Recursive Estimation Algorithms in Matlab & Simulink Development Environment

PETR NAVRÁTIL, JÁN IVANKA

Department of Process Control, Department of Security Engineering
Tomas Bata University in Zlin
nám. T.G. Masaryka 5555, 760 01 Zlin
CZECH REPUBLIC
{p1navratil, ivanka}@fai.utb.cz

Abstract: - The article deals with recursive estimation algorithms realized in Matlab&Simulink development environment. These algorithms are realized as a blocks in simple SIMULINK library. Proposed library can be used for recursive parameter estimation of linear dynamic models ARX, ARMAX and OE. The library implements several recursive estimation methods: Least Squares Method, Recursive Leaky Incremental Estimation, Damped Least Squares, Adaptive Control with Selective Memory, Instrumental Variable Method, Extended Least Squares Method, Prediction Error Method and Extended Instrumental Variable Method. Several forgetting factor and modification of basic algorithm are taken into consideration in order to cope with tracking the time-variant parameters.

Key-Words: - Recursive estimation, ARX model, ARMAX model, OE model, forgetting factors, Matlab, Simulink

1 Introduction

The field of system identification uses statistical methods to build mathematical models of dynamical systems from measured data. A dynamical mathematical model in this context is a mathematical description of the dynamic behavior of a system or process in either the time or frequency domain. There exist many complex packages for system identification purposes in MATLAB and SIMULINK environment. These toolboxes provide solution to wide range of the problems from the area of system identification, e.g. System Identification Toolbox [11] and Continuous Identification Toolbox [6].

There also exist many special-purpose programs and libraries for MATLAB and SIMULINK, e.g. Idtool [3]. These simple tools provide solution to specific problems from the concrete part of the area of system identification. The proposed Recursive Identification Algorithms Library (RIA) fall into category of simple libraries for SIMULINK environment and is designed for recursive estimation of the parameters of the linear dynamic models ARX, ARMAX and OE. The Recursive Identification Algorithms Library consists of several user-defined blocks. These blocks implement several recursive identification algorithms: Least Squares Method (RLS) and its modifications, Recursive Leaky Incremental Estimation (RLIE), Damped Least Squares (DLS), Adaptive Control with Selective Memory (ACSM), Instrumental Variable Method (RIV), Extended Least Squares Method (ERLS), Prediction Error Method (RPEM) and Extended Instrumental Variable Method (ERIV). The Recursive Identification Algorithms Library can be used for simulation or real-time experiment (e.g. Real Time Toolbox) in educational process when it is possible to demonstrate the properties and behavior of the recursive identification algorithms and forgetting factors under various conditions and can be also used in the identification part of self-tuning controllers.

2 Model structure

The basic step in identification procedure is the choice of suitable type of the model. General linear model takes the following form:

$$y(k)=\frac{B(q^{-1})}{A(q^{-1})F(q^{-1})}u(k)+\frac{C(q^{-1})}{A(q^{-1})D(q^{-1})}n(k) \qquad (1)$$

where

$$\begin{aligned} &A\left(q^{-1}\right)=1+a_1q^{-1}+\ldots+a_{na}q^{-na} \\ &B\left(q^{-1}\right)=b_1q^{-1}+b_2q^{-2}+\ldots+b_{nb}q^{-nb} \\ &C\left(q^{-1}\right)=1+c_1q^{-1}+\ldots+c_{nc}q^{-nc} \\ &D\left(q^{-1}\right)=1+d_1q^{-1}+\ldots d_{nd}q^{-nd} \\ &F\left(q^{-1}\right)=1+f_1q^{-1}+\ldots+f_{nf}q^{-nf} \end{aligned} \quad (2)$$

are shift operator polynomials and $y(k)$, $u(k)$ are output and input signals. White noise $n(k)$ is assumed to have zero mean value and constant variance.

All linear models can be derived from general linear model by simplification. In the Recursive Identification Library following linear dynamic models are taken into consideration. These are ARX, ARMAX, OE models.

ARX model (*C*=*D*=*F*=1):

$$y(k)=\frac{B\left(q^{-1}\right)}{A\left(q^{-1}\right)}u(k)+\frac{1}{A\left(q^{-1}\right)}n(k) \quad (3)$$

ARMAX model (*D*=*F*=1):

$$y(k)=\frac{B\left(q^{-1}\right)}{A\left(q^{-1}\right)}u(k)+\frac{C\left(q^{-1}\right)}{A\left(q^{-1}\right)}n(k) \quad (4)$$

OE model (*A*=*C*=*D*=1):

$$y(k)=\frac{B\left(q^{-1}\right)}{F\left(q^{-1}\right)}u(k)+n(k) \quad (5)$$

3 Recursive Parameter Estimation

The recursive parameter estimation algorithms are based on the data analysis of the input and output signals from the process to be identified. Many recursive identification algorithms were proposed [4, 5]. In this part several recursive algorithms with forgetting factors implemented in Recursive Identification Algorithms Library are briefly summarized.

3.1 RLS

This method can be used for parameter estimate of ARX model. The algorithm can be written in following form:

$$\begin{aligned} &\hat{e}(k)=\boldsymbol{y}(k)-\boldsymbol{\phi}^T(k)\hat{\boldsymbol{\Theta}}(k-1) \\ &\boldsymbol{L}(k)=\frac{\boldsymbol{C}(k-1)\boldsymbol{\phi}(k)}{1+\boldsymbol{\phi}^T(k)\boldsymbol{C}(k-1)\boldsymbol{\phi}(k)} \\ &\hat{\boldsymbol{\Theta}}(k)=\hat{\boldsymbol{\Theta}}(k-1)+\boldsymbol{L}(k)\hat{e}(k) \\ &\boldsymbol{C}(k)=\boldsymbol{C}(k-1)-\boldsymbol{L}(k)\boldsymbol{\phi}^T(k)\boldsymbol{C}(k-1) \end{aligned} \quad (6)$$

where: $\boldsymbol{L}(k)$ denote gain matrix, $\boldsymbol{C}(k)$ is the covariance matrix of the estimated parameters, $\hat{\boldsymbol{\Theta}}(k)$ is the vector that contains the estimated parameters and $\boldsymbol{\phi}(k)$ is the data or regression vector

$$\hat{\boldsymbol{\Theta}}(k)=\left[a_1,\ldots a_{na},b_1,\ldots,b_{nb}\right]^T \quad (7)$$

$$\begin{aligned}\boldsymbol{\phi}^T(k)=\big[&y(k-1),\ldots,y(k-na),\\ &u(k-1),\ldots,u(k-nb)\big]\end{aligned} \quad (8)$$

This RLS algorithm assumes that the parameters of the model process are constant. In many cases, however, the estimator will be required to track changes in a set of parameters. To cope with tracking the time-variant parameters some adjustment mechanism must be introduced in the previous basic equations. Several implementations have been proposed [4, 9, 10, 16].

3.1.1 RLS with exponential forgetting

Covariance matrix is given by

$$\boldsymbol{C}(k)=\frac{1}{\lambda}\left(\boldsymbol{C}(k-1)-\frac{\boldsymbol{C}(k-1)\boldsymbol{\phi}(k)\boldsymbol{\phi}^T(k)\boldsymbol{C}(k-1)}{\lambda+\boldsymbol{\phi}^T(k)\boldsymbol{C}(k-1)\boldsymbol{\phi}(k)}\right) \quad (9)$$

where $0<\lambda<1$ is forgetting factor.

The algorithm is convenient for identification and adaptive control of slowly varying systems. This method has the main disadvantages that when the inputs is not persistent, and as the old data is discarded in the estimation procedure, the matrix $\boldsymbol{C}(k)$ increases exponentially with rate λ. This is called estimator wind-up.

3.1.2 RLS with variable exponential forgetting

The variable exponential forgetting is given by relation

$$\lambda(k)=\lambda_0\lambda(k-1)+1-\lambda_0 \quad (10)$$

with $\lambda(0)=\lambda_0 \in \langle 0{,}95;\, 0{,}99 \rangle$

This algorithm is convenient for identification of time-invariant systems and self-tuning controllers.

3.1.3 RLS with fixed directional forgetting

To solve the problem of estimator wind-up, an estimator with directional forgetting can be used. This estimator forgets the information only in the directions in which new information is gathered and assures the convergence of the estimations and avoids large changes in the parameters.

Covariance matrix is

$$\boldsymbol{C}(k)=\boldsymbol{C}(k-1)-\frac{\boldsymbol{C}(k-1)\boldsymbol{\phi}(k)\boldsymbol{\phi}^T(k)\boldsymbol{C}(k-1)}{\varepsilon^{-1}+\boldsymbol{\phi}^T(k)\boldsymbol{C}(k-1)\boldsymbol{\phi}(k)} \tag{11}$$

and directional forgetting factor

$$\varepsilon(k-1)=\lambda'-\frac{1-\lambda'}{\boldsymbol{\phi}^T(k)\boldsymbol{C}(k-1)\boldsymbol{\phi}(k)} \tag{12}$$

where λ' can be chosen as in exponential forgetting algorithm.

3.1.4 RLS with adaptive directional forgetting

Detailed description of this algorithm can be found in [8].

$$\varepsilon(k)=\varphi(k)-\frac{1-\varphi(k)}{\xi(k-1)} \tag{13}$$

where

$$\xi(k-1)=\boldsymbol{\phi}^T(k)\boldsymbol{C}(k-1)\boldsymbol{\phi}(k) \tag{14}$$

The value of adaptive directional forgetting factor is

$$\begin{aligned}\varphi(k)=\Big\{&1+(1+\rho)\big[\ln\big(1+\xi(k-1)\big)\big]+\\&+\left[\frac{\big(\upsilon(k-1)+1\big)\eta(k-1)}{1+\xi(k-1)+\eta(k-1)}-1\right]\frac{\xi(k-1)}{1+\xi(k-1)}\Big\}^{-1}\end{aligned} \tag{15}$$

and

$$\begin{aligned}\eta(k)&=\frac{\hat{e}^2(k)}{\lambda(k)}\\ \upsilon(k)&=\varphi(k)\big[\upsilon(k-1)+1\big]\\ \lambda(k)&=\varphi(k)\left[\lambda(k-1)+\frac{\hat{e}^2(k-1)}{1+\xi(k-1)}\right]\end{aligned} \tag{16}$$

3.1.5 RLS with exponential forgetting matrix

This technique is able to cope with the cases where parameters have distinct rates of change in time. Here, is described a recursive estimation algorithm with exponential forgetting matrix factors in order to provide distinct information discounts for each parameter. The RLS with exponential forgetting matrix is governed by the following equations [10]:

$$\begin{aligned}\Lambda(k-1)&=\boldsymbol{\Omega}\boldsymbol{C}(k-1)\boldsymbol{\Omega}^T\\ L(k)&=\frac{\Lambda(k-1)\boldsymbol{\phi}(k)}{1+\boldsymbol{\phi}^T(k)\Lambda(k-1)\boldsymbol{\phi}(k)}\\ \boldsymbol{C}(k)&=\Lambda(k-1)\left(\boldsymbol{I}-\frac{\boldsymbol{\phi}(k)\boldsymbol{\phi}^T(k)\Lambda(k-1)}{1+\boldsymbol{\phi}^T(k)\Lambda(k-1)\boldsymbol{\phi}(k)}\right)\end{aligned} \tag{17}$$

with

$$\boldsymbol{\Omega}=\begin{bmatrix}\frac{1}{\sqrt{\lambda_1}} & 0 & 0\\ 0 & \ddots & 0\\ 0 & 0 & \frac{1}{\sqrt{\lambda_1}}\end{bmatrix} \tag{18}$$

representing a matrix with diagonal elements equal to square roots of the forgetting factors associated to each column of the regression vector $\boldsymbol{\phi}$.

3.1.6 RLS with constant trace algorithm

Constant trace algorithm could also be used to keep the matrix $\boldsymbol{C}(k)$ limited; by scaling the matrix at each iteration in a way that trace of $\boldsymbol{C}(k)$ is constant. The regularized constant-trace algorithm is given by the following equations:

$$\hat{\boldsymbol{\Theta}}(k)=\hat{\boldsymbol{\Theta}}(k-1)+\boldsymbol{L}(k)\big(y(k)-\boldsymbol{\phi}^T(k)\hat{\boldsymbol{\Theta}}(k-1)\big) \tag{19}$$

$$\boldsymbol{L}(k)=\frac{\boldsymbol{C}(k-1)\boldsymbol{\phi}(k)}{\lambda+\boldsymbol{\phi}^T(k)\boldsymbol{C}(k-1)\boldsymbol{\phi}(k)} \tag{20}$$

$$\bar{\boldsymbol{C}}(k)=\frac{1}{\lambda}\left(\boldsymbol{C}(k-1)-\frac{\boldsymbol{C}(k-1)\boldsymbol{\phi}(k)\boldsymbol{\phi}^T(k)\boldsymbol{C}(k-1)}{1+\boldsymbol{\phi}^T(k)\boldsymbol{C}(k-1)\boldsymbol{\phi}(k)}\right) \tag{21}$$

$$\boldsymbol{C}(k)=c_1\frac{\bar{\boldsymbol{C}}(k)}{tr\big(\bar{\boldsymbol{C}}(k)\big)}+c_2\boldsymbol{I} \tag{22}$$

in which c_1 and c_2 have positive values given by,

$$\frac{c_1}{c_2}=10000,\;\; \boldsymbol{\phi}^T(k)\boldsymbol{\phi}(k)c_1 \gg 1 \tag{23}$$

3.1.7 Exponential Forgetting and Resetting Algorithm

This modification of RLS places upper and lower bounds on the trace of the covariance matrix while maintaining a robustly valued forgetting factor. The algorithm takes the following form:

$$\hat{\boldsymbol{\Theta}}(k)=\hat{\boldsymbol{\Theta}}(k-1)+\alpha\boldsymbol{L}(k)\hat{e}(k) \tag{24}$$

$$\boldsymbol{L}(k)=\frac{\boldsymbol{C}(k-1)\boldsymbol{\phi}(k)}{\lambda+\boldsymbol{\phi}^T(k)\boldsymbol{C}(k-1)\boldsymbol{\phi}(k)} \tag{25}$$

$$\begin{aligned}\boldsymbol{C}(k)=&\frac{1}{\lambda}\left[C(k-1)-\boldsymbol{L}(k)\boldsymbol{\phi}^T(k)C(k-1)\right]\\&+\beta\boldsymbol{I}-\gamma C(k-1)^2\end{aligned} \tag{26}$$

$$\sigma_{\min}\boldsymbol{I}\le C(k-1)\le\sigma_{\max}\boldsymbol{I}\qquad\forall k \tag{27}$$

$$\sigma_{\min}\approx\frac{\beta}{\alpha-\eta},\sigma_{\max}\approx\frac{\eta}{\gamma}+\frac{\beta}{\eta},\eta=\frac{1-\lambda}{\lambda}$$

$$\begin{aligned}&\alpha=0,5;\beta=\gamma=0,005;\lambda=0,95;\\&\sigma_{\min}=0,01;\sigma_{\max}=10\end{aligned} \tag{28}$$

3.2 RWLS

The recursive weighted least square [15] where the weighting data $\boldsymbol{\phi}(k)$ is denoted as $q(k)$ becomes

$$\begin{aligned}&\hat{e}(k)=\boldsymbol{y}(k)-\boldsymbol{\phi}^T(k)\hat{\boldsymbol{\Theta}}(k-1)\\&\boldsymbol{L}(k)=\frac{\boldsymbol{C}(k-1)\boldsymbol{\phi}(k)}{\boldsymbol{\phi}^T(k)\boldsymbol{C}(k-1)\boldsymbol{\phi}(k)+\dfrac{\lambda}{q(k)}}\\&\hat{\boldsymbol{\Theta}}(k)=\hat{\boldsymbol{\Theta}}(k-1)+\boldsymbol{L}(k)\hat{e}(k)\\&\boldsymbol{C}(k)=\boldsymbol{C}(k-1)-\boldsymbol{L}(k)\boldsymbol{\phi}^T(k)\boldsymbol{C}(k-1)\end{aligned} \tag{29}$$

where $0<\lambda<1$ is forgetting factor.

3.3 RLIE

With any parameter estimation algorithm, it is unavoidable that certain errors or noise will be present in the estimation loop. It can be shown when estimation algorithm contains an integrator in the estimation loop it may reduce its stability margin and accumulate the error effect, causing possible parameter estimate drift. To overcome this problem, it can be possible to modify the algorithm structure so that integral action is somewhat blunted. This can be achieved by introducing some leakage into the integration.

The recursive leaky incremental estimation [18] can be describes as follows:

$$\begin{aligned}&\hat{\boldsymbol{\Theta}}(k)=\gamma\hat{\boldsymbol{\Theta}}(k-1)+\Gamma\hat{\boldsymbol{\Theta}}(k)\\&\Gamma\hat{\boldsymbol{\Theta}}(k)=\Gamma\hat{\boldsymbol{\Theta}}(k-1)+\\&+\boldsymbol{C}(k)\boldsymbol{\phi}(k)\left(y(k)-\gamma\hat{\boldsymbol{\Theta}}(k-1)-\boldsymbol{\phi}^T(k)\Gamma\hat{\boldsymbol{\Theta}}(k-1)\right)\\&\boldsymbol{C}(k)=\frac{1}{\lambda}\left(\boldsymbol{C}(k-1)-\frac{\boldsymbol{C}(k-1)\boldsymbol{\phi}(k)\boldsymbol{\phi}^T(k)\boldsymbol{C}(k-1)}{\lambda+\boldsymbol{\phi}^T(k)\boldsymbol{C}(k-1)\boldsymbol{\phi}(k)}\right)\end{aligned} \tag{30}$$

where Γ denotes the stabilizing operator, defined as $\Gamma=1-\gamma q^{-1}$ and $\gamma\in[0,1]$ is the stabilizing parameter. This parameter is preselected by the user.

3.4 ACSM

Adaptive control with selective memory [7] updates parameter estimates only when there is new information present. The information increases and estimator eventually stops. The parameter estimates are updated only when the information matrix and/or estimated variance of the prediction error increases.

The algorithm consists of several steps, equations (31) - (34):

Step 0: Choose $r_0>0,\boldsymbol{\Theta}(0),\boldsymbol{C}(0)>0,1<M_0<\infty$

Set $r(0)=r_0>0,\sigma=1-\dfrac{1}{M_0},\varepsilon_0=\dfrac{1}{M_0}$.

Step 1:

$$r(k)=\max\left\{\sigma r(k-1)+(1-\sigma)\hat{e}(k-1)^2,r_0\right\} \tag{31}$$

Step 2: Set $B(k)=0$ and

$$A(k)=\begin{cases}1 & if\ \dfrac{\boldsymbol{\phi}^T(k)\boldsymbol{C}(k-1)\boldsymbol{\phi}(k)}{r(k)}\ge\varepsilon_0\\ 0 & otherwise\end{cases} \tag{32}$$

Step 3: If $A(k)=0$, set

$$B(k)=\begin{cases}1 & if\ r(k)\ge\max_{1\le i\le k}r(k-i)\\ 0 & otherwise\end{cases} \tag{33}$$

Set $\Delta(k)=A(k)+B(k)$

Step 4:

$$\begin{aligned}
&\hat{\boldsymbol{\Theta}}(k)=\hat{\boldsymbol{\Theta}}(k-1)+\\
&\quad+\Delta(k)\frac{\boldsymbol{C}(k-1)\phi(k)}{r(k)+\phi^T(k)\boldsymbol{C}(k-1)\phi(k)}\hat{e}(k)\\
&\boldsymbol{C}(k)=\boldsymbol{C}(k-1)-\\
&\quad-\Delta(k)\frac{\boldsymbol{C}(k-1)\phi(k)\phi^T(k)\boldsymbol{C}(k-1)}{r(k)+\phi^T(k)\boldsymbol{C}(k-1)\phi(k)}
\end{aligned} \tag{34}$$

Step 5: Set $k=k+1$ and go to step 1

3.5 ERIV

This method ensures improved accuracy and greater speed of convergence than RIV. The method is based on choice of instruments vector which has more elements than there are parameters in the model to be estimated. Derivation of this algorithm can be found in [16]. Instruments can be chosen according to [2], [16].

The set of equations describe this algorithm

$$\begin{aligned}
&\hat{\boldsymbol{\Theta}}(k)=\hat{\boldsymbol{\Theta}}(k-1)+\boldsymbol{L}(k)\left(\boldsymbol{v}(k)-\boldsymbol{\Phi}^T(k)\hat{\boldsymbol{\Theta}}(k-1)\right)\\
&\boldsymbol{L}(k)=\boldsymbol{P}(k-1)\boldsymbol{\Phi}(k)\left(\boldsymbol{\Lambda}(k)+\boldsymbol{\Phi}^T(k)\boldsymbol{P}(k-1)\boldsymbol{\Phi}(k)\right)^{-1}\\
&\boldsymbol{\Phi}(k)=\left[\boldsymbol{w}(k)\quad \boldsymbol{\phi}(k)\right]\\
&\boldsymbol{w}(k)=\boldsymbol{R}^T(k-1)\boldsymbol{z}(k)\\
&\Lambda(k)=\begin{bmatrix}-\boldsymbol{z}^T(k)\boldsymbol{z}(k) & 1\\ 1 & 0\end{bmatrix}\\
&\boldsymbol{v}(k)=\begin{bmatrix}\boldsymbol{z}^T(k)\boldsymbol{r}(k-1)\\ \boldsymbol{y}(k)\end{bmatrix}\\
&\boldsymbol{R}(k)=\boldsymbol{R}(k-1)+\boldsymbol{z}(k)\boldsymbol{\phi}^T(k)\\
&\boldsymbol{r}(k)=\boldsymbol{r}(k-1)+\boldsymbol{z}(k)\boldsymbol{y}(k)\\
&\boldsymbol{P}(k)=\boldsymbol{P}(k-1)-\boldsymbol{L}(k)\boldsymbol{\Phi}^T(k)\boldsymbol{P}(k-1)
\end{aligned} \tag{35}$$

3.6 RPEM

The recursive prediction error method (RPEM) allows the online identification of all linear model structure. Since all model structure except ARX are nonlinearly parameterized, no exact recursive algorithm can exist; rather some approximations must be made [13]-[16]. In fact, the RPEM can be seen as a nonlinear least squares Gauss-Newton method.

The Gauss-Newton technique is based on the approximation of the Hessian by the gradients. Thus, the RPEM requires the calculation of the gradient $\psi(k)$ of the model output with respect to its parameters:

$$\psi^T(k)=\frac{\partial\hat{y}(k)}{\partial\boldsymbol{\Theta}(k)}=\left[\frac{\partial\hat{y}(k)}{\partial\Theta_1(k)}\;\frac{\partial\hat{y}(k)}{\partial\Theta_2(k)}\;\cdots\;\frac{\partial\hat{y}(k)}{\partial\Theta_n(k)}\right] \tag{36}$$

RPEM algorithm takes the form

$$\begin{aligned}
&\hat{e}(k)=y(k)-\boldsymbol{\phi}^T(k)\hat{\boldsymbol{\Theta}}(k-1)\\
&\boldsymbol{L}(k)=\frac{\boldsymbol{P}(k-1)\boldsymbol{\psi}(k)}{1+\boldsymbol{\psi}^T(k)\boldsymbol{P}(k-1)\boldsymbol{\psi}(k)}\\
&\hat{\boldsymbol{\Theta}}(k)=\hat{\boldsymbol{\Theta}}(k-1)+\boldsymbol{L}(k)\hat{e}(k)\\
&\boldsymbol{P}(k)=\boldsymbol{P}(k-1)-\boldsymbol{L}(k)\boldsymbol{\psi}^T(k)\boldsymbol{P}(k-1)
\end{aligned} \tag{37}$$

where $\boldsymbol{P}(k)$ denotes covariance matrix.

The model structure will influence the way in which the quantities $\hat{e}(k)$ and $\psi(k)$ in the algorithm are computed from data and the previously computed parameter estimate.

3.7 RELS

This method is used for parameter estimations of ARMAX model. Formally it takes the same form as RLS. However, the regression and parameter vector are different.

Parameter vector

$$\hat{\boldsymbol{\Theta}}(k)=\left[a_1,\dots a_{na},b_1,\dots,b_{nb},c_1,\dots,c_{nc}\right]^T \tag{38}$$

Regression vector

$$\begin{aligned}
\boldsymbol{\phi}^T(k)=\big[&y(k-1),\dots,y(k-na),\\
&u(k-1),\dots,u(k-nb),\\
&\eta(k-1),\dots,\eta(k-nc)\big]
\end{aligned} \tag{39}$$

or

$$\begin{aligned}
\boldsymbol{\phi}^T(k)=\big[&y(k-1),\dots,y(k-na),\\
&u(k-1),\dots,u(k-nb),\\
&\hat{e}(k-1),\dots,\hat{e}(k-nc)\big]
\end{aligned} \tag{40}$$

where $\eta(k)$ denotes the residual and $\hat{e}(k)$ is the prediction error. It usually speeds up the convergence of the RELS algorithm if the residuals (*a posteriori*) rather than the prediction errors (*a priori*) are used.

3.8 DSL

Damped least squares (DLS) algorithm is an extended version of the recursive simple least squares (RLS) algorithm [12]. The DSL algorithm is more appropriate for adaptive control, since it weights increments of estimated parameter vector (41). This gives more control on the adaptation rate.

The DLS criterion is

$$J\left(\hat{\boldsymbol{\Theta}}\right)=\sum_{k=t-N}^{t}\lambda^{t-k}\left(\begin{array}{l}\left[y(k)-\phi^{T}(k)\hat{\boldsymbol{\Theta}}(k)\right]^{2}+\\+\left[\boldsymbol{\Lambda}_{d}(k)\left(\hat{\boldsymbol{\Theta}}(k)-\hat{\boldsymbol{\Theta}}(k-1)\right)\right]^{2}\end{array}\right) \tag{41}$$

The weighting matrix $\boldsymbol{\Lambda}_d(k)$ is diagonal and weights the parameters variations. For an n-parameters model,

$$\boldsymbol{\Lambda}_d(k-1)=diag\left[\alpha_1(k)\ \alpha_2(k)\ldots\alpha_n(k)\right] \tag{42}$$

A standard form of the DLS algorithm is given

$$\begin{aligned}
\hat{\boldsymbol{\Theta}}(k)&=\hat{\boldsymbol{\Theta}}(k-1)+\boldsymbol{L}\left[y(k)-\hat{\boldsymbol{\Theta}}^{T}(k)\boldsymbol{\phi}(k)\right]+\\
&\quad+\boldsymbol{C}(k)\lambda(k)\boldsymbol{\Lambda}_{d}(k)\left[\hat{\boldsymbol{\Theta}}(k-1)-\hat{\boldsymbol{\Theta}}(k-2)\right]\\
\boldsymbol{L}(k)&=\frac{\boldsymbol{C}(k-1)\boldsymbol{\phi}(k)}{\lambda(k)+\boldsymbol{\phi}^{T}(k)\boldsymbol{C}(k-1)\boldsymbol{\phi}(k)}\\
\boldsymbol{C}(k)&=\frac{1}{\lambda(k)}\left(\boldsymbol{C}'(k-1)-\right.\\
&\quad\left.-\frac{\boldsymbol{C}'(k-1)\boldsymbol{\phi}(k)\boldsymbol{\phi}^{T}(k)\boldsymbol{C}'(k-1)}{\lambda+\boldsymbol{\phi}^{T}(k)\boldsymbol{C}'(k-1)\boldsymbol{\phi}(k)}\right)\\
\boldsymbol{C}'(k)&=\boldsymbol{C}'(k-1)-\\
&\quad-\sum_{i=1}^{n}\frac{\boldsymbol{C}'_{i-1}(k-1)r_i r_i^{T}\boldsymbol{C}'_{i-1}(k-1)\alpha'_i(k)}{1+r_i^{T}\boldsymbol{C}'_{i-1}(k-1)r_i\alpha'_i(k)}\\
\boldsymbol{C}'_1(k-1)&=\boldsymbol{C}'(k-1)-\\
&\quad-\frac{\boldsymbol{C}'_{i-1}(k-1)r_i r_i^{T}\boldsymbol{C}'_{i-1}(k-1)\alpha'_i(k)}{1+r_i^{T}\boldsymbol{C}'_{i-1}(k-1)r_i\alpha'_i(k)}\\
\boldsymbol{C}'_0(k-1)&=\boldsymbol{C}(k-1)\\
\alpha'_i(k)&=\frac{\alpha_i(k)-\lambda(k)\alpha_i(k-1)}{\lambda(k)}
\end{aligned} \tag{43}$$

where r_i are the succesive basic vectors, e.g. $r_i=\left[1..0\cdots0\right]^T$

3.9 RIV

It can be shown that if the process does not meet the noise assumption made by the ARX model, the parameters are estimated biased and non-consistent. This problem can be avoided using instrumental variable method.

The algorithm takes the form

$$\begin{aligned}
\hat{e}(k)&=\boldsymbol{y}(k)-\boldsymbol{\phi}^{T}(k)\hat{\boldsymbol{\Theta}}(k-1)\\
\boldsymbol{L}(k)&=\frac{\boldsymbol{C}(k-1)\boldsymbol{z}(k)}{1+\boldsymbol{\phi}^{T}(k)\boldsymbol{C}(k-1)\boldsymbol{z}(k)}\\
\hat{\boldsymbol{\Theta}}(k)&=\hat{\boldsymbol{\Theta}}(k-1)+\boldsymbol{L}(k)\hat{e}(k)\\
\boldsymbol{C}(k)&=\boldsymbol{C}(k-1)-\boldsymbol{L}(k)\boldsymbol{\phi}^{T}(k)\boldsymbol{C}(k-1)
\end{aligned} \tag{44}$$

where: $\boldsymbol{L}(k)$ denote gain matrix, $\boldsymbol{C}(k)$ is the covariance matrix of the estimated parameters, $\hat{\boldsymbol{\Theta}}(k)$ is the vector that contains the estimated parameters, $\boldsymbol{\phi}(k)$ is the data or regression vector, $\boldsymbol{z}(k)$ is instrumental variable

$$\hat{\boldsymbol{\Theta}}(k)=\left[a_1,\ldots a_{na},b_1,\ldots,b_{nb}\right]^T \tag{45}$$

$$\begin{aligned}\boldsymbol{\phi}^{T}(k)=\big[&y(k-1),\ldots,y(k-na),\\&u(k-1),\ldots,u(k-nb)\big]\end{aligned} \tag{46}$$

Choice of instrumental variable determines behaviour of the IV method in usage. Some common choices for generating instruments are proposed in [2, 16].

Typical choice of model independent instrumental variable is

$$\boldsymbol{z}(k)=\left[u(k-1),\ldots,u(k-na-nb)\right]^T \tag{47}$$

and model dependent instrument is

$$\begin{aligned}z(k)=\big[&y_u(k-1),\ldots,y_u(k-na),\\&u(k-1),\ldots,u(k-nb)\big]^T\end{aligned} \tag{48}$$

where $y_u(k-1)$ is generated by calculating following difference equation with current parameter estimates

$$y_u(k)=\hat{b}_1(k)u(k-1)+\ldots+\hat{b}_{nb}(k)u(k-nb)- \\ -\hat{a}_1(k)y_u(k-1)+\ldots+\hat{a}_{na}(k)y_u(k-na) \qquad (49)$$

4 Recursive Identification Algorithms Library (RIA)

The Recursive Identification Algorithm Library is realized in Matlab&Simulink environment. The proposed library is designed for recursive parameter estimation of linear dynamics model ARX, ARMAX, OE using recursive identification methods mentioned in previous chapters.

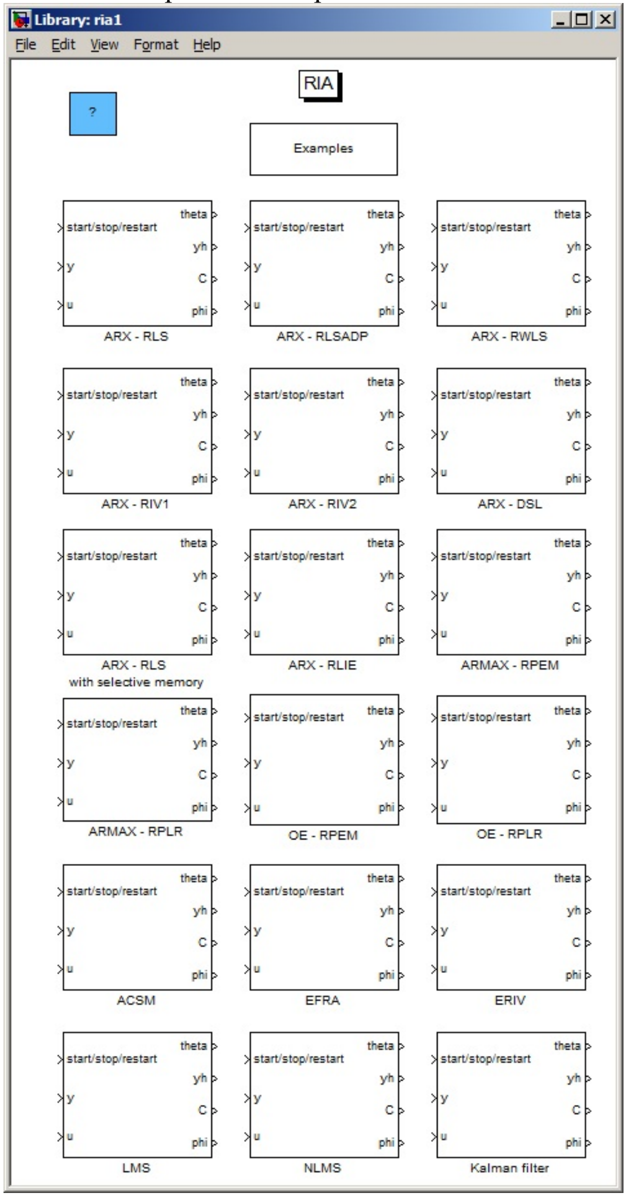
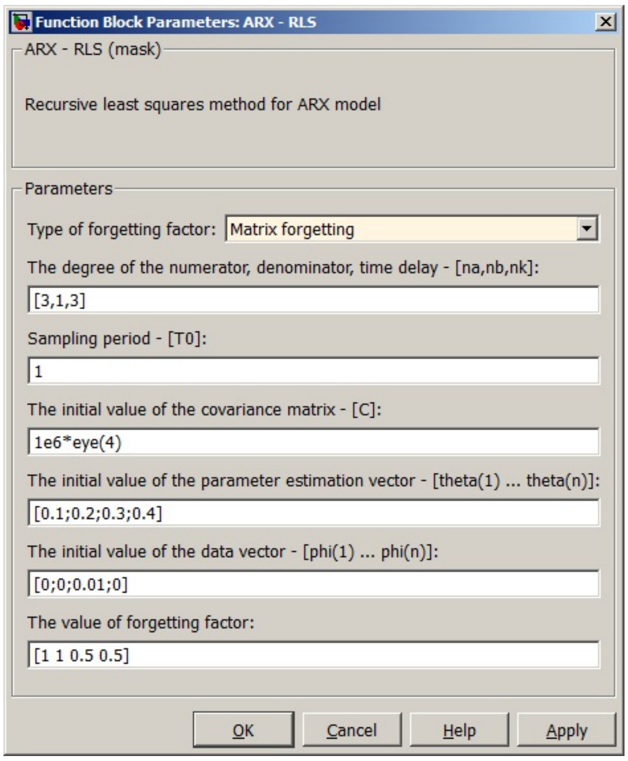


Fig.1 Recursive identification Algorithms Library

The Recursive Identification Algorithm Library is depicted in Fig.1. The Library consists of 18 user-defined blocks and is designed for MATLAB&SIMULINK environment. Each block is realized as an s-function.

Each block is masked by user-defined dialog. Several necessary input parameters should be input through this dialog. These are: type of forgetting factor and its value, degrees of polynomials, sampling period, initial values of parameter estimate, covariance matrix and data vector, etc. Each block also contains the help describes the meaning of each parameter, inputs and outputs and used recursive identification algorithms. Example of input dialog is shown in Fig.2.



Function Block Parameters: ARX - RLS

ARX - RLS (mask)

Recursive least squares method for ARX model

Parameters

Type of forgetting factor: Matrix forgetting

The degree of the numerator, denominator, time delay - [na,nb,nk]: [3,1,3]

Sampling period - [T0]: 1

The initial value of the covariance matrix - [C]: 1e6*eye(4)

The initial value of the parameter estimation vector - [theta(1) ... theta(n)]: [0.1;0.2;0.3;0.4]

The initial value of the data vector - [phi(1) ... phi(n)]: [0;0;0.01;0]

The value of forgetting factor: [1 1 0.5 0.5]

Fig.2 Input dialog of the identification block

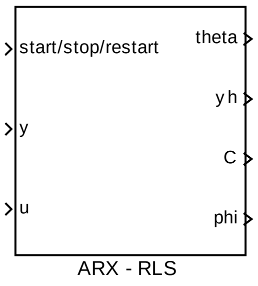


Fig.3. Inputs/outputs of the identification block

Input/output data from object under identification process are inputs to the identification block. Another input (start/stop/restart) is used for control the identification algorithm. This input provides possibility of start, stop and restart the identification algorithm in selected instant of time. Outputs of the block are estimate of parameter vector, one-step prediction of output of model, covariance matrix and data vector. The inputs and outputs of the block are shown in Fig.3.

4.1 Examples

Using the recursive identification algorithms library is illustrated in two examples. The first example shows the simple application of the identification block in the model. Simulink diagram and the results are shown in Fig4.

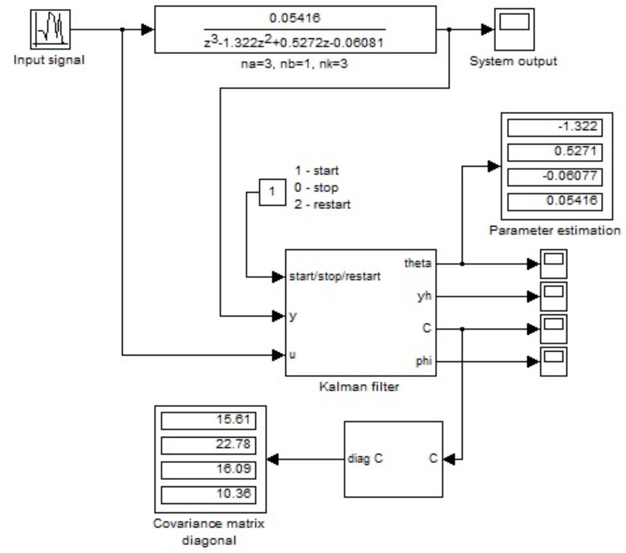


Fig.4 Example of application of identification block

Second example shows application of recursive identification algorithms library in identification part of self-tuning controller.

Several recursive identification algorithms mentioned above were tested in closed loop on system given by transfer function (50) with self-tuning LQ controller. The controller is based on minimization of quadratic criterion with controller output signal penalization. The minimization of quadratic criterion is realized by spectral factorization.

Continuous-time transfer function of the controlled system is

$$G(s)=\frac{s+1}{50s^2+15s+1},\quad \textit{for } t\le 1000s$$
$$G(s)=\frac{s+1{,}5}{50s^2+15s+1},\quad \textit{for } t\ge 1000s \qquad (50)$$

The block diagram of controlled system is shown in Fig.5. It can be seen that the system output is directly influenced by non-measurable disturbance. This case is commonly fulfilled in practice.

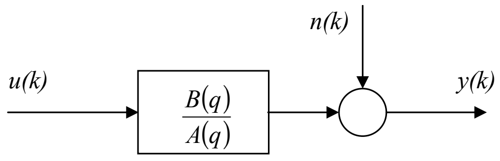


Fig.5 Block diagram of controlled system

Discrete transfer function is then

$$G\left(z^{-1}\right)=\frac{0.1529\mathrm{z}^{-1}+0.0287\mathrm{z}^{-2}}{1\text{-}1.1196\,\mathrm{z}^{-1}+0.3012\mathrm{z}^{-2}},\quad \textit{for } k\le 250$$
$$G\left(z^{-1}\right)=\frac{0.2072\mathrm{z}^{-1}+0.0651\mathrm{z}^{-2}}{1\text{-}1.1196\,\mathrm{z}^{-1}+0.3012\mathrm{z}^{-2}\,2},\quad \textit{for } k\ge 250 \qquad (51)$$

The sampling period was chosen $T_0=4s$.

The same initial conditions for system identification were used for all the types of recursive algorithms we tested. The initial parameter estimates were chosen to be

for ARX, OE model

$$\hat{\boldsymbol{\Theta}}(k)=[0,0,0,0]^T \qquad (52)$$

for ARMAX model

$$\hat{\boldsymbol{\Theta}}(k)=[0,0,0,0,0,0]^T \qquad (53)$$

Estimation algorithms with fixed and variable exponential forgetting were applied using a forgetting factor $\lambda=\lambda(0)=0.985$. Forgetting factor for fixed directional forgetting was set to $\lambda'=0.985$. Initial values for adaptive forgetting were chosen to be $\varphi(0)=1,\ \rho(0)=0.99,\ \upsilon(0)=10^{-6},\ \lambda(0)=0.001$

System dynamics were described by ARX, OE, ARMAX model, respectively. Parameters of ARX model were identified by RLS, RIV and ERIV methods, RELS and RPEM methods were used to parameter estimation of ARMAX model and parameters of OE model were estimated by RPEM method. To assure parameters tracking the forgetting factors were used. Only parameters of deterministic part of the estimated models were utilized for controller synthesis.

4.1.1 Estimation of ARX model

RLS with variable exponential forgetting

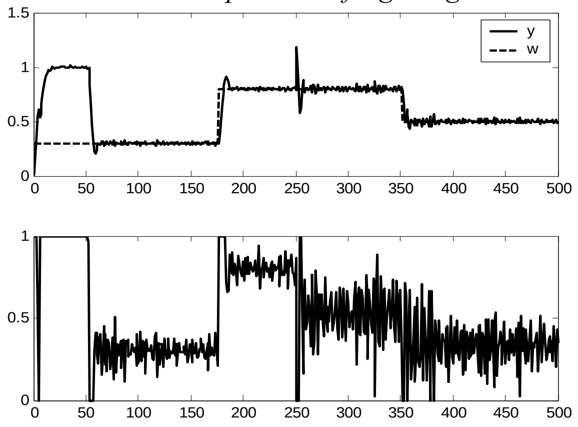


Fig.6 The adaptive control with LQ controller

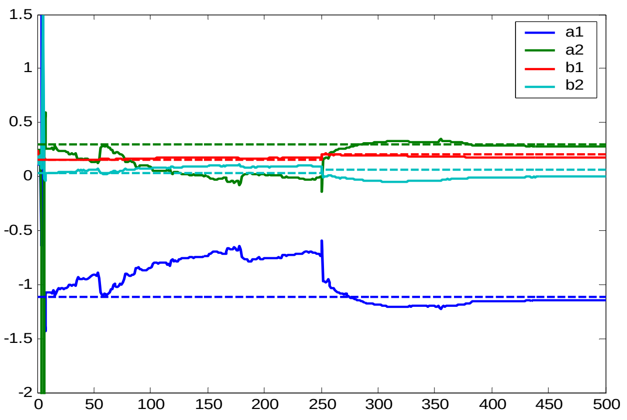


Fig.7 Parameter estimates (solid line) computed with the RLS with variable exponential forgetting – true parameters (dashed line)

RLS with adaptive directional forgetting

From Fig.8 and Fig.9 can be seen that adaptive directional forgetting can improve control quality but there is fluctuation in parameters.

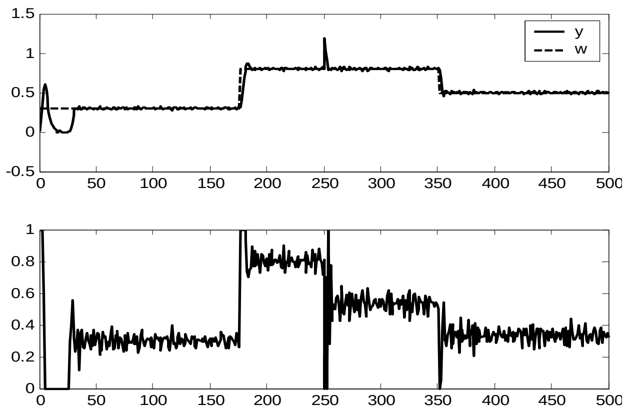


Fig.8 The adaptive control with LQ controller

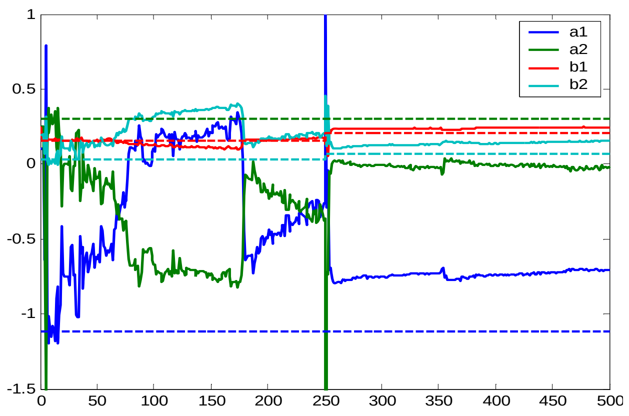


Fig.9. Parameter estimates (solid line) computed with the RLS with adaptive directional forgetting – true parameters (dashed line)

RIV with adaptive directional forgetting

Fig.10 shows that the adaptive control with RIV method provide better results from system output point of view than adaptive control with RLS method.

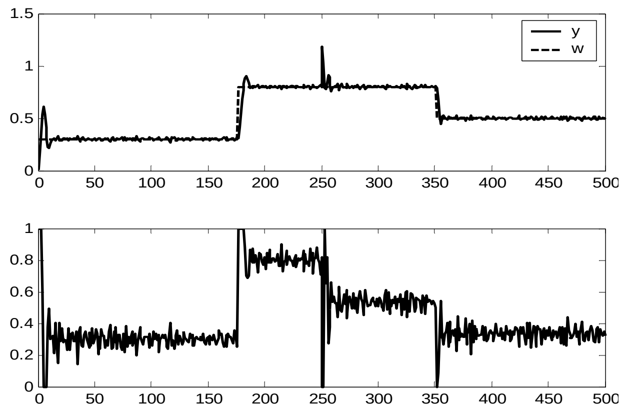


Fig.10. The adaptive control with LQ controller

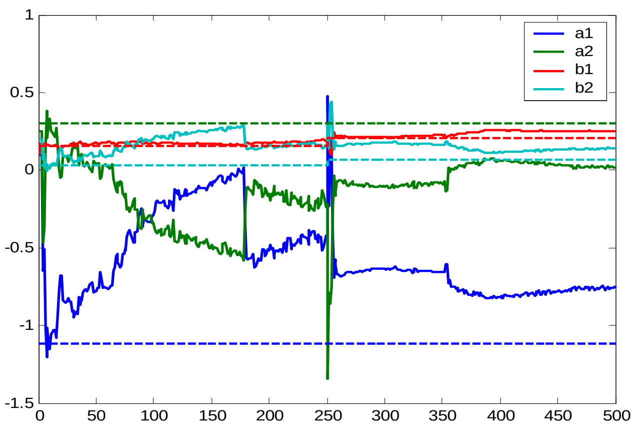


Fig.11. Parameter estimates (solid line) computed with the RIV with adaptive directional forgetting – true parameters (dashed line)

ERIV with fixed exponential forgetting

Fig.13 shows that despite the fact that parameters are estimated correctly the adaptive controller does not provide appropriate output signal. From Fig.14 can be seen that the speed of convergence is faster that in RLS but the estimator is not able to track changes in parameters.

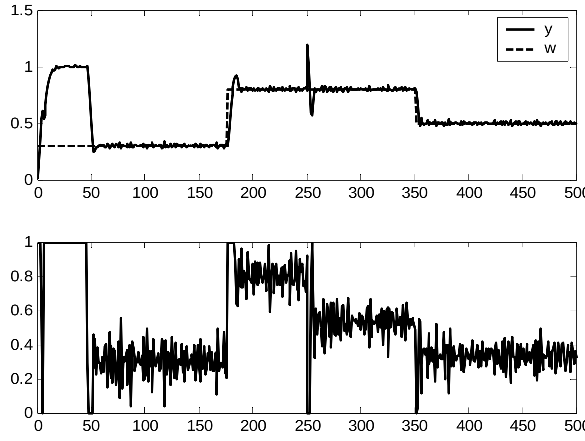


Fig.12 The adaptive control with LQ controller

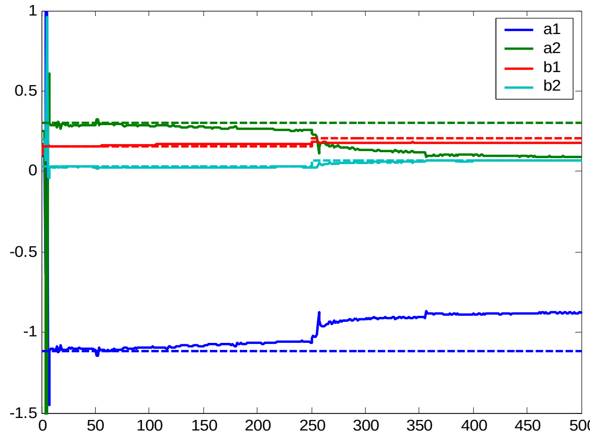


Fig.13 Parameter estimates (solid line) computed with the ERIV with fixed exponential forgetting - true parameters (dashed line)

4.1.2 Estimation of ARMAX model

RPEM-ARMAX with fixed exponential forgetting

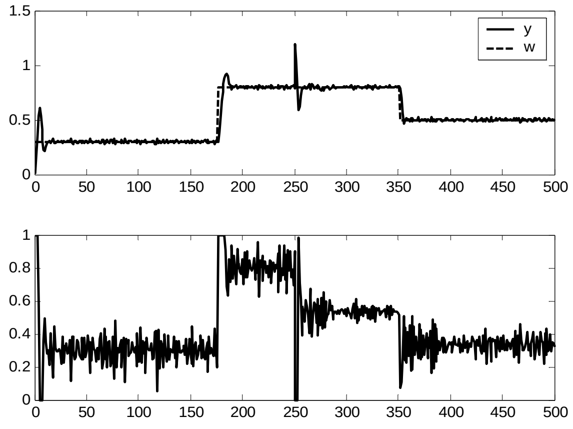


Fig.14 The adaptive control with LQ controller

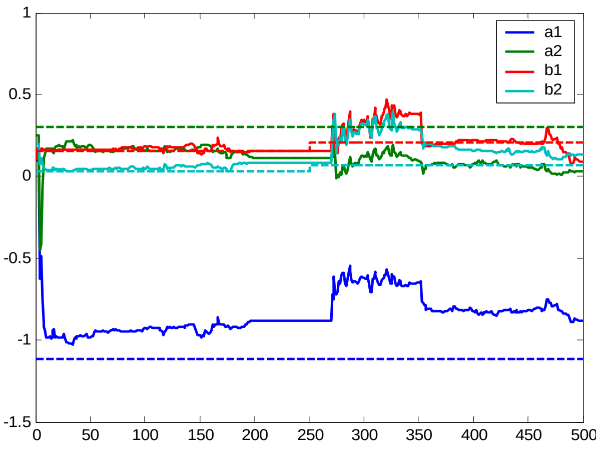


Fig. 15 Parameter estimates (solid line) computed with the RPEM-ARMAX with fixed exponential forgetting – true parameters (dashed line)

From step k=200 to k=270 the parameters estimates is maintained constant and in step k=270 the restart of covariance matrix is made. This setting improves the adaptive controller behaviour.

RELS with adaptive directional forgetting

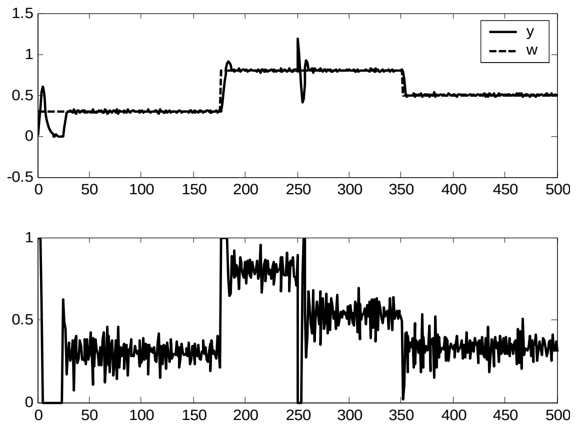


Fig.16 The adaptive control with LQ controller

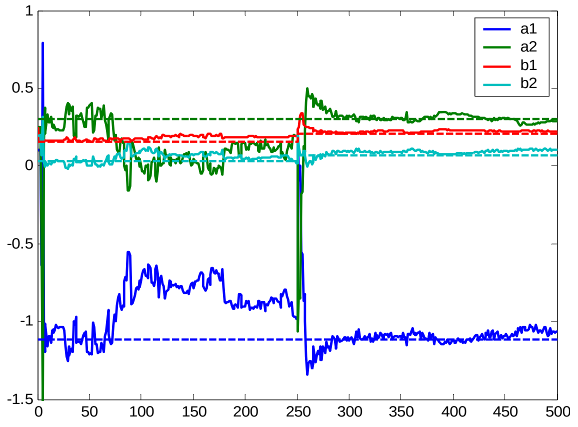


Fig.17 Parameter estimates (solid line) computed with the RELS with adaptive directional forgetting – true parameters (dashed line)

4.1.2 Estimation of OE model

RPEM-OE with fixed exponential forgetting

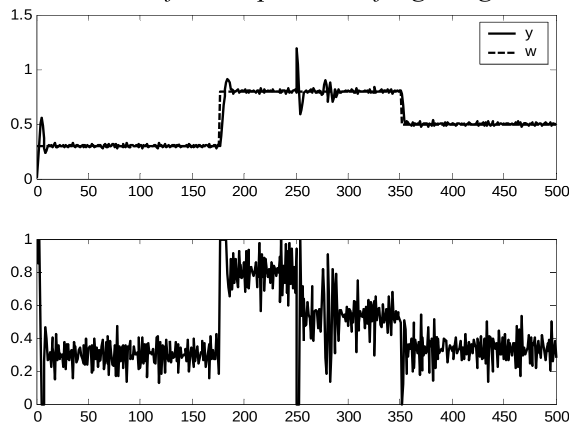


Fig.18 Simulation results: the adaptive control with LQ controller

From step k=200 to k=270 the parameters estimates is maintained constant and in step k=270 the restart of covariance matrix is made. This setting improves the adaptive controller behaviour.

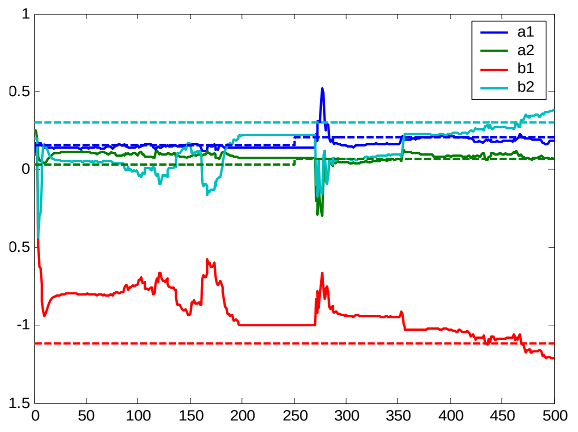


Fig.19 Parameter estimates (solid line) computed with the RPEM-OE with fixed exponential forgetting – true parameters (dashed line)

Influence of each recursive algorithms on adaptive control performance were evaluated from quality control point of view. The results can be seen in Table 1.

The criteria were defined

$$S_y = \frac{1}{k_2 - k_1 + 1} \sum_{k=k_1}^{k_2} e^2(k); \qquad (54)$$

$$S_u = \frac{1}{k_2 - k_1 + 1} \sum_{k=k_1}^{k_2} \Delta u^2(k) \qquad (55)$$

where $e(k)$ denotes control error, $u(k)$ is controller output and $k_1 = 1, \quad k_2 = 500$.

Methods	S_y	S_y
RLS with VEF	0.0460	0.0429
RLS with ADF	0.0054	0.0146
ERIV with FEF	0.0404	0.0276
RIV with ADF	0.0025	0.0126
RPEM-ARMAX with FEF	0.0027	0.0205
RELS with ADF	0.0058	0.0207
RPEM-OE with FEF	0.0026	0.0294

Table 1 Influence of recursive algorithm on adaptive control performance

From Table 1 can be seen that the minimum of sum of squared control error and minimum of sum of squared difference of controller output signal were achieved by RIV method with adaptive directional forgetting for parameter estimate of ARX model. Other methods listed in Table 1 also provide satisfactory results except RLS with variable exponential forgetting.

The results in Table 1 can be expressed in more transparent form. Fig.21 and Fig.22 show the control quality results in graphic form.

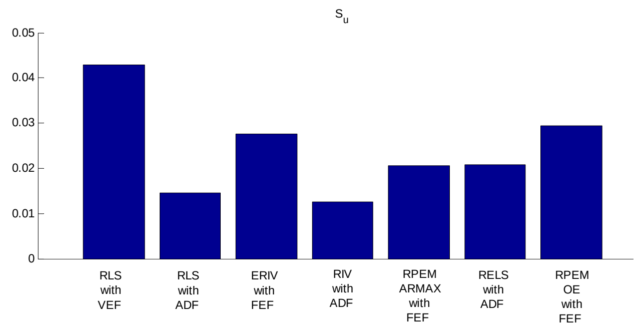


Fig.20 The criterion control quality S_u in graphic representation

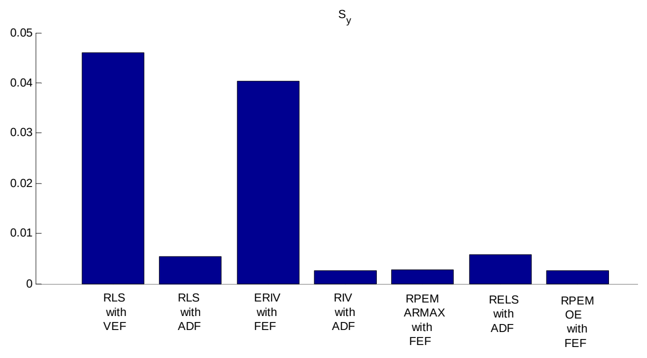


Fig.21 The criterion control quality S_y in graphic representation

5 Conclusion

The Recursive Identification Algorithm Library is designed for recursive parameter estimation of linear dynamic model ARX, ARMAX, OE using recursive identification methods. The simple library can be used e.g. in identification part of self-tuning controller or in educational process when it is possible to demonstrate the properties and behavior of the recursive identification algorithms and forgetting factors under various conditions. Proposed library can be used not only in educational process to demonstrate the behavior and properties of recursive identification algorithms, but for example in connection with such Real Time Toolbox to identify in real time the parameters of the model of real systems.

References

[1] Bobál, V.; J. Böhm, fessl, J.; macháček, J. *Digital Self-tuning Controllers: Algorithms, Implementation and Applications,* Springer-Verlag, London, 2005. ISBN 1-85233-980-2.

[2] Branica, I.; Peric, N.; Petrovic, I., Comparison of Several Recursive Identification Methods. *Automatika*, Vol. 37, No. 3-4, 1996, pp. 99-104.

[3] Cirka, L.; Fikar, M. *Identification tool for Simulink,* Department of Process Control, FCT STU, Technical Report KAMF9803, 1998.

[4] Corriou, J. P. Process Control : *Theory and Applications,* Springer-Verlag, London, 2004.

[5] Cunha, J. B.; Oliviera, P.B.; Coelho, J.P., Recursive Parameter Estimation Algorithms, *In Controlo 2004: 6th Portuguese Conference on Automatic Control,* pp. 109-114, 2004.

[6] Garnier, H., *Continuous-Time System Identification Toolbox,* [online]. <http://www.iris.cran.uhp-nancy.fr/contsid/>, 2006.

[7] Hill, J.H.; Ydstie, B.E., Adaptive control with selective memory, *International Journal of Adaptive Control and Signal Processing*, Vol. 18, No 7, 2004, pp. 571-587.

[8] Kulhavý, R., Restricted exponential forgetting in real time identification, *Automatica*, Vol. 23, 1987, pp. 586-600.

[9] Kulhavý, R.; Zarrop, M.B., On a general concept of forgetting, *International Journal of Control*, Vol.58, No.4, 1993, pp. 905–924.

[10] Ljung, L., *System identification – theory for user*, Prentice-Hall, Englewood Cliffs, N.J., 1987. ISBN 0-13-881640-9

[11] Ljung, L., *System Identification Toolbox: For use with MATLAB,* Prentice-Hall, Englewood Cliffs, N.J., 2004.

[12] Lambert, E. P., *Process control applications of long-range prediction*, DPhill thesis, University of Oxford, 1987.

[13] Moore, J. B.; Boel, R. K. Asymptotically Optimum Recursive Prediction Error Methods in Adaptive Estimation and Control. *Automatica*, Vol.22, No.2, 1986, pp. 237-240.

[14] Moore, J. B.; Weiss, H., Recursive Prediction Error Methods for Adaptive Estimation, *IEEE Trans. on Systems, Man, and Cybernetics,* Vol.SMC-9, No.4, 1979, pp. 197-205.

[15] Nelles, O., *Nonlinear system identification*. Springer-Verlag, Berlin, 2001.

[16] Söderström, T.; Stoica, P., *System Identification,* Prentice Hall, University Press, Cambridge, UK., 1989, ISBN 0-13-881236-5.

[17] Wellstead, P.E.; Zarrop, M.B., *Self-Tuning System – Control and Signal Processing*, John Wiley & Sons Ltd. Chichester.

[18] Zhou, Q.; Cluett, W.R., Recursive Identification of Time-varying Systems via Incremental Estimation. *Automatica*, Vol.32, No.10, 1996, pp 1427-1431.