JAIC
Journal of Applied Instrumentation and Control

Control of a Thermal Airflow Process with Time-Delay - Part I: System Identification

Sidney A. A. Viana

***Abstract* — This article was motivated from a practical work on modeling and control of a time-delayed thermal airflow process using adaptive techniques. The work was divided into two parts: (I) the modeling of the process using system identification methods, with main concerns to the numerical robustness of the identification, and (II) the digital control of the process using adaptive self-tuning control, with main concerns to the adaptation of the controller to changes in the process dynamics. This article presents the first part of the work. The thermal airflow system was represented by an ARMAX model, whose parameters were identified using the Recursive Least Squares method, based on two approaches: the Matrix Inversion Lemma, and the Bierman's UD Factorization. The results obtained show that the last approach has greater numerical robustness and is more suitable for applications of adaptive control – the second part of the work, described in a separate article.**

***Index Terms* — least squares, parameter estimation, recursive identification, system identification.**

I. Introduction

Many problems on the design of automatic controllers for dynamic systems involve the determination of a suitable mathematical model for the system to be controlled. For digital control applications, a discrete-time model can be obtained either from the discretization of an available continuous-time model or from the direct determination of a discrete model by means of *system identification* or *parameter estimation* methods. Identification means the estimation of the parameters of a given discrete-time model, and can be implemented by either off-line (batch) or on-line (recursive) calculations. In the **off-line identification**, a set of input-output values (observations) from the system is measured along a suitable period of time, and then used as a whole data set in a batch calculation to estimate the parameters of a given model structure. In the **on-line identification**, also known as real-time identification, the model parameters are continuously estimated in a recursive fashion, by which the results computed at the previous sample time are used to compute the results for the current sample time. Recursive identification is particularly advantageous to track eventual changes on the system parameters, allowing an adaptive tuning of digital controllers.

Mathematically, parameter identification is set of mathematical computations that intends to determine, under a certain optimality criterion, the values of model parameters that better match the input-output observations from the system. Recursive identification has fundamental importance in adaptive control, by which the *controller parameters* are automatically adjusted due to changes in the *process parameters*, provided that these changes are tracked by on-line identification. An interesting review of system identification methods can be found in [3].

This article describes the identification of an **ARMAX** (**A**uto-**R**egressive **M**oving **A**verage with E**x**ogenous Input) model for a thermal airflow system with time-delay. The identification was performed with the Least Squares method, based on two approaches: the Matrix Inversion Lemma [4][6][12], and the Bierman's UD Factorization [4], with a comparison of the effectiveness of the two implementations.

The article is organized as follows: Section II describes the mathematical formulation of the Least Squares method in its non-recursive (off-line) and recursive (on-line) forms; Section III cares about the generation of suitable input signals for the system to be identified, in order to meet the requirement of *persistent excitation*. Section IV presents the model structure chosen for the process, and the results of the model identification using the Least Squares method. Finally, Section V summarizes the conclusions about this first part of the work, and its relationship with the second part.

II. The Least Squares Method for System Identification

A. Non-Recursive Form

The mathematical formulation of the Least Squares method was originally developed by Karl Friedrich Gauss for a problem of estimating the orbit of the Ceres asteroid [7]. Later, the method was extended to other kinds of estimation

S. A. A. Viana is a Senior Member of the IEEE – The Institute of Electrical and Electronics Engineers. He is currently with the Ferrous Automation Engineering Department of VALE, Belo Horizonte, MG, Brazil (e-mail: sidney.viana@vale.com).

DOI: ISSN:

problems.

For purposes of system identification, let us consider a simple **ARX** (**A**uto-**R**egressive with E**x**ogenous Input) model:

$$G(z)=\frac{Y(z)}{U(z)}=z^{-k}\frac{B(z^{-1})}{A(z^{-1})}=z^{-k}\frac{\sum_{j=0}^{n_b}b_j z^{-j}}{1+\sum_{j=1}^{n_a}a_j z^{-j}} \qquad (1)$$

$$=z^{-k}\frac{b_0+b_1z^{-1}+b_2z^{-2}+\ldots+b_{n_b}z^{-n_b}}{1+a_1z^{-1}+a_2z^{-2}+\ldots+a_{n_a}z^{-n_a}}$$

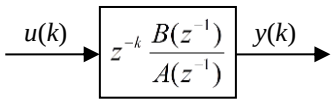

Fig. 1. Block diagram of an ARX model.

The discrete-time delay exponent k is a positive integer (k = 1, 2, 3, …), and its value depends on the sampling period T of the model. Without lack of generality, we can set k = 1. Cases in which k >1 can be considered similarly, increasing the order of the polynomial $B(z^{-1})$ by k –1 and assuming the first k–1 coefficients b_j are identically null. From the inverse Z transform of $Y(z)$ in (1), with k =1, the system output $y(k)$ at the discrete time $t = kT$ is:

$$y(k)=-a_1y(k-1)-a_2y(k-2)-\ldots-a_{n_a}y(k-n_a)+ \\ +b_0u(k-1)+b_1u(k-2)+\ldots+b_0u(k-n_b-1) \qquad (2a)$$

Similarly, the system output for past discrete times $t=(k-1)T,(k-2)T,\ldots,(k-N)T$ are:

$$y(k-1)=-a_1y(k-2)-a_2y(k-3)-\ldots-a_{n_a}y(k-n_a-1)+ \\ +b_0u(k-2)+b_1u(k-3)+\ldots+b_0u(k-n_b-2) \qquad (2b)$$

$$y(k-N)=-a_1y(k-1-N)-a_2y(k-2-N)-\ldots- \\ -a_{n_a}y(k-n_a-N)+b_0u(k-1-N)+\ldots+ \\ +b_1u(k-2-N)+\ldots+b_0u(k-n_b-1-N) \qquad (2c)$$

The number of observations N must provide a sufficient number of equations to determine all the n_a+n_b+1 unknown parameters a_j and b_j, that is:

$$1+N\geq n_a+n_b+1 \quad \therefore \quad N\geq n_a+n_b \qquad (3)$$

Equations (2a) to (2c) can be written in matrix form as:

$$\mathbf{Y}=\mathbf{\Phi\Theta} \qquad (4)$$

where $\mathbf{Y}$ is the (N+1)×1 **vector of outputs**:

$$\mathbf{Y}=\begin{bmatrix} y(k) \\ y(k-1) \\ y(k-2) \\ \vdots \\ y(k-N)\end{bmatrix} \qquad (5)$$

$\mathbf{\Phi}$ is the (N+1)×(n_a+n_b+1) **matrix of observations**:

$$\mathbf{\Phi}=\begin{bmatrix} -y(k-1) & -y(k-2) & \cdots & -y(k-n_a) \\ -y(k-2) & -y(k-3) & \cdots & -y(k-n_a-1) \\ -y(k-3) & -y(k-4) & \cdots & -y(k-n_a-2) \\ \vdots & \vdots & \ddots & \vdots \\ -y(k-1-N) & -y(k-2-N) & \cdots & -y(k-n_a-N) \end{bmatrix}$$

$$\begin{bmatrix} u(k-1) & u(k-2) & \cdots & u(k-n_b-1) \\ u(k-2) & u(k-3) & \cdots & u(k-n_b-2) \\ u(k-3) & u(k-4) & \cdots & u(k-n_b-3) \\ \vdots & \vdots & \ddots & \vdots \\ u(k-1-N) & u(k-2-N) & \cdots & u(k-n_b-1-N) \end{bmatrix} \qquad (6)$$

and $\mathbf{\Theta}$ is the (n_a+n_b+1)×1 **vector of parameters**:

$$\mathbf{\Theta}=\begin{bmatrix} a_1 & a_2 & \ldots & a_{n_a} & b_0 & b_1 & \ldots & b_{n_b}\end{bmatrix}^{\mathrm{T}} \qquad (7)$$

The main goal of system identification is to obtain an *estimate* of the "true" vector $\mathbf{\Theta}$. Suppose that $\hat{\mathbf{\Theta}}$ is an estimate of $\mathbf{\Theta}$. From (4), a *prediction* of $\mathbf{Y}$ is:

$$\hat{\mathbf{Y}}=\mathbf{\Phi}\hat{\mathbf{\Theta}} \qquad (8)$$

A **vector of prediction errors** is defined as the difference between the actual outputs and the predicted outputs:

$$\mathbf{E}=\mathbf{Y}-\hat{\mathbf{Y}}=\mathbf{Y}-\mathbf{\Phi}\hat{\mathbf{\Theta}} \qquad (9)$$

Ideally, if the estimate $\hat{\mathbf{\Theta}}$ is identically equal to the actual parameters vector $\mathbf{\Theta}$, the prediction errors vector $\mathbf{E}$ will be null. Therefore, the best estimate $\hat{\mathbf{\Theta}}$ is the one that *minimizes the estimation errors* $\mathbf{E}$. Since the elements of $\mathbf{E}$ can assume either positive or negative values (and null value as well), we consider to minimize the sum of squares of the elements of $\mathbf{E}$, leading to the following *cost function*:

$$J=\left\|\mathbf{E}\right\|^2=\mathbf{E}^{\mathrm{T}}\mathbf{E}=\left(\mathbf{Y}-\mathbf{\Phi}\hat{\mathbf{\Theta}}\right)^{\mathrm{T}}\left(\mathbf{Y}-\mathbf{\Phi}\hat{\mathbf{\Theta}}\right) \qquad (10)$$

The minimization of the quadratic cost function (10) is the optimality criterion for the Least Squares method. The estimate $\hat{\mathbf{\Theta}}$ that minimizes J is obtained from:

$$\frac{dJ}{d\hat{\mathbf{\Theta}}}=-2\mathbf{\Phi}^{\mathrm{T}}\mathbf{Y}+2\mathbf{\Phi}^{\mathrm{T}}\mathbf{\Phi}\hat{\mathbf{\Theta}}=0 \qquad (11)$$

Since $\mathbf{\Phi}^{\mathrm{T}}\mathbf{\Phi}$ is non-negative definite, J is minimized for:

$$\hat{\boldsymbol{\Theta}} = \left(\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}\right)^{-1}\boldsymbol{\Phi}^{\mathrm{T}}\mathbf{Y} \tag{12}$$

Equation (12) is called the *Least Squares estimate* of $\boldsymbol{\Theta}$. It only exists if $\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}$ is non-singular, and then invertible. A necessary condition for this is that the system is *persistent excited*, so that $\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}$ does not have common lines. A formal definition of persistent excitation is given in [11]. The matrix $\left(\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}\right)^{-1}\boldsymbol{\Phi}^{\mathrm{T}}$ is called the *pseudo-inverse* of $\boldsymbol{\Phi}$ [4][6] [9][10].

In practical applications, physical system usually involves a stochastic disturbance $v(k)$, and the more generic **ARMAX** model shown in Fig. 2 should be used instead of the ARX model. The ARMAX model considers that the output $y(k)$ is a result from the process input $u(k)$ and a disturbance $v(k)$, leading to a more accurate model than an ARX model. By this way, equation (4) is modified to:

$$\mathbf{Y} = \boldsymbol{\Phi}\boldsymbol{\Theta} + \mathbf{V} \tag{13}$$

Where $\mathbf{V}$ is the $(N+1)\times 1$ **vector of disturbances**:

$$\mathbf{V} = \begin{bmatrix} v(k) \\ v(k+1) \\ v(k+2) \\ \vdots \\ v(k+N) \end{bmatrix} \tag{14}$$

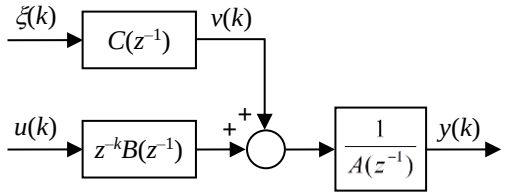

Fig. 2. Block diagram of an ARMAX model.

Multiplying (13) by the pseudo-inverse $\left(\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}\right)^{-1}\boldsymbol{\Phi}^{\mathrm{T}}$ and using (12), we obtain:

$$\hat{\boldsymbol{\Theta}} = \boldsymbol{\Theta} + \left(\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}\right)^{-1}\boldsymbol{\Phi}^{\mathrm{T}}\mathbf{V} \tag{15}$$

This means that estimate $\hat{\boldsymbol{\Theta}}$ may be *biased*, with standard deviation given by the following expected value:

$$E\left\{\left(\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}\right)^{-1}\boldsymbol{\Phi}^{\mathrm{T}}\mathbf{V}\right\} \tag{16}$$

When the disturbance $v(k)$ is an uncorrelated stochastic signal with null average, that is, a discrete white noise $v(k) = \xi(k)$, the expected value (16) will be null, and the estimate $\hat{\boldsymbol{\Theta}}$ will be unbiased. Otherwise, when the disturbance $v(k)$ is a *colored noise*:

$$v(k) = C(z^{-1})\xi(k)$$
$$C(z^{-1}) = 1 + c_1 z^{-1} + c_2 z^{-2} + \ldots + c_{n_c} z^{-n_c}$$
$$v(k) = \xi(k) + c_1\xi(k-1) + c_2\xi(k-2) + \ldots + c_{n_c}\xi(k-n_c) \tag{17}$$

then the estimate $\hat{\boldsymbol{\Theta}}$ given by (12) for an ARX model will be biased, unless the disturbance $v(k)$ is included in the model. Therefore, to account for a stochastic disturbance in the system being identified, the ARMAX model must be used. The disturbance $v(k)$ is assumed a white noise $\xi(k)$ filtered by polynomial $C(z^{-1})$, whose coefficients are included in the *extended* parameters vector:

$$\boldsymbol{\Theta} = \begin{bmatrix} a_1 & a_2 & \ldots & a_{n_a} & b_0 & b_1 & \ldots & b_{n_b} & c_1 & c_2 & \ldots & c_{n_c} \end{bmatrix}^{\mathrm{T}} \tag{18}$$

and the identification is referred as **Extended Least Squares (ELS)** [3].

B. Recursive Form

For purposes of adaptive control, the process parameters must be identified at each sampling interval, so that any change in those parameters can be tracked and used do adjust the controller parameters. This allows the controller to adapt to changes in the process dynamics. In this sense, the batch solution given by equation (12) is completely unpractical, since as the number of equations N increases, the dimensions of $\mathbf{Y}$ and $\boldsymbol{\Phi}$ will also increase, causing computing overload in the digital computer that implements the identification and the control. It's necessary that the previous parameters estimate $\hat{\boldsymbol{\Theta}}(N-1) = \hat{\boldsymbol{\Theta}}_{N-1}$ can be used to compute the estimate $\hat{\boldsymbol{\Theta}}(N) = \hat{\boldsymbol{\Theta}}_N$ for the current time $k = N$. This is the key idea behind the recursive form of the Least Squares method.

C. Recursive Least Squares with the Matrix Inversion Lemma

Recall the non-recursive formulation of the Least Squares given by equations (4) to (6). At the current time $k = N$, a new observation ϕ_N is added to the observations matrix $\boldsymbol{\Phi}$, and a new element y_N is added to the vector of outputs $\mathbf{Y}$, that is:

$$\boldsymbol{\Phi}_N = \begin{bmatrix} \boldsymbol{\Phi}_{N-1} \\ \phi_N \end{bmatrix} \quad ; \quad \mathbf{Y}_N = \begin{bmatrix} \mathbf{Y}_{N-1} \\ y_N \end{bmatrix} \tag{19}$$

where:

$$\phi_N = \begin{bmatrix} -y_{N-1} & -y_{N-2} & -y_{N-3} & \cdots \\ \cdots & -y_{N-n_a} & u_{N-1} & u_{N-2} & u_{N-n_b-1} \end{bmatrix} \tag{20}$$

Defining the **covariance matrix P** as:

$$\mathbf{P} = \left(\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}\right)^{-1} \tag{21}$$

from (12), the parameters estimate $\hat{\boldsymbol{\Theta}}_N$ for all N observations at the sample time $k = N$ is given by:

$$\hat{\boldsymbol{\Theta}}_N = \mathbf{P}_N\boldsymbol{\Phi}_N^{\mathrm{T}}\mathbf{Y}_N \tag{22}$$

Using the Matrix Inversion Lemma [4][6][12], equation (22) can be modified to the following recursive form (see Section V):

$$\hat{\mathbf{\Theta}}_N = \hat{\mathbf{\Theta}}_{N-1} + \mathbf{K}_N \varepsilon_N \tag{23}$$

where ε is the **output prediction error**:

$$\varepsilon_N = y_N - \hat{y}_N = y_N - \phi_N \hat{\mathbf{\Theta}}_{N-1} \tag{24}$$

K is the **vector of estimation gains**:

$$\mathbf{K}_N = \frac{\mathbf{P}_{N-1}\phi_N^{\mathrm{T}}}{h_N} \tag{25}$$

P is the **covariance matrix**:

$$\mathbf{P}_N = \mathbf{P}_{N-1} - \frac{\mathbf{P}_{N-1}\phi_N^{\mathrm{T}}\phi_N\mathbf{P}_{N-1}}{h_N} = (\mathbf{I} - \mathbf{K}_N\phi_N)\mathbf{P}_{N-1} \tag{26}$$

and

$$h_N = 1 + \phi_N \mathbf{P}_{N-1}\phi_N^{\mathrm{T}} \tag{27}$$

Equation (23) means that the parameters estimate is updated at each sampling time as:

$$\begin{pmatrix}\text{Current}\\ \text{Estimate}\end{pmatrix} = \begin{pmatrix}\text{Previous}\\ \text{Estimate}\end{pmatrix} + \text{Gain} \times \begin{pmatrix}\text{Prediction}\\ \text{Error}\end{pmatrix} \tag{28}$$

Therefore, as the estimation proceeds and converges to the "true" parameters in $\mathbf{\Theta}$, the prediction error tends to zero, being an indicator of the convergence of the identification. The covariance matrix **P** is also indicative of the convergence. Since the magnitude of the elements in the main diagonal of **P** are related to the variances of the corresponding elements in the parameters estimate $\hat{\mathbf{\Theta}}$, a small element p_{jj} (low variance) in **P** means that the corresponding parameter θ_j is a good estimate. For practical implementation, when there is no good guess about the actual values of the parameters in $\mathbf{\Theta}$, the diagonal elements in **P** can be set to high values (high initial variances). As the estimation proceeds, the updates in **P** by equation (24) will reduce the magnitudes of its diagonal elements, and the elements of the gain vector **K** tend to zero as the number of observations increases and the estimates converge. Therefore, by avoiding the diagonal elements in **P** to become overmuch low, the prediction error will provide a continuous correction of the parameters estimate $\hat{\mathbf{\Theta}}$ to its "true" value $\mathbf{\Theta}$, allowing to track eventual changes in $\mathbf{\Theta}$. This can be done by periodically resetting the covariance matrix **P** with high diagonal values.

A more efficient way to avoid the diagonal elements in **P** to decrease to zero is the use of a *forgetting factor* λ. It's a factor between 0 and 1, set lightly small to 1, with the aim to progressively eliminate the influence of past observations on the updates of **P**, which is modified to:

$$\mathbf{P}_N = \left(\mathbf{\Phi}_N^{\mathrm{T}}\mathbf{\Phi}_N\right)^{-1} = \left(\lambda\mathbf{\Phi}_{N-1}^{\mathrm{T}}\mathbf{\Phi}_{N-1} + \phi_N^{\mathrm{T}}\phi_N\right)^{-1} \tag{29}$$

The factor h_N in (27) and the covariance matrix **P** in (26) are now updated as:

$$h_N = \lambda + \phi_N \mathbf{P}_{N-1}\phi_N^{\mathrm{T}} \tag{30}$$

$$\mathbf{P}_N = \frac{1}{\lambda}(\mathbf{I} - \mathbf{K}_N\phi_N)\mathbf{P}_{N-1} \tag{31}$$

The forgetting factor is usually set as $0.97 \le \lambda \le 1$. By using a forgetting factor, the elements of **P** should not decrease to zero, due to the repetitive division of **P** by λ. However, if no new information enters the observations matrix $\mathbf{\Phi}$ for a long period of time, that is, if the system is not *persistently excited*, the repetitive divisions by λ may increase overmuch the values of **P** and cause computing overflow. Therefore, λ should be set close to 1. Fig. 3 shows the cumulative effect of the forgetting factor on the computations done at past sample times.

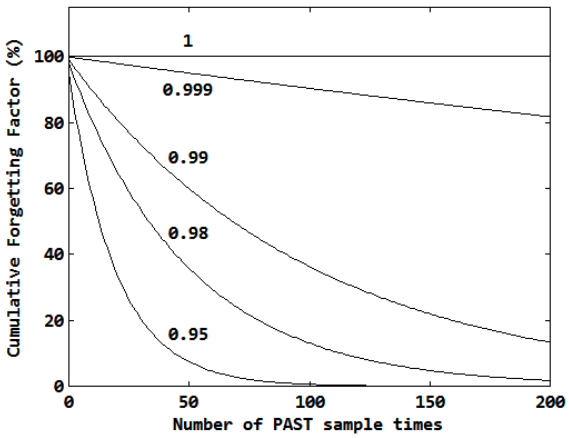

Fig. 3. Effect of the forgetting factor on past estimations.

D. Recursive Least Squares with UD Factorization

The Least Squares method based on the Matrix Inversion Lemma is a powerful parameter estimator. However, it is sensitive to the propagation of numerical errors, which may turn the covariance matrix **P** negative semidefinite, causing divergence of the parameters estimate $\hat{\mathbf{\Theta}}$. A solution for this problem is the factorization of **P** as a product of matrices. Factorization methods were developed in past decades to improve the robustness of computations performed with early computers with low arithmetic precision. However, those methods remain useful for use in today's modern computers.

Several methods for matrix factorization are available for computing the RLS recursions. The most widely used is the is the **UD Factorization** due to Bierman [4]. In this method, the covariance matrix **P** is factorized as a product of two matrices, **U** and **D**, in the following form:

$$\mathbf{P} = \mathbf{U} \times \mathbf{D} \times \mathbf{U}^{\mathrm{T}} \tag{32}$$

where **U** is an upper triangular matrix with unity diagonal elements, and **D** is a diagonal matrix. The Least Squares

recursions based on UD Factorization are summarized in Section VI.

The algorithm for parameter estimation using the Recursive Least Squares method is summarized below:

1. At the current time $k = N$, acquire the process input $u(k)$ and output $y(k)$.
2. Compute the vector of observations ϕ_k (equation (19)).
3. Compute the prediction error ε_k (equation (24)).
4. Compute the gain vector $\mathbf{K}$ (equation (25), or (48) if using UD Factorization).
5. Update the vector of parameter estimates $\hat{\boldsymbol{\Theta}}$ (equation (23)).
6. Update the covariance matrix $\mathbf{P}$ (equations (26) or (31) if using the Matrix Inversion Lemma; or (45) to (47) if using UD Factorization).
7. Wait for the next sampling time and return to step 1.

Fig. 4 illustrates the idea of the Recursive Least Squares method for system identification.

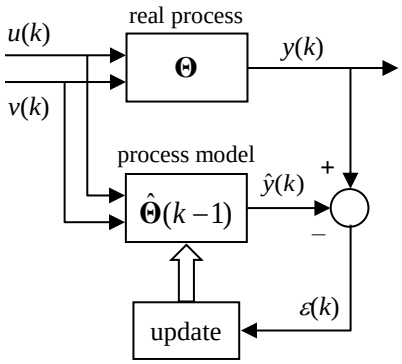

Fig. 4. Illustration of the recursive estimation procedure.

III. Persistent Excitation

For reliable parameter estimation, the output signal of the system under identification must carry representative information about the system dynamics. In other words, the input signal must excite properly the system so that its output carries rich information about the system dynamics. The ideal signal for this purpose must have an uniform power spectrum. A discrete *white noise* meets this requirement, however, it is an ideal model of signal. Nevertheless, it's possible to synthetize physical signals with approximate uniform spectrum on a suitable frequency range. One of such signals is the PRBS.

A **Pseudo-Random Binary Signal (PRBS)** [13], is an uncorrelated binary sequence. It can be generated by a *shift-register*, which is binary register whose bits are moved in a FIFO (first-in first-out) manner, with a trigger frequency f_0. Since a shift-register is a deterministic device, its PRBS signal is actually predictable and cyclic (periodic). However, by choosing a suitable length (number of bits) for the shift-register, it's possible to generate PRBS signals that persist without repetition for an arbitrary long time, even by thousands of years.

For analog applications, including a low-pass filter at the output of a PRBS generator will produce a *band-limited gaussian white noise*, that is, a noise with flat (uniform) power spectrum up to the cut-off frequency of the filter. Alternatively, one may compute a weighted sum of the bits of the shift-register, using a set of resistors, with similar result. Those are some ways to generate analog signals with uniform spectrum up to several MHz. Such digitally synthetized analog noises are easy to generate using the same computer that implements the digital controller.

The simpler and more commonly used PRBS generator is the *feedback shift-register*, shown in Fig. 5. A shift-register with length of n bits triggered by a clock frequency f_0. An exclusive-or gate is used to generate a serial input bit to the shift-register, from its latest (n^{th}) bit and an specific intermediate (m^{th}) bit. The maximum number of states for a shift-register of length n is 2^n, but the state $\{0_1 0_2 0_3 0_4 \ldots 0_n\}$ gets stuck, since the exclusive-or gate regenerates a bit 0 in the serial input of the shift-register. Therefore, the maximum number of states for this feedback shift-register is $\mathbf{2^n–1}$.

Starting from a given non-zero initial state, the shift-register gets several states, eventually repeating its states history after $2^n–1$ clock pulses. The PRBS signal is the binary sequence generated by the n^{th} bit. It is cyclic with period $\boldsymbol{T_0 = (2^n–1)/f_0}$. This period can be made arbitrary long by choosing suitable values for the trigger frequency f_0, the length n of the shift-register, and the feedback bit m, so that the PSBS signal becomes an uncorrelated noise. It's not necessary to build a physical circuit to generate a PRBS. It is easily generated by a software simulated shift-register, with the same language used to perform the identification, or with native functions of the software, like "idinput" from MATLAB®. In this work, the A/D conversion to acquire $y(k)$ and the D/A conversion to synthetize $u(k)$ were implemented with a data acquisition board based on the C++ programming language. Therefore, the shift-register was also implemented in C++, within the system identification program.

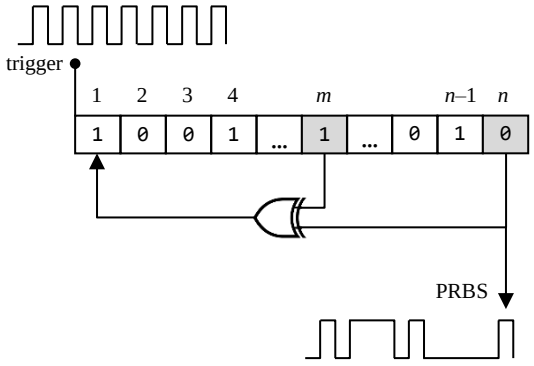

Fig. 5. Feedback shift-register with length n and feedback bit m.

For system identification purposes, the system can be excited by a single PRBS signal, or by a combination of an specific input signal (e.g.: step, square wave, etc) plus a small PRBS, which will act as a *persistent excitation* to the process, mainly when the process is at a steady state condition, to avoid the matrix $\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}$ in (12) becoming non-singular.

IV. System Identification in Practice

The system identification was applied to the thermal airflow system "Process Trainer PT-326" [14], by Feedback Instruments Limited, shown in Fig. 6. In this system, a rotating impeller generates an airflow through a open-end

duct. The impeller rotation is manually adjusted to generate a fixed air flowrate. At the inlet of the duct, immediately after the impeller, there is a heating wire grid of Ni-Cr alloy, powered by a variable current source excited by the process input signal (or control signal), to heat the airflow generated by the impeller. The duct has three (left, central, and right) ports for insertion of a sensor (thermistor) probe into the duct, to measure the airflow temperature, which is the process variable to be controlled. A particular aspect of this thermal process is the existence of a small time-delay in the temperature response due to the displacement between the heating grid (actuator) at the inlet of the duct and the port where the thermistor probe (sensor) is inserted into the duct. Those time-delays were [15]: τ_1 **= 0.214 sec** (left port, at 28 mm), τ_2 **= 0.257 sec** (central port, at 140 mm), and τ_3 **= 0.341 sec** (right port, at 274 mm). Another particular aspect is that the process dynamics may change with variations in the external ambient air impelled into the duct. Even though those changes are small, they do exist.

A schematic of the thermal airflow system PT-326 is shown in Fig. 7. The red lines indicate the output temperature signal, which can be taken from socket "Y". The blue lines correspond to the process input, which is delivered to the system through socket "A". When socket "Y" is not connected to socket "X", the system is in open loop.

The speed of the rotating impeller can be adjusted at specific fixed values by the "throttle control" dial. After fixing the impeller speed, the time delay of the airflow temperature process will depend on the location of the temperature sensor probe within the duct. In Fig. 7, the sensor probe is placed at the right port of the duct, so that the time delay of the process will be longer.

For open loop system identification, the output temperature signal is acquired by an A/D converter from socket "Y", and the excitation signal for the process input is delivered by a D/A converter through socket "A". Signal conditioning circuits are necessary to connect the A/D and D/A converters to the proper sockets of the system.

The Recursive Extended Least Squares (RELS) method based on the Matrix Inversion Lemma and on the UD Factorization was used to estimate the model parameters of the PT-326 system. It was observed that the step response of this system follows approximately the response of a *first order plus time-delay* (**FOPTD**) model. For purposes of identification and control of the PT-326 system, the sampling time was chosen to be T **= 0.2 sec**.

Fig. 6. Front view of the thermal airflow system PT-326.

For the sampling time T = 0.2 sec, the discrete model of the system was chosen as [15]:

$$G_P(z) = \frac{Y(z)}{U(z)} = z^{-2}\,\frac{b_0 + b_1 z^{-1}}{1 + a_1 z^{-1}} \tag{33}$$

The exact value of the time-delay exponent k depends on the sampling period T. However, since k is at least 1 ($k \geq 1$), the discrete transfer function (33) can be written in a more generic form:

$$G_P(z) = \frac{Y(z)}{U(z)} = z^{-1}\,\frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1}} \tag{34}$$

The system identification was implemented in a PC computer using the C++ programming language. A data acquisition board [1] installed in the computer provided the necessary A/D and D/A interfaces with the process to acquire the process output signal $y(k)$ and to produce the process input signal $u(k)$.

The identification using the RELS method was performed over 60 seconds, which corresponds to 300 sampling intervals ($60/T = 60/0.2 = 300$). The input signal to the system was a pure PRBS generated by software and synthetized by a D/A converter [1][2], with amplitude 0–4 volts and trigger

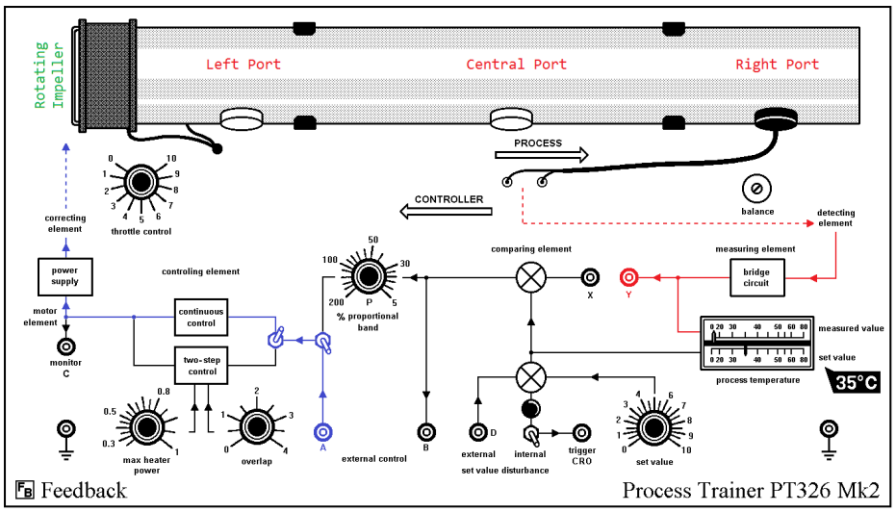

Fig. 7. Schematic of the thermal airflow system Process Trainer PT-326.

period equal to the sampling period *T*. The first 20 seconds of this input signal are shown in Fig. 8, and the corresponding process output is shown in Fig. 9. Recall that the output signal is the temperature of the air stream at the point where the temperature sensor is located.

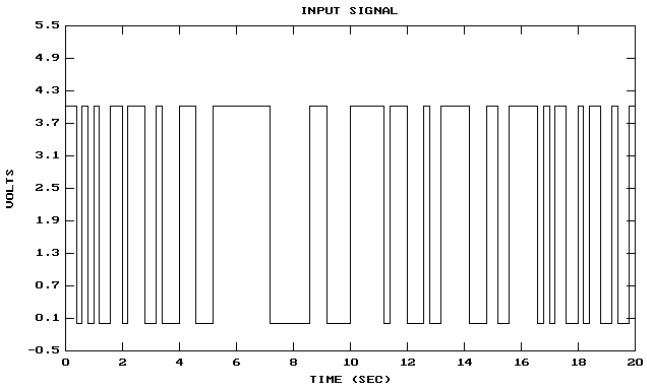

Fig. 8. PRBS signal used as process input.

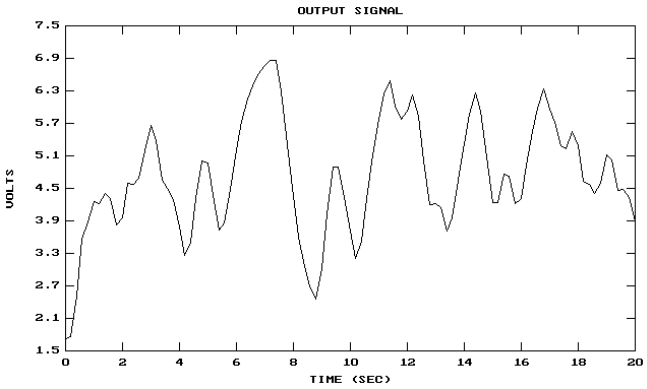

Fig. 9. Output temperature signal resulting from the input PRBS signal.

Fig. 10 and 11 show the results of the identification for the RELS method with UD factorization, when the system has its temperature sensor located at the right port. The initial values of all the parameters were set to 0.1. The forgetting factor for the recursive identification was set as λ **= 0.985**. The matrix **U** was initialized as an unity triangular superior matrix, and the matrix **D** was initialized as a diagonal matrix **D** = 5000**I**. The parameters estimate resulted as:

$$\hat{\boldsymbol{\Theta}} = \begin{bmatrix} b_0 & b_1 & b_2 & a_1 \end{bmatrix}^{\mathrm{T}} = \begin{bmatrix} 0.012 & 0.167 & 0.130 & -0.871 \end{bmatrix}^{\mathrm{T}} \quad (35)$$

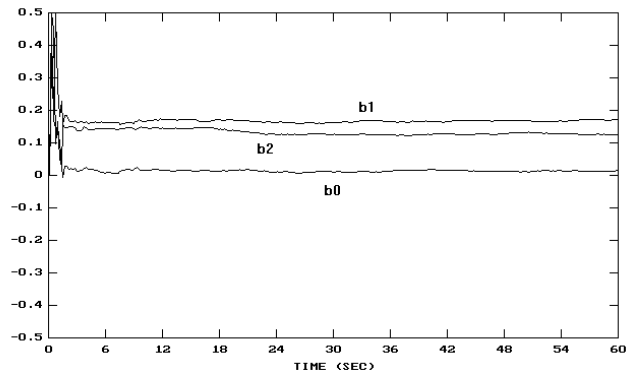

Fig. 10. Estimates for parameters b_j.

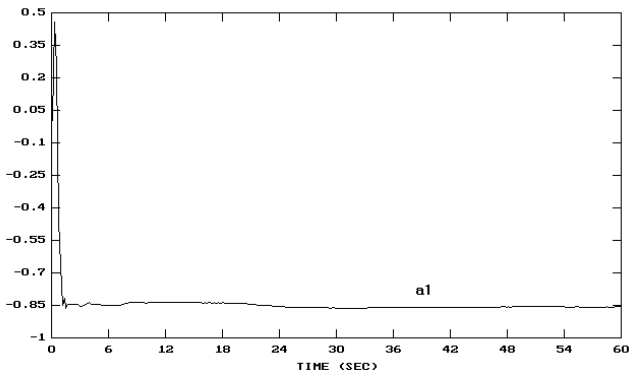

Fig. 11. Estimate for parameter a_1.

The RELS method based on the Matrix Inversion Lemma was also implemented. The same initial values for the parameter estimates and for the forgetting factor were used. The covariance matrix **P** was initialized as **P** = 5000**I**, and the initial values for elements of gain vector **K** were set to 5000. The results for the identification are shown in Fig. 12 and 13. Notice that estimates values obtained using the Matrix Inversion Lemma are statistically consistent with the values obtained with UD factorization. Compare Fig. 12 and 13 to Fig. 10 and 11. Clearly, the approach based on the Matrix Inversion Lemma is more sensitive to numerical calculations, although it was not ill-conditioned.

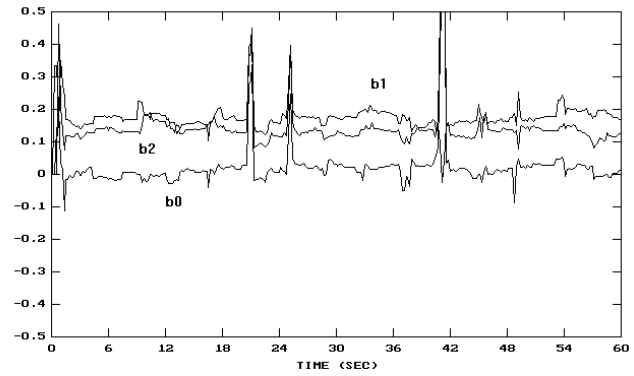

Fig. 12. Estimates for parameters b_j (matrix inversion lemma)

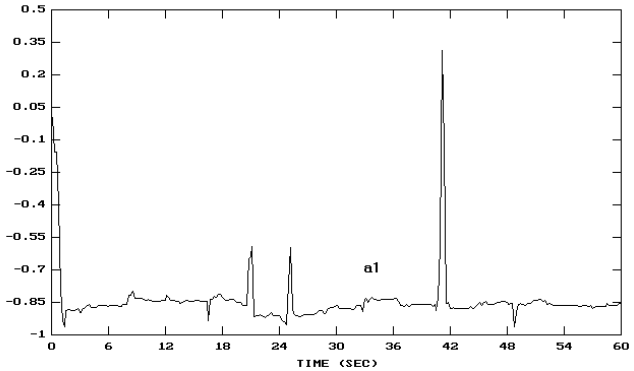

Fig. 13. Estimate for parameter a_1 (matrix inversion lemma)

Since the identification method based on the Matrix Inversion Lemma gives imprecise results, the RELS method based on UD Factorization, due to its greater numerical robustness, was used in model identification for the two remaining positions of the temperature sensor. For the central position, the parameter estimates are shown in Fig. 14 and 15, with the following result:

$$\hat{\boldsymbol{\Theta}} = \begin{bmatrix} b_0 & b_1 & b_2 & a_1 \end{bmatrix}^{\mathrm{T}} = \begin{bmatrix} 0.072 & 0.205 & 0.068 & -0.856 \end{bmatrix}^{\mathrm{T}} \quad (36)$$

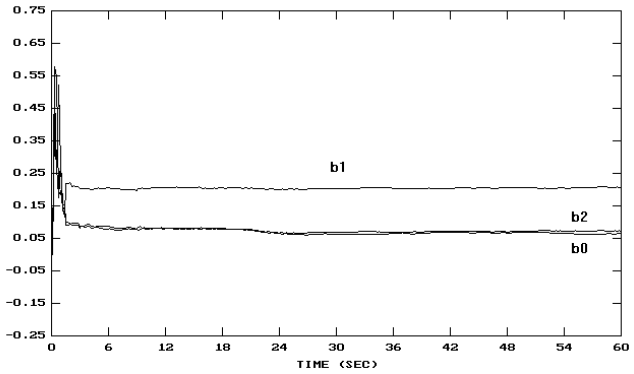

Fig. 14. Estimates for parameters b_j.

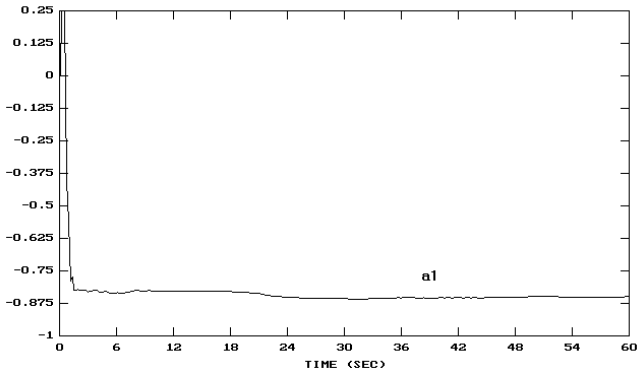

Fig. 15. Estimate for parameter a_1.

Finally, for the temperature sensor located at the left position, the parameter estimates are show in Fig. 16 and 17, with the following result:

$$\hat{\boldsymbol{\Theta}} = \begin{bmatrix} b_0 & b_1 & b_2 & a_1 \end{bmatrix}^{\mathrm{T}} = \begin{bmatrix} 0.123 & 0.128 & 0.020 & -0.845 \end{bmatrix}^{\mathrm{T}} \quad (37)$$

By comparing (35), (36), and (37), the absolute value of the time-constant parameter a_1 decreases as the temperature sensor is positioned more and more to the left, that is, closer and closer to the heating wire grid after the air impeller. This is consistent with the fact that the system response must become faster as the sensor gets closer to the actuator.

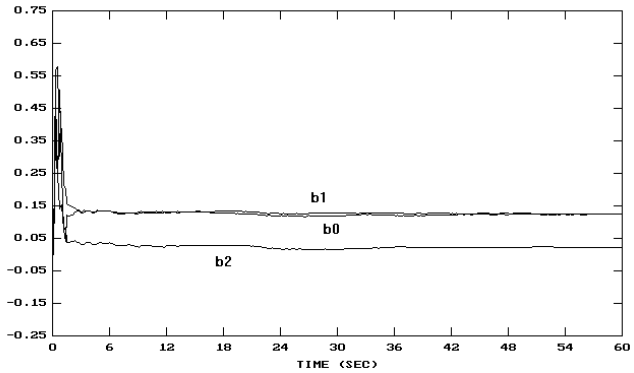

Fig. 16. Estimates for parameters b_j.

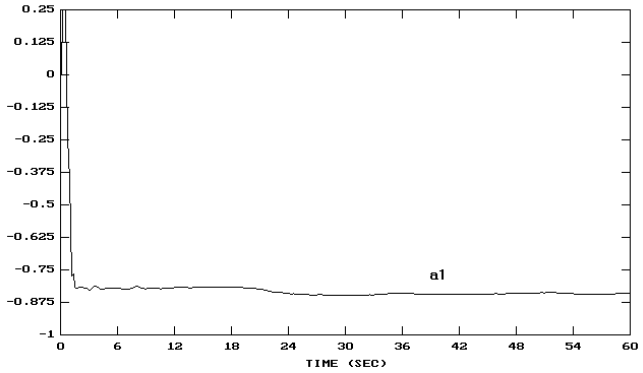

Fig. 17. Estimate for parameter a_1.

V. Recursive Least Squares Based on the Matrix Inversion Lemma

The equations for the recursive form of the Least Squares method based on the **Matrix Inversion Lemma** are derived here. The covariance matrix **P** is defined as:

$$\mathbf{P} = \left(\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}\right)^{-1} \quad (38)$$

From (18), the covariance matrix at the discrete time $k = N$ is:

$$\mathbf{P}_N = \left(\boldsymbol{\Phi}_N^{\mathrm{T}}\boldsymbol{\Phi}_N\right)^{-1} = \left(\begin{bmatrix}\boldsymbol{\Phi}_{N-1}^{\mathrm{T}} & \boldsymbol{\phi}_N^{\mathrm{T}}\end{bmatrix}\begin{bmatrix}\boldsymbol{\Phi}_{N-1} \\ \boldsymbol{\phi}_N\end{bmatrix}\right)^{-1} = \left(\boldsymbol{\Phi}_{N-1}^{\mathrm{T}}\boldsymbol{\Phi}_{N-1} + \boldsymbol{\phi}_N^{\mathrm{T}}\boldsymbol{\phi}_N\right)^{-1}$$

$$\mathbf{P}_N = \left(\mathbf{P}_{N-1}^{-1} + \boldsymbol{\phi}_N^{\mathrm{T}}\boldsymbol{\phi}_N\right)^{-1} \quad (39)$$

Equation (39) can be expanded using the Matrix Inversion Lemma [6]:

$$\left(\mathbf{A}+\mathbf{BCD}\right)^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{B}\left(\mathbf{DA}^{-1}\mathbf{B} + \mathbf{C}^{-1}\right)^{-1}\mathbf{DA}^{-1} \quad (40)$$

By choosing: $\mathbf{A} = \mathbf{P}_{N-1}^{-1}$, $\mathbf{B} = \boldsymbol{\phi}_N^{\mathrm{T}}$, $\mathbf{C} = 1$, and $\mathbf{D} = \boldsymbol{\phi}_N$, equation (39) can be written as:

$$\mathbf{P}_N = \mathbf{P}_{N-1} - \mathbf{P}_{N-1}\boldsymbol{\phi}_N^{\mathrm{T}}\left(\boldsymbol{\phi}_N\mathbf{P}_{N-1}\boldsymbol{\phi}_N^{\mathrm{T}} + 1\right)^{-1}\boldsymbol{\phi}_N\mathbf{P}_{N-1}$$

$$\mathbf{P}_N = \mathbf{P}_{N-1} - \frac{\mathbf{P}_{N-1}\boldsymbol{\phi}_N^{\mathrm{T}}\boldsymbol{\phi}_N\mathbf{P}_{N-1}}{1+\boldsymbol{\phi}_N\mathbf{P}_{N-1}\boldsymbol{\phi}_N^{\mathrm{T}}} \quad (41)$$

Defining:

$$h_N = 1 + \boldsymbol{\phi}_N\mathbf{P}_{N-1}\boldsymbol{\phi}_N^{\mathrm{T}} \quad (42)$$

$$\mathbf{K}_N = \frac{\mathbf{P}_{N-1}\boldsymbol{\phi}_N^{\mathrm{T}}}{h_N} \quad (43)$$

Notice that the Matrix Inversion Lemma allows the calculation of the covariance matrix in (41) without the need to perform matrix inversions, which is often an ill-conditioned computation.

From (12), (18), (41), and (40), the parameters estimate $\hat{\boldsymbol{\Theta}}_N$ at the discrete time $k = N$ is:

$$\begin{aligned}
\hat{\boldsymbol{\Theta}}_N &= \mathbf{P}_N \boldsymbol{\Phi}_N^{\mathrm{T}} \mathbf{Y}_N \\
&= \mathbf{P}_N \begin{bmatrix} \boldsymbol{\Phi}_{N-1}^{\mathrm{T}} & \phi_N^{\mathrm{T}} \end{bmatrix} \begin{bmatrix} \mathbf{Y}_{N-1} \\ y_N \end{bmatrix} \\
&= \mathbf{P}_N \left(\boldsymbol{\Phi}_{N-1}^{\mathrm{T}} \mathbf{Y}_{N-1} + \phi_N^{\mathrm{T}} y_N \right) \\
&= \left(\mathbf{P}_{N-1} - \frac{\mathbf{P}_{N-1} \phi_N^{\mathrm{T}} \phi_N \mathbf{P}_{N-1}}{h_N} \right) \left(\boldsymbol{\Phi}_{N-1}^{\mathrm{T}} \mathbf{Y}_{N-1} + \phi_N^{\mathrm{T}} y_N \right) \\
&= \mathbf{P}_{N-1} \boldsymbol{\Phi}_{N-1}^{\mathrm{T}} \mathbf{Y}_{N-1} - \frac{\mathbf{P}_{N-1} \phi_N^{\mathrm{T}} \phi_N \mathbf{P}_{N-1} \boldsymbol{\Phi}_{N-1}^{\mathrm{T}} \mathbf{Y}_{N-1}}{h_N} + \\
&\quad + \mathbf{P}_{N-1} \phi_N^{\mathrm{T}} y_N - \frac{\mathbf{P}_{N-1} \phi_N^{\mathrm{T}} \phi_N \mathbf{P}_{N-1} \phi_N^{\mathrm{T}} y_N}{h_N} \\
&= \hat{\boldsymbol{\Theta}}_{N-1} - \frac{\mathbf{P}_{N-1} \phi_N^{\mathrm{T}} \phi_N \hat{\boldsymbol{\Theta}}_{N-1}}{h_N} + \\
&\quad + \left(\mathbf{P}_{N-1} - \frac{\mathbf{P}_{N-1} \phi_N^{\mathrm{T}} \phi_N \mathbf{P}_{N-1}}{h_N} \right) \phi_N^{\mathrm{T}} y_N \\
&= \hat{\boldsymbol{\Theta}}_{N-1} - \frac{\mathbf{P}_{N-1} \phi_N^{\mathrm{T}} \hat{y}_N}{h_N} + \mathbf{P}_N \phi_N^{\mathrm{T}} y_N \\
&= \hat{\boldsymbol{\Theta}}_{N-1} - \frac{\mathbf{P}_{N-1} \phi_N^{\mathrm{T}}}{h_N} \hat{y}_N + \frac{\mathbf{P}_{N-1} \phi_N^{\mathrm{T}}}{h_N} y_N \\
&= \hat{\boldsymbol{\Theta}}_{N-1} + \mathbf{K}_N \left(y_N - \hat{y}_N \right)
\end{aligned}$$

$$\hat{\boldsymbol{\Theta}}_N = \hat{\boldsymbol{\Theta}}_{N-1} + \mathbf{K}_N \varepsilon_N \tag{44}$$

where **K** is the **vector of estimation gains**, and ε is the **prediction error**.

Equation (44) is the *recursive form of the Least Squares estimation* of $\boldsymbol{\Theta}$ based on the Matrix Inversion Lemma.

VI. Recursive Least Squares Based on UD Factorization

When the **Bierman's UD Factorization** method is used instead of the Matrix Inversion Lemma, the Least Squares recursions follow the steps bellow:

1) Initialize matrices **U** and **D**. Let *n* be the number of parameters to estimate.

2) Compute the *n* vectors **F** and **G** as:

$$\begin{cases} \mathbf{F}_N = \mathbf{U}_{N-1}^{\mathrm{T}} \phi_N \\ \mathbf{G}_N = \mathbf{D}_{N-1} \mathbf{F}_N \end{cases} \tag{45}$$

3) Set $\beta_0 = \lambda$, where λ is the forgetting factor.

4) For j = 1 to n, repeat:

$$\begin{cases} \beta_j = \beta_{j-1} + f_j g_j \\ d_{jj(N)} = \dfrac{\beta_{j-1} d_{j(N-1)}}{\beta_j \lambda} \\ v_j = g_j \\ \mu_j = -\dfrac{f_j}{\beta_{j-1}} \end{cases} \tag{46}$$

where d_{jj} are the diagonal elements of **D**.

If j<1:

For i = 1 to j–1 compute:

$$\begin{cases} u_{ij(N)} = u_{ij(N-1)} + v_{i(N-1)} \mu_j \\ v_{i(N)} = v_{i(N-1)} + u_{ij(N-1)} v_j \end{cases} \tag{47}$$

where u_{ij} are the upper triangular elements of **U**.

5) Compute the vector of estimation gains:

$$\mathbf{K}_N = \frac{1}{\beta_n} \begin{bmatrix} v_1 & v_2 & \dots & v_n \end{bmatrix}^{\mathrm{T}} \tag{48}$$

6) Update the parameters estimate:

$$\hat{\boldsymbol{\Theta}}_N = \hat{\boldsymbol{\Theta}}_{N-1} + \mathbf{K}_N \varepsilon_N \tag{49}$$

7) Wait for the next sampling time and return to step 2.

Notice that when using the UD Factorization method to implement the Least Squares recursions, the covariance matrix **P** is not explicitly calculated. Instead, matrices **D** and **U** are updated from their elements d_j and u_{ij} in (46) and (47), and then the gain vector **K** is explicitly calculated, and used to update the parameters estimate $\hat{\boldsymbol{\Theta}}$ in (49).

VII. Conclusion

This article presented how system identification can be used to estimate model parameters of a physical system.

The results obtained from the identification of the thermal airflow process PT-326 indicate the suitability of the Recursive Least Squares method, which can be easily applied to the identification of industrial process, provided that the process signals required by the identification can be acquired from the industrial instrumentation system. By using off-line system identification to estimate suitable models for industrial processes, those models can be used in the design of controllers, including ordinary PID controllers, for further implementation in the Plant Control System (SCADA or DCS).

Among the two mathematical approaches investigated in this work to solve the least squares identification problem, the approach based on UD Factorization showed greater numerical robustness when compared to the approach based on the Matrix Inversion Lemma. Therefore, UD Factorization should be preferable for implementing system identification applications.

The software routine developed for identification was combined with a routine for adaptive self-tuning control of the system – the second part of the work, described in a separate article.

ACKNOWLEDGMENT

The author thankfully acknowledges Dr. J.A.L. Barreiros, and Dr. Orlando "Nick" F. Silva for the helpful discussions about theoretical and practical aspects of this work.

REFERENCES

[1] *ACL-812PG Enhanced Multi-Function Data Acquisition Card*, ADClone Inc., 1994.
[2] *ACL-812PG Software Utility and C Language Library*, ADClone Inc., 1994.
[3] F. M. Hughes, "Self-Tuning and Adaptive Control: A Review of Some Basic Techniques", *Trans. Inst. MC*, 1986, vol. 8, nº 2, pp. 100-110.
[4] G. J. Bierman, *Factorization Methods for Discrete Sequential Estimation*, 1st ed. New York, USA: Academic Press, 1977.
[5] J. A. L. Barreiros, "Aplicação de Técnicas de Controle Adaptativo Auto-Ajustável à Síntese de Estabilizadores de Sistemas de Potência", Tese de Doutorado, Departamento de Engenharia Elétrica, Universidade Federal de Santa Catarina - UFSC, Florianópolis, SC, Brasil, 1994.
[6] K. J. Åström, and B. Wittenmark, *Adaptive Control*, 2nd ed. Mineola, NY, USA: Dover, 2008, ch. 2, pp. 41-89.
[7] K. J. Åström, and B. Wittenmark, *Computer Controlled Systems: Theory and Design*, 1st ed. Englewood Cliffs, New Jersey, USA: Prentice-Hall International, 1996, ch. 13, pp. 505-527.
[8] K. J. Åström, U. Borisson, L. Ljung, and B. Wittenmark, "Theory and Applications of Self-Tuning Regulators", *Automatica*, 1977, vol. 13, pp. 457-476. Pergamon Press.
[9] K. Ogata, *Discrete-Time Control Systems*, 2nd ed. Upper Saddle River, NJ, USA: Prentice-Hall International, 1995.
[10] L. Ljung, *System Identification: Theory for the User*, 1st ed. Englewood Cliffs, NJ, USA: Prentice-Hall, 1987, ch. 7 and 11.
[11] L. Ljung, and T. Söderström, *Theory and Practice of Recursive Identification*, 1st ed. Cambridge, MA, USA: The MIT Press, 1983.
[12] P. E. Wellstead, "Introduction to Self-Tuning Systems", Lecture Notes, Control System Centre, University of Manchester Institute of Science and Technology - UMIST, Manchester, UK, 1986.
[13] P. Horowitz, and W. Hill, *The Art of Electronics*, 2nd ed. Cambridge, MA, USA: Cambridge University Press, 1989, pp. 655-659.
[14] *Process Trainer PT-326 User Manual*, Feedback Instruments Limited, Crowborough, East Sussex, England, 1996.
[15] S. A. A. Viana, "Modelamento e Controle Contínuo, Digital e Adaptativo de um Sistema de Injeção de Ar Aquecido", Trabalho de Conclusão de Curso, Departamento de Engenharia Elétrica, Universidade Federal do Pará - UFPA, Belém, PA, Brasil, 1997.
[16] T. Söderström, L. Ljung, and I. Gustavssson, "A Theoretical Analysis of Recursive Identification Methods", *Automatica*, 1978, vol. 14, pp. 231-244. Pergamon Press.
[17] T. Söderström, and P. Stoica, *System Identification*, 1st ed. Hertfordshire, UK: Prentice-Hall International, 1989, ch. 9, pp. 320-380.

Received: 07 November 2016
Accepted: 28 July 2017
Published: 15 August 2017

Sidney A.A. Viana was born in Belém, PA, Brazil, in 1974. He received a Graduate degree in Electronics Engineering, in the field of Control Systems, from Universidade Federal do Pará (UFPA), Belém, PA, Brazil, in 1997; a Master of Science degree, in the fields of Control Systems and Artificial Intelligence, from Instituto Tecnológico de Aeronáutica (ITA), São José dos Campos, SP, Brazil, in 1999; and a MBA degree in Project Management, from Fundação Getúlio Vargas (FGV), Rio de Janeiro, RJ, Brazil, in 2013.

In 2000 he joined former Companhia Vale do Rio Doce (now VALE), a global mining company, where he started working in the Carajás Iron Ore Processing Plant as Instrumentation & Automation Engineer, and later as Automation Projects Engineer. In 2007 he moved to VALE's Onça Puma Nickel Project as Lead Automation Engineer and member of the Operational Readiness Team, being responsible for the assessment of the plantwide automation system design, and keeping up with the manufacturing of the automation systems at vendors' factories. In 2009 he joined VALE's Paragominas Bauxite Processing Plant as Process Automation Specialist, being responsible for plant performance assessment and improvement projects, and plantwide data reconciliation. Finally, in 2013 he moved to his current position at VALE's Ferrous Automation Engineering Department, in Belo Horizonte, MG, Brazil, where he works as Specialist Automation Engineer and Project Coordinator.

Mr. Viana was associated to the Institute of Electrical and Electronics Engineers, USA, from 1998 to 2006, when he received the grade of IEEE Senior Member, in recognition for outstanding achievements in the fields of Industrial Instrumentation, Control & Automation. He is author of several papers and technical works on Applied Control Systems, Industrial Automation, and Data Analytics. Two of his works were awarded with the first place of the Brazilian Industry Prize of the National Industry Confederation (CNI), in 2001 and 2004. His main professional interests are Industrial Engineering, industrial applications of Classical, Adaptive and Optimal Control Systems, Applied Computing, Numerical Optimization Methods, Plant Performance Management, Project Management; and Industrial Data Analytics, and Machine Learning.