fully recursive version of [10].

We base our developments in this paper on the weighted null-space fitting (WNSF) method [11]: a three-step weighted least-squares method for estimation of structured models. In the first step, a high-order ARX model is estimated with least squares; in the remaining two steps, (weighted) least squares is used to reduce this high-order estimate to an asymptotically efficient estimate of the model of interest [12].

Using a high-order ARX model estimated with least squares makes WNSF appropriate for recursive identification, because least squares has an exact recursive formulation. This guarantees that the ARX-model estimate converges as data are collected over time. In turn, because convergence of the model-reduction step depends only on convergence of the ARX model, the low-order estimate obtained recursively has the same statistical properties as its off-line counterpart. The theoretical foundation for this argument has been laid out in [12], based on the analysis in [13].

In this paper, we propose a recursive WNSF algorithm, discuss issues with practical implementation, and perform a simulation study. This study shows that recursive WNSF performs identically to its off-line version, while recursive PEM often fails to converge.

Notation: $\mathcal{T}_{n \times m}[\{x_1, \dots, x_n\}]$ is the Toeplitz matrix of size $n \times m$ with $\{x_1, \dots, x_n\}$ the elements of the first column and zeros above the main diagonal.

II. PROBLEM STATEMENT

Consider that data are generated by

$$y_k = G_o(q)u_k + H_o(q)e_k, \quad (1)$$

where $\{u_k\}$ is a known input sequence (eventually obtained in closed-loop), $\{e_k\}$ is Gaussian white noise with variance σ_e^2 , and $G_o(q)$ and $H_o(q)$ are rational stable transfer functions

This work was supported by the Swedish Research Council under contracts 2015-05285 and 2016-06079, and by the National Science Foundation of China (No. 61673343).

M. Fang is with School of Control Science & Engineering, Zhejiang University, Hangzhou 310027, China myfang@iipc.zju.edu.cn

M. Galrinho and H. Hjalmarsson are with the Department of Automatic Control, School of Electrical Engineering, KTH Royal Institute of Technology Sweden, SE 10044, Sweden {galrinho,hjalmars}@kth.se

(in addition, $H_0^{-1}(q)$ is also stable) parametrized as

$$\begin{aligned} G(q, \theta) &= \frac{l_1 q^{-1} + \dots + l_m q^{-m}}{1 + f_1 q^{-1} + \dots + f_m q^{-m}} =: \frac{L(q, \theta)}{F(q, \theta)} \\ H(q, \theta) &= \frac{1 + c_1 q^{-1} + \dots + c_m q^{-m}}{1 + d_1 q^{-1} + \dots + d_m q^{-m}} =: \frac{C(q, \theta)}{D(q, \theta)} \end{aligned} \quad (2)$$

(for notational simplicity only, we consider all the polynomials to be of order m), where q^{-1} is the backward time-shift operator and

$$\theta = [f_1 \ \dots \ f_m \ l_1 \ \dots \ l_m \ c_1 \ \dots \ c_m \ d_1 \ \dots \ d_m]^\top.$$

We assume that there exists a unique $\theta = \theta_0$ such that $G(q, \theta_0) = G_0(q)$ and $H(q, \theta_0) = H_0(q)$.

The problem we consider is how to estimate the parameter-vector θ recursively. In recursive identification, we assume that we have data $\{u_k, y_k\}_{k=1}^{t-1}$ and a parameter-estimate $\hat{\theta}_{t-1}$. Then, as new data samples $\{u_t, y_t\}$ become available, we want to update the estimate. The new estimate $\hat{\theta}_t$ must be computed in a computationally efficient way, before new data samples $\{u_{t+1}, y_{t+1}\}$ become available. For this reason, it is not realistic to estimate $\hat{\theta}_t$ using the complete data set $\{u_k, y_k\}_{k=1}^t$. With a recursive method, the estimate update is based on the information contained in the previous estimate and the new data samples.

III. RECURSIVE PREDICTION ERROR IDENTIFICATION

A standard method for recursive identification of structured models is recursive PEM. In this section, we review this method and consider its drawbacks. Then, we consider the particular case of recursive least squares, which will be instrumental to the proposed method.

A. Recursive Prediction Error Method

The prediction error method consists in estimating the parameters of interest by minimizing a cost function of the prediction errors. With a quadratic cost is used, we minimize

$$V_t(\theta) = \frac{1}{t} \sum_{k=1}^t \varepsilon_k^2(\theta), \quad (3)$$

where t is the sample size and $\varepsilon_k(\theta) = y_k - \hat{y}_k(\theta)$, with

$$\hat{y}_k(\theta) = [1 - H^{-1}(q, \theta)]y_k + H^{-1}(q, \theta)G(q, \theta)u_k.$$

In general, minimizing (3) requires a search algorithm. For example, with Gauss-Newton the minimum is sought by updating the estimate iteratively as (at iteration i)

$$\hat{\theta}_t^{(i)} = \hat{\theta}_t^{(i-1)} - \Lambda_t^{-1} V_t'(\hat{\theta}_t^{(i-1)}),$$

where $V_t'(\theta)$ is the gradient of $V_t(\theta)$, given by

$$V_t'(\theta) = - \sum_{k=1}^t \psi_k(\theta) \varepsilon_k(\theta) \quad (4)$$

with $\psi_k(\theta) = d\hat{y}_k(\theta)/d\theta$, and Λ_t is typically taken as an approximation of the Hessian of $V_t(\theta)$ (see [7]).

If, for each iteration i , we also collect one more data point, we have

$$\hat{\theta}_t^{(i)} = \hat{\theta}_{t-1}^{(i-1)} - \Lambda_t^{-1} V_t'(\hat{\theta}_{t-1}^{(i-1)}). \quad (5)$$

Writing (4) recursively, we have

$$V_t'(\theta) = \frac{t-1}{t} V_{t-1}'(\theta) - \frac{1}{t} \psi_t(\theta) \varepsilon_t(\theta). \quad (6)$$

Then, assuming that $\hat{\theta}_{t-1}^{(i-1)}$ minimized $V_{t-1}(\theta)$, we have, replacing (6) in (5),

$$\hat{\theta}_t^{(i)} = \hat{\theta}_{t-1}^{(i-1)} + \frac{1}{t} \Lambda_t^{-1} \psi_t(\hat{\theta}_{t-1}^{(i-1)}) \varepsilon_t(\hat{\theta}_{t-1}^{(i-1)}).$$

The problem is that, because $\psi_t(\theta)$ and $\varepsilon_t(\theta)$ are derived from $\hat{y}_t(\theta)$, which is obtained as the output of a linear filter whose coefficients depend on θ , these variables cannot be computed with fixed-size memory. This is solved by approximating the required filters with time-varying filters where measurement updates are based on the latest available parameter estimate. These approximations often lead to convergence problems, as illustrated in Section VI.

B. Recursive Least Squares

For model structures that are linear in the parameters, there is no need for the aforementioned approximations. Consider an ARX model,

$$A(q, \eta)y_k = B(q, \eta)u_k + e_k, \quad (7)$$

where

$$\begin{aligned} A(q, \eta) &= 1 + \sum_{k=1}^n a_k q^{-k}, & B(q, \eta) &= \sum_{k=1}^n b_k q^{-k}, \\ \eta &= [a_1 \ \dots \ a_n \ b_1 \ \dots \ b_n]^\top. \end{aligned}$$

Because the ARX-model predictor is linear in the model parameters, the PEM estimate is obtained with least squares. This is done by computing

$$\hat{\eta}_t = R_t^{-1} r_t, \quad (8)$$

where

$$R_t = \frac{1}{t} \sum_{k=1}^t \varphi_k \varphi_k^\top, \quad r_t = \frac{1}{t} \sum_{k=1}^t \varphi_k y_k, \quad (9)$$

$$\varphi_k = [-y_{k-1} \ \dots \ -y_{k-n} \ u_{k-1} \ \dots \ u_{k-n}]^\top.$$

The estimate $\hat{\eta}_t$ has asymptotic covariance matrix $\sigma_e^2 R_t^{-1}$.

Because we have that

$$R_t = \frac{t-1}{t} R_{t-1} + \frac{\varphi_t \varphi_t^\top}{t}, \quad (10)$$

the ARX-model estimate can be updated recursively with [8]

$$\begin{aligned} \hat{\eta}_t &= \hat{\eta}_{t-1} + \frac{P_{t-1} \varphi_t}{1 + \varphi_t^\top P_{t-1} \varphi_t} (y_t - \varphi_t^\top \hat{\eta}_{t-1}), \\ P_t &= P_{t-1} - \frac{P_{t-1} \varphi_t \varphi_t^\top P_{t-1}}{1 + \varphi_t^\top P_{t-1} \varphi_t}, \end{aligned} \quad (11)$$

where $P_t := R_t^{-1}/t$.

If P_{t-1} and $\hat{\eta}_{t-1}$ are initialized correctly, the recursive formulation (11) is exact. Otherwise, the initialization only causes transient effects, as P_t and $\hat{\eta}_t$ have guaranteed convergence [8]. This beneficial property of recursive least squares will be fundamental for the proposed method.

IV. WEIGHTED NULL-SPACE FITTING

The weighted null-space fitting method is a method for identification of structured models. We proceed to review the method, which will be suitable for recursive identification without the issues of PEM.

The method consists of three steps. In the first step, we estimate a high-order ARX model with least squares. In the second step, we reduce the high-order model to a structured one with least squares. This step does not take into account the errors in the high-order estimate, but it provides a consistent estimate of the structured model. So, in the third step, we re-estimate the structured model with weighted least squares, where we use the estimate obtained in the second step to construct the weighting. The estimate obtained in the third step is asymptotically efficient.

For the first step, consider the true system (2) written as

$$A_o(q)y_k = B_o(q)u_k + e_k,$$

where

$$\begin{aligned} A_o(q) &:= \frac{1}{H_o(q)} =: 1 + \sum_{k=1}^{\infty} a_k^o q^{-k}, \\ B_o(q) &:= \frac{G_o(q)}{H_o(q)} =: \sum_{k=1}^{\infty} b_k^o q^{-k}. \end{aligned} \quad (12)$$

If $1/H_o(q)$ and $G_o(q)/H_o(q)$ are stable, the coefficients $\{a_k^o, b_k^o\}$ decay exponentially to zero [13] (see [14,15] for how the unstable case can be dealt with). Therefore, we can estimate a truncated version of the polynomials in (12), where we choose the order large enough for the truncation error to be small. This corresponds to estimating the ARX-model (7) with least squares, according to (8).

For the second step, we use $\hat{\eta}_t$ to obtain an estimate of θ . If we choose n sufficiently large for the truncation error to be small, using (12) and (2), we have that θ and η are related by

$$\begin{cases} A(q, \eta)C(q, \theta) - D(q, \theta) \approx 0 \\ B(q, \eta)F(q, \theta) - A(q, \eta)L(q, \theta) \approx 0 \end{cases} \quad (13)$$

Because (13) consists of polynomial equations in q^{-1} , (13) is satisfied if all the polynomial coefficients are approximately zero. Moreover, (13) is linear in θ , allowing us to write the relation between the polynomial coefficients in vector form:

$$Q(\eta)\theta - \eta \approx 0, \quad (14)$$

where $Q(\eta)$ is a block-Toeplitz matrix given by

$$Q(\eta) = \begin{bmatrix} 0 & 0 & -Q_A(\eta) & -Q_I(\eta) \\ -Q_B(\eta) & Q_A(\eta) & 0 & 0 \end{bmatrix},$$

with $Q_A = \mathcal{T}_{n \times m}[\{1, a_1, \dots, a_{n-1}\}]$, $Q_I = \mathcal{T}_{n \times m}[\{1, 0, \dots, 0\}]$, and $Q_B = \mathcal{T}_{n \times m}[\{0, b_1, \dots, b_{n-1}\}]$. Then, if we plug the estimate (8) in (14), we can obtain the estimate of θ that minimizes the squared residuals as

$$\hat{\theta}_t^{\text{LS}} = \left[Q^\top(\hat{\eta}_t) Q(\hat{\eta}_t) \right]^{-1} Q^\top(\hat{\eta}_t) \hat{\eta}_t. \quad (15)$$

When computing (15), we have not taken into account for the errors in $\hat{\eta}_t$. The third step remedies this by re-estimating θ taking into consideration that the residuals

$$z(\hat{\eta}_t, \theta_o) = Q(\hat{\eta}_t)\theta_o - \hat{\eta}_t,$$

are distributed as [12]

$$z(\hat{\eta}_t, \theta_o) \sim \mathcal{ASN} \left(0, T(\theta_o) R_t^{-1} T^\top(\theta_o) \right), \quad (16)$$

where R_t is given by (9) and

$$T(\theta) = \begin{bmatrix} T_C(\theta) & 0 \\ -T_L(\theta) & T_F(\theta) \end{bmatrix},$$

with $T_C(\theta) = \mathcal{T}_{n \times n}[\{1, c_1, \dots, c_m, 0, \dots, 0\}]$, $T_L = \mathcal{T}_{n \times n}[\{0, l_1, \dots, l_m, 0, \dots, 0\}]$, and $T_F = \mathcal{T}_{n \times n}[\{1, f_1, \dots, f_m, 0, \dots, 0\}]$. Then, the estimate of θ with minimum variance with respect to the residuals (16) is obtained by weighted least squares, with the weighting given by the inverse of the covariance matrix in (16). Because this covariance matrix depends on the true parameters θ_o , we replace them by their estimate $\hat{\theta}_t^{\text{LS}}$ obtained in the second step. Then, the third step consists of computing

$$\hat{\theta}_t^{\text{WLS}} = \left[Q^\top(\hat{\eta}_t) W(\hat{\theta}_t^{\text{LS}}) Q(\hat{\eta}_t) \right]^{-1} Q^\top(\hat{\eta}_t) W(\hat{\theta}_t^{\text{LS}}) \hat{\eta}_t, \quad (17)$$

where $W(\hat{\theta}_t^{\text{LS}}) = [T(\hat{\theta}_t^{\text{LS}}) R_t^{-1} T^\top(\hat{\theta}_t^{\text{LS}})]^{-1}$.

In summary, WNSF consists of the following steps:

- 1) estimate a high-order ARX model with (8),
- 2) reduce to a low-order model with (15),
- 3) re-estimate the low-order model with (17).

In [12], it is shown that the estimate obtained in the third step is asymptotically efficient. However, for finite data, it may be beneficial to continue to iterate. In this case, the low-order model is re-estimated iteratively, with the weighting matrix constructed using the estimate obtained in the previous iteration. Then, we can write, at iteration i ,

$$\hat{\theta}_t^{(i)} = \left[Q^\top(\hat{\eta}_t) W(\hat{\theta}_t^{(i-1)}) Q(\hat{\eta}_t) \right]^{-1} Q^\top(\hat{\eta}_t) W(\hat{\theta}_t^{(i-1)}) \hat{\eta}_t. \quad (18)$$

From (18), to update a WNSF estimate we require a previous low-order estimate, and a high-order ARX-model estimate and covariance. This makes WNSF appropriate for recursive estimation: the previous estimate is available from the previous sample time; the high-order ARX-model estimate and covariance can be updated recursively. In the following section, we formalize this procedure.

V. RECURSIVE WEIGHTED NULL-SPACE FITTING

We proceed by presenting the recursive weighted null-space fitting algorithm and discussing a few practical aspects.

A. The algorithm

The idea of the recursive WNSF algorithm is to update the ARX-model estimate recursively with (11), which is used to update the low-order estimate based on the off-line iterative formulation (18). However, because in the recursive

formulation iteration i corresponds to the sample size t , we may simply write

$$\hat{\theta}_t = \left[Q^\top(\hat{\eta}_t) W(\hat{\theta}_{t-1}) Q(\hat{\eta}_t) \right]^{-1} Q^\top(\hat{\eta}_t) W(\hat{\theta}_{t-1}) \hat{\eta}_t. \quad (19)$$

where

$$W(\hat{\theta}_{t-1}) = T^{-\top}(\hat{\theta}_{t-1}) R_t T^{-1}(\hat{\theta}_{t-1}). \quad (20)$$

The algorithm can be described as follows. Assume that we have an estimate $\hat{\theta}_{t-1}$ based on the data set $\{y_k, u_k\}_{k=1}^{t-1}$, a high-order ARX-model estimate $\hat{\eta}_{t-1}$, and the covariance matrix inverse R_{t-1} . Then, as data samples $\{y_t, u_t\}$ become available, recursive WNSF consists of two steps:

- 1) compute $\hat{\eta}_t$ with (11) and R_t with (10);
- 2) compute $\hat{\theta}_t$ with (19).

Compared to off-line WNSF, the second step of that algorithm is not required in the recursive version, as a low-order estimate to construct the weighting is in principle already available from the previous time step.

B. Practical Aspects

We now discuss a few practical aspects of the algorithm.

1) *Initialization*: The parameters $\hat{\eta}_0$, R_0 , and $\hat{\theta}_0$ are in principle required for initialization. If an initial model is unavailable, $\hat{\eta}_0$ and R_0 can be initialized with small values (because the estimate $\hat{\eta}_0$ will be poor, this ensures a large initial covariance matrix R_0^{-1}). Accurate values are not necessary, as recursive least squares will converge irrespective of initialization. Concerning $\hat{\theta}_0$, there is no difference with respect to the off-line case: if an estimate of θ is not available, a least-squares step (15) can be taken instead.

2) *ARX-model order*: With off-line WNSF, the order of the ARX model can be optimized by applying WNSF to a grid of ARX-model orders, and picking the low-order model that minimizes the prediction error cost function [11]. With recursive WNSF, this approach may be too computationally heavy for some applications. In this paper, we assume that we have an idea of the class of systems under study (i.e., if they are fast or slow), and choose the ARX-model order sufficiently high to capture the dynamics of such class of systems. In Section VI-B, we observe that the order is not so critical as long as it is large enough to accurately capture the system dynamics. Nevertheless, we briefly discuss in Section VII how an algorithm that adapts the high order will be implemented in future work.

3) *High-dimension matrix inversion*: From (20), we observe that the recursive WNSF algorithm requires that we invert the high-dimension matrix $T(\hat{\theta}_{t-1})$, $2n \times 2n$. It is not feasible to explicitly compute the inverse at every recursion. However, the inverse of this matrix,

$$T^{-1}(\theta) = \begin{bmatrix} T_C^{-1}(\theta) & 0 \\ T_F^{-1}(\theta) T_L(\theta) T_C^{-1}(\theta) & T_F^{-1}(\theta) \end{bmatrix}, \quad (21)$$

can be computed efficiently: because $T_F(\theta)$ and $T_C(\theta)$ are lower-triangular Toeplitz matrices whose first column contains the coefficients of $F(q, \theta)$ and $C(q, \theta)$, respectively,

the inverses are given by $T_F^{-1}(\theta) = \mathcal{T}_{n \times n}[\{1, \bar{f}_1, \dots, \bar{f}_{n-1}\}]$ and $T_C^{-1}(\theta) = \mathcal{T}_{n \times n}[\{1, \bar{c}_1, \dots, \bar{c}_{n-1}\}]$, where

$$\frac{1}{F(q, \theta)} =: 1 + \sum_{k=1}^{\infty} \bar{f}_k q^{-k}, \quad \frac{1}{C(q, \theta)} =: 1 + \sum_{k=1}^{\infty} \bar{c}_k q^{-k}. \quad (22)$$

Therefore, to invert $T(\theta)$, we need only to compute the matrix products in (21) and the filters in (22).

4) *Instability*: If the current estimate of θ leaves the stability region, the weighting matrix $W(\theta)$ is not numerically well behaved. Therefore, if such a case happens (which is likely while the available sample size is small) the current estimate of θ should not be used, and a least-squares step can be taken instead. Weighted least squares may be used again when the estimate of θ is inside the stability region. Such a “reset” of θ does not have the same consequences as resetting in recursive PEM. In the latter, information is lost because of the unavailability of past data. In recursive WNSF, the information is condensed in the high-order ARX model, and this estimate is never discarded.

VI. NUMERICAL SIMULATIONS

In this section, we first perform two simulation studies: with a fixed system and with a certain class of randomly generated systems. In the first study, we consider two scenarios: with and without initial model available. In the second study, there is no information about the model to start the recursive algorithms.

Four methods are compared:

- 1) off-line prediction error method (PEM);
- 2) recursive prediction error method (rPEM);
- 3) off-line weighted null-space fitting, with 20 iterations of (18) (WNSF);
- 4) recursive weighted null-space fitting (rWNSF).

The hardware used was a MacBook Pro equipped with an Intel Core i5-1600M CPU running at 2.60 GHz and with 8GB RAM. The software was Matlab R2015b, whose PEM implementation and default settings were used. For the recursive version, we use the “ForgettingFactor” option (with forgetting factor set to 1, as we do not consider time-varying systems), which is computationally heavier than other available options, but has better convergence properties. All methods estimate a model with correct structure.

A. Fixed system

In this simulation, we generate data using (1), where

$$G_o(q) = \frac{q^{-1} + 0.5q^{-2} - 2q^{-3} + q^{-4}}{1 - 1.5q^{-1} + 0.7q^{-2} + 0.3q^{-3} - 0.2q^{-4}},$$

$$H_o(q) = \frac{1 + 0.5q^{-1}}{1 - 0.6q^{-1}},$$

the input is a general binary noise (GBN) signal with average switching time equals to 10, and the noise variance relative to the variance of the noise-free output being 30%. For WNSF and rWNSF, the order of the high-order ARX model is $n = 80$. Two hundred Monte-Carlo simulations are performed with independent noise realizations. The data length is 3000.

Fig. 1. FITs of initialization model obtained with PEM (PEM_init), recursive PEM with default covariance initialization (rPEM_def), recursive PEM with covariance initialization using asymptotic covariance evaluated at PEM_init (rPEM_cov), and recursive WNSF (rWNSF), using a fixed system.

In the first scenario of this simulation, we consider that we have an initial estimate available. To obtain the initial model, we use the first 300 data samples. To initialize recursive WNSF, we estimate a high-order ARX-model and its covariance matrix (as discussed in Section V-B.1, an estimate of θ is not required for initialization). To initialize recursive PEM, we estimate a structured model with the correct orders using off-line PEM (we denote this initial estimate by PEM_init). When initializing recursive PEM, an initial parameter covariance matrix should also be assigned. One possibility is to use the default version of the algorithm, which initializes the parameter covariance to an identity matrix scaled by 10000 (we denote the estimate obtained recursively with this initialization rPEM_def). However, a more sensible initialization is to estimate the covariance based on the the asymptotic covariance formula [7], and evaluate it at the available parameter estimates (we denote the estimate obtained recursively with this initialization rPEM_cov). This is not an issue with recursive WNSF, as the initial parameter covariance for the high-order ARX-model is readily available from the least-squares estimator.

Performance is evaluated by computing the FIT of simulated data on noise-free data, given by, in percent,

$$\text{FIT} = 100 \left(1 - \frac{\|y^o - \hat{y}\|_2}{\|y^o - \text{mean}(y^o)\|_2} \right),$$

where y^o and $\hat{y}$ are vectors containing, respectively, the noise-free output and the simulated output (in recursive methods, the simulated output uses the final estimate obtained).

The results for the scenario with an initial model are presented in Fig. 1. Here, we observe how recursive PEM is sensitive to the initial parameter covariance initialization. The default version often diverges, degrading the initial parameter estimate. In this case, using the asymptotic covariance evaluated at the initial model parameters was good enough for recursive PEM to improve the accuracy of the initial estimate. However, recursive WNSF, whose initialization only required solving a least-squares problem, performs better than both version of recursive PEM.

In the second scenario, we do not use any model for initialization. In this case, we apply recursive PEM with default initialization of the parameters and the covariance.

Fig. 2. FITs of estimates obtained with, from left to right, off-line PEM, recursive PEM, off-line WNSF, and recursive WNSF, using a fixed system and no initial model estimate.

For recursive WNSF, we initialize the high-order ARX-model $\hat{\eta}_0$ with zeros and R_0 with an identity matrix scaled by 0.0001 (i.e., the parameter covariance is initialized as identity scaled by 10000, analogously to the default version of recursive PEM). Here, we also compare with the respective off-line versions using the complete data set.

The results for this scenario are presented in Fig. 2. Here, the off-line versions of PEM and WNSF perform almost identically. Moreover, the performance of recursive WNSF coincide with the performance in Fig. 1, where an initial model was available.

With these simulations, we have illustrated the argument that recursive WNSF has the same convergence properties of its off-line version, and is robust against poor initial conditions. These properties are a consequence of WNSF condensing the data information in a high-order ARX model, which has guaranteed convergence.

B. Estimating random systems

To better study the robustness of the method, we perform a simulation study using random systems. We generate 200 4th order discrete-time random systems using the MATLAB command `drss`, with the following two constraints: 1) each system must not have real valued negative poles (such systems are not physical, as they cannot be obtained by discretizing continuous-time systems); 2) the impulse response takes between 10 and 50 coefficients to settle. The reason for using a limited class of systems has to do with the choice of the high-order ARX model. In practice, this is not a very limiting assumption, as the user usually has an approximate idea of the speed of the system to identify. Also, the exact order is not very critical: in this simulation, we will use $n = 50$, even if some systems decay much faster. If slower systems were used, the order of the ARX model would have to be chosen larger. In Section VII, we discuss how future developments of recursive WNSF will automatically and adaptively choose this order.

In this simulation, the input is a GBN signal with average switching time equals to 5. Independent noise realizations are used for each of the random systems, with the noise variance relative to the variance of the noise-free output being 30%. As in the previous simulation, the data length is 3000.

In Fig. 3, we compare WNSF and PEM, both in off-line and recursive versions. Here, off-line WNSF and PEM

Fig. 3. FITs of estimates obtained with, from left to right, off-line PEM, recursive PEM, off-line WNSF, and recursive WNSF, using random systems and no initial model estimate.

perform similarly except for two outliers of PEM. This is not surprising, as according to the results in [11], WNSF can converge faster than PEM and be more robust against being trapped in non-global minima. However, the main observation is that, also with random systems, recursive WNSF performs similarly to its off-line version, while recursive PEM has worse performance due to convergence problems.

VII. DISCUSSION

In this paper, we proposed a recursive identification algorithm based on the WNSF method. In WNSF, the information contained in the data is condensed in a high-order ARX-model estimate and its covariance matrix. Then, the structured estimate of interest is obtained from the high-order estimate by weighted least squares. In order to guarantee an asymptotically efficient estimate, the weighting requires a consistent estimate of the structured model, which can be obtained with an additional least-squares step.

This theoretical analysis has been conducted in [12] for off-line WNSF. For recursive WNSF, the ARX-model related quantities are obtained recursively with recursive least squares; thus, they will converge to the same values as if obtained off-line. Because convergence of the low-order estimate depends only on the convergence of the high-order quantities, for fixed n the estimate of interest will also be guaranteed to converge to an estimate with the same properties as the one obtained off-line.

In this paper, we performed simulation studies to illustrate the performance of the method. We observed that recursive WNSF always performed similarly to its off-line version and to off-line PEM. Moreover, it is robust to initialization of the high-order model estimate and covariance, as recursive least-squares has guaranteed convergence. On the other hand, recursive PEM often failed to converge, unless initialized with an acceptable model estimate and a decent estimate of its covariance. However, even in this case, it did not attain the performance of recursive WNSF.

The price to pay for the improved convergence properties of recursive WNSF is a larger computational time. For example, in the simulation in 2, the average computational time per iteration was 2.4ms for recursive WNSF and 0.7ms for recursive PEM. This is a consequence of recursive WNSF requiring one additional step compared to recursive PEM: the high-order ARX-model estimate.

Despite the good performance, some improvements and extensions are of importance. Besides a theoretical analysis, the following are already in preparation.

1) *ARX-model order*: From theory, it is known that consistency and asymptotic efficiency are achieved only if the order of the ARX model tends to infinity as function of the sample size at a particular rate [12,13]. In practice, however, as was observed in the simulation with random systems, the order of the ARX model is not so critical, as long as it can reasonably capture the dynamics of the true system. Nevertheless, it makes sense that the recursive algorithm is able to adapt the order of the ARX model, increasing it as more data samples become available, if the increase provides an improvement in accuracy of the low-order model.

2) *Adaptation to Time-Varying Properties*: Online identification is often useful with systems that change over time. In this case, it is important that the recursive algorithm is adaptive and can capture the changing properties of the system. This can be done by including a forgetting factor in the high-order ARX-model estimate.

In conclusion, recursive WNSF is a promising algorithm for online identification of structured models, having the same convergence properties as its off-line version. This is a great improvement with respect to recursive PEM. To make sure the algorithm is competitive with state-of-the-art approaches, it is on the agenda to provide a detailed theoretical analysis, as well as to study the topics discussed in this section and perform more exhaustive numerical simulations.

REFERENCES

- [1] L. Ljung and T. Söderström, *Theory and Practice of Recursive Identification*. MIT Press, 1983.
- [2] P. C. Young, *Recursive Estimation and Time-Series Analysis*. Prentice-Hall, 1984.
- [3] B. Widrow and S. Stearns, *Adaptive Signal Processing*. Prentice-Hall, 1985.
- [4] N. Kaloupsides and S. Theodoridis, *Adaptive System Identification and Signal Processing Algorithms*. Prentice-Hall International, 1993.
- [5] V. Solo and X. Kong, *Adaptive Signal Processing Algorithms*. Prentice-Hall, 1995.
- [6] S. Haykin, *Adaptive Filter Theory*. Prentice-Hall, 1996.
- [7] L. Ljung, *System Identification. Theory for the User, 2nd ed.* Prentice-Hall, 1999.
- [8] —, “Analysis of a general recursive prediction error identification,” *Automatica*, vol. 17, no. 1, pp. 89–99, 1981.
- [9] L. Gerencsér, “Rate of convergence of recursive estimators,” *SIAM J. Control & Optimization*, vol. 30, no. 5, pp. 1200–1227, 1992.
- [10] P. Young and A. Jakeman, “Refined instrumental variable methods of recursive time-series analysis part I. single input, single output systems,” *Int. Journal of Control*, vol. 29, no. 1, pp. 1–30, 1979.
- [11] M. Galrinho, C. R. Rojas, and H. Hjalmarsson, “A weighted least-squares method for parameter estimation in structured models,” in *53rd IEEE Conference on Decision and Control*, 2014, pp. 3322–3327.
- [12] M. Galrinho, “Least squares methods for system identification of structured models,” Licentiate thesis, KTH Royal Institute of Technology, 2016.
- [13] L. Ljung and B. Wahlberg, “Asymptotic properties of the least-squares method for estimating transfer functions and disturbance spectra,” *Adv. Appl. Prob.*, vol. 24, pp. 412–440, 1992.
- [14] M. Galrinho, N. Everitt, and H. Hjalmarsson, “Arx modeling of unstable linear systems,” *Automatica*, vol. 75, pp. 167–171, 2017.
- [15] M. Galrinho, C. R. Rojas, and H. Hjalmarsson, “A weighted least squares method for estimation of unstable systems,” in *55th IEEE Conf. on Decision and Control*, 2016, pp. 341–346.