On-Line Estimation of Time Delay and Continuous-Time Process Parameters

Agarwal, Mukul; Canudas de Wit, Carlos

1985

Document Version:
Publisher's PDF, also known as Version of record

CODEN: LUTFD2/(TFRT-7291)/1-63/(1985)

On-Line Estimation of Time Delay and Continuous-Time Process Parameters

Mukul Agarwal
Carlos Canudas

Department of Automatic Control
Lund Institute of Technology
September 1985

Department of Automatic Control
Lund Institute of Technology
P.O. Box 118
S-221 00 Lund Sweden

Document name
REPORT

Date of issue
September 1985

Document Number
CODEN: LUTFD2/(TFRT-7291)/1-63/(1985)

Author(s)
Mukul Agarwal and Carlos Canudas

Supervisor

Sponsoring organisation

Title and subtitle
On-Line Estimation of Time Delay and Continuous-Time Process Parameters

Abstract

A simple technique is presented for on-line estimation of constant or slowly-varying continuous-time process parameters and time delay. The method is shown to allow considerable flexibility for application to systems of varying complexity. A major advantage of the algorithm lies in its ability to track time-delay variations over a practically unlimited range.

The technique is based on approximation of time delay in the frequency domain by a rational transfer function, construction of the derivatives of process input and output using multiple filters, and estimation using a model nonlinear in the desired parameters. In spite of this inherent nonlinearity with respect to the sought parameters, the estimation schemes lead to the true, unique solution, in general. The cases when this is not true are shown to not be of serious consequence.

Key words
Parameter estimation, time-delay estimation, on-line identification, continuous-time parameters.

Classification system and/or index terms (if any)

Supplementary bibliographical information

ISSN and key title

ISBN

Language
English

Number of pages
63

Recipient's notes

Security classification

ON-LINE ESTIMATION OF TIME DELAY AND CONTINUOUS-TIME PROCESS PARAMETERS

by

Mukul Agarwal* and Carlos Canudas**

Department of Automatic Control

Lund Institute of Technology

Box 118, S-221 00 Lund

Sweden

September, 1985

* Department of Chemical Engineering, University of California, Santa Barbara, CA 93106, U.S.A.

** Laboratoire d'Automatique de Grenoble, B.P. 46, 38402 Saint Martin d'Heres, France. Member of the research group GRECO-Systemes adaptatifs.

Key Words--Parameter estimation, time-delay estimation, on-line identification, continuous-time parameters.

ABSTRACT

A simple technique is presented for on-line estimation of constant or slowly-varying continuous-time process parameters and time delay. The method is shown to allow considerable flexibility for application to systems of varying complexity. A major advantage of the algorithm lies in its ability to track time-delay variations over a practically unlimited range.

The technique is based on approximation of time delay in the frequency domain by a rational transfer function, construction of the derivatives of process input and output using multiple filters, and estimation using a model nonlinear in the desired parameters. In spite of this inherent nonlinearity with respect to the sought parameters, the estimation schemes lead to the true, unique solution, in general. The cases when this is not true are shown to not be of serious consequence.

1. INTRODUCTION

Many practical systems can be described reasonably well by linear models including time delays. But the usefulness of these models for reliable, optimal application has been seriously hindered by the elusive nature of time delays. Unknown, time-varying time delays have frustrated attempts to identify the model parameters under the assumption of a known, constant time delay, by imparting multiple minima to the cost function that these identification methods seek to minimize (Pupeikis, 1985; Kaminskas, 1979). Methods seeking to circumvent the problem of multiple minima caused by unknown time delay, by resorting to such techniques as identification of several different model structures from the available data followed by selection of one, nevertheless restrict themselves to off-line procedures ill-equipped for time-variations in the time delay or the system parameters (Marchand and Fu, 1985; Bohn, 1985; Lilja, 1985; Song and Yu, 1985; Abrishamkar and Bekey, 1985; Rao and Sivakumar, 1976)

On-line procedures capable of tracking time-varying time delays and system parameters have resorted to descrete-time representation of the model. Of these, the methods that overparameterize the discrete model estimate the time delay only to its closest value that is an integer multiple of the sampling period, by utilizing computationally elaborate ways of rejecting the extraneous estimated parameters (Bokor and Keviczky, 1985; Kurz and Goedecke, 1981). On the other hand, the methods that modify the structure of the discrete model, in order to estimate the time delay true to a fraction of the sampling period, face minimization of a cost function exhibiting multiple minima among other numerical problems (Åström and

Wittenmark, 1984, 1985; Lee and Hang, 1985).

This work presents an on-line method for recursive estimation of the continuous-time process parameters and time delay, which is conceptually simple and does not suffer from the above drawbacks. The technique constructs the derivatives of the system input and output by use of multiple filters, approximates the expression for the time delay in the frequency domain by a rational transfer function, discretizes the resultant model while retaining the continuous-time system parameters and time delay in the discrete structure, and employs standard estimation algorithms to track time-varying time delay and parameters.

The next section presents the transformation of the continuous-time system model into the final discrete form suitable for identification. Section 3 develops the application of two recursive estimation schemes and introduces an important stepping mechanism that circumvents the bounds on identifiable time-delay variations dictated by their frequency-domain approximation. Section 4 deals with special situations that warrant certain precautions in implementing the proposed technique. Illustrative simulation examples are interspersed in Sections 3 and 4 at appropriate junctures. The final section summarizes the analysis and states the conclusions.

2. SYSTEM FORMULATION FOR ESTIMATION

Consider a continuous-time, linear representation of a deterministic, single-input, single-output system

$$y(t) + \sum_{i=1}^{n} a_i \frac{d^i y(t)}{dt^i} = \sum_{i=0}^{m} b_i \frac{d^i u(t-\tau_d)}{dt^i} \quad , \quad n \geq m \tag{1}$$

where y(t) and u(t) are output and input, respectively, n and m are known model orders, and τ_d is the total time delay of the system. The time delay is divided into two parts as

$$\tau_d = \tau_o + \tau' \tag{2}$$

where τ_o is a prespecified part, and τ' is the unknown (positive or negative) variation around τ_o. The problem is to estimate on-line the constant or slowly-varying, unknown system parameters a_i, b_i, and τ'.

The approach taken here involves transformation of equation (1) to the frequency domain as

$$\left[1 + \sum_{i=1}^{n} a_i s^i\right] y(s) + Y(0) = \sum_{i=0}^{m} b_i s^i \left[u(s) e^{-\tau_o s}\right] e^{-\tau' s} + U(0) \tag{3}$$

where Y(0) and U(0) are the initial conditions given by

$$Y(0) \equiv \sum_{i=0}^{n-1} \sum_{j=i+1}^{n} y^i(0) s^{n-j} \quad , \quad \text{and} \quad U(0) \equiv \sum_{i=0}^{m-1} \sum_{j=i+1}^{m} u^i(0) s^{m-j} \quad ,$$

$y^i(0)$ and $u^i(0)$ being the initial conditions of the i^{th} derivatives of y(t) and u(t), respectively.

For the proposed estimation technique, equation (3) needs to be expressed in a form where all the unknown parameters appear only in conjunction with terms involving finite powers in s. Hence, the factor $e^{-\tau' s}$ is approximated as a finite-dimensional rational transfer function of the form

$$e^{-\tau' s} \approx \frac{N(\tau' s)}{D(\tau' s)} \qquad (4)$$

where N and D are polynomials in τ's.

Alternatives for approximation of $e^{-\tau' s}$

Several alternatives are possible for approximating $e^{-\tau' s}$ according to equation (4). The simplest and most common is the Padé approximation which has the general form

$$e^{-\tau' s} \approx \left[\frac{1 - \frac{\tau'}{2\ell} s}{1 + \frac{\tau'}{2\ell} s} \right]^{\ell} , \quad \ell = 1,2,\ldots \qquad (5)$$

This approximation is exact in the limit as ℓ tends to infinity. For finite ℓ, its amplitude matches exactly that of $e^{-\tau' s}$, whereas the range of $\omega\tau'$ for which the phase match is exact increases as ℓ becomes large. For instance, the phase match is excellent for values of $\omega\tau'$ upto 1 for ℓ=1 and upto 2.2 for ℓ=4. The disadvantage of selecting large ℓ, of course, is the resultant increase in the order

of equation (3). Since the range of validity of the approximation improves only marginally for a significant increase in the model order, it is computationally unattractive to select ℓ larger than 1.

Another important alternative is to minimize a frequency-domain objective function to determine the coefficients of the polynomials N(τ's) and D(τ's) of prespecified orders (Lilja, 1985). These coefficients, n_i, d_i, are obtained as

$$(n_i, d_i) = \min_{n_i, d_i} \sum_{k=1}^{M} \left| e^{-j\omega_k} D(j\omega_k) - N(j\omega_k) \right|^2 W(j\omega_k) \qquad (6)$$

where ω_k are M number of different frequencies, each with a weighting factor $W(j\omega_k)$, spanning the range of $\omega\tau$' for which the approximation represented by (n_i,d_i) is sought to be valid in a least-squares sense. This method offers the flexibility of extending the range of applicability of the approximation in equation (4) beyond that possible with a Padé approximation of the same order, at the expense of achieving poorer match in both the amplitude and the phase compared to the exact match provided by the Padé approximation over its narrower range. Again, it is found in this case that, for a fixed order, the approximation quality deteriorates substantially for a marginal increase in the range of $\omega\tau$' for which it is valid.

Hence, it is clear that different choices of the approximation in equation (4) offer trade-offs between its range of validity, the goodness of its match, and the increase in the order of equation (3). The selection of a particular choice will be dictated by considerations of the specific process application; in particular, the largest expected absolute value of τ' and the process frequency range that the

model is expected to emulate will determine how complex the approximation needs to be. However, as will become clear later, slow variations in τ_d over a practically unlimited range can be tracked without straining the expected value of τ', so that even a very simple choice of the approximation, as, for instance, the first-order Padé approximation, should suffice in most cases.

Model reformulation

Indeed, the first-order Padé approximation will be used in the development to follow, in order to preserve clarity of presentation. No generality is lost, of course, and the technique is applicable for any other form of approximation selected.

Using the approximation in equation (5), for ℓ=1, to substitute for $e^{-\tau' s}$ in equation (3) gives

$$\left[1 + \sum_{i=1}^{n} a_i s^i\right] y(s) + Y(0) = \sum_{i=0}^{m} b_i s^i \left[u(s) e^{-\tau_o s}\right] \left[\frac{1 - 0.5\tau' s}{1 + 0.5\tau' s}\right] + U(0) \quad (7)$$

The use of equation (7) for parameter estimation requires knowledge of signals that are time-derivatives of y(t) and u(t). Since these signals cannot be usually measured in an easy, reliable manner, they will be constructed using the multifilter technique employed before by Young (1965, 1969) and Eykhoff (1974) for off-line estimation and by Canudas (1985) for on-line estimation. Using a dynamic operator, F(s), on both sides of equation (7) and defining

$$\begin{aligned} y_i(s) &\equiv s^i y(s)F(s) \quad \forall\, i \in [1,n] , \quad Y_o(s) \equiv Y(0)F(s)(1 + 0.5\tau' s) \\ u_i(s) &\equiv s^i u(s)F(s) \quad \forall\, i \in [0,m] , \quad U_o(s) \equiv U(0)F(s)(1 + 0.5\tau' s) \end{aligned} \tag{8}$$

the equation (7) becomes

$$\begin{aligned} & y_o(s) + (a_1+\tau)\, y_1(s) + \sum_{i=2}^{n} (a_i+\tau a_{i-1})\, y_i(s) + \tau a_n y_{n+1}(s) + Y_o(s) \\ & = b_o u_o(s) e^{-\tau_o s} + \sum_{i=1}^{m} (b_i - \tau b_{i-1})\, u_i(s) e^{-\tau_o s} - \tau b_m u_{m+1}(s) e^{-\tau_o s} + U_o(s) \end{aligned} \tag{9}$$

where τ is defined as $0.5\tau'$. Reverting to the time domain to enable estimation, the inverse Laplace transformation of equation (9) yields

$$y_o(t) + \sum_{i=1}^{n+1} \alpha_i y_i(t) + \mathcal{L}^{-1}\,[Y_o(s)] = \sum_{i=0}^{m+1} \beta_i u_i(t-\tau_o) + \mathcal{L}^{-1}\,[U_o(s)] \tag{10}$$

where

$$\begin{aligned} \alpha_1 &= a_1 + \tau , \quad \alpha_i = a_i + \tau a_{i-1} \;\forall\, i \in [2,n] , \quad \alpha_{n+1} = \tau a_n \\ \beta_o &= b_o , \quad \beta_i = b_i - \tau b_{i-1} \;\forall\, i \in [1,m] , \quad \beta_{m+1} = -\tau b_m \end{aligned} \tag{11}$$

The initial conditions appearing in equation (10) will vanish as time increases, provided that the operator F(s) is stable, which will be ensured by its choice. It is also worth noting that for negative values of τ' the initial conditions will take longer to disappear due to the nonminimum-phase character evident from the definitions in equation (8).

Use of the model form in equation (10) for recursive parameter estimation requires sampling the filtered signals at discrete instants kh, k=0,1,2... The sampling period, h, should be chosen small enough to avoid any significant

overlap of the sampled frequency bands of the process. Sampling leads to a discrete model suitable for parameter estimation,

$$y_o(k) = -\sum_{i=1}^{n+1} \alpha_i y_i(k) + \sum_{i=0}^{m+1} \beta_i u_i(k - \tau_o/h) \qquad (12)$$

where the initial conditions have been neglected for the above-mentioned reason. The signals $u_i(k-\tau_o/h)$ can be obtained by sampling with time-period h the delayed signals $u_i(t-\tau_o)$. In principle, these signals can be delayed using specific-purpose routines, although perhaps at extra computational burden. Thus, it would be usually advisable for the user to specify τ_o as an integer multiple of the sampling period, h, even though the development presented here does not require this restriction.

3. PARAMETER ESTIMATION

Several methods can be used to estimate the parameters a_i, b_i, τ, using equations (11) and (12). Here we restrict our attention to the well-known class of least-squares algorithms that seek to minimize the objective function

$$V_L(\theta) = \frac{1}{L} \sum_{k=1}^{L} \left[y_o(k) + \sum_{i=1}^{n+1} \alpha_i(\theta) y_i(k) - \sum_{i=0}^{m+1} \beta_i(\theta) u_i(k-\tau_o/h) \right]^2 \qquad (13)$$

with respect to the parameter vector, θ, based on information available upto the current sampling instant, L. Two algorithms utilizing different definitions of the vector θ are presented below.

Linear estimation scheme

In this scheme the parameter vector, θ, is defined as (α_i, β_i) rendering equation (13) linear with respect to the elements of θ. Standard recursive least-squares (RLS) algorithm is, therefore, employed to obtain estimates of α_i, β_i. The goal of obtaining the parameters a_i, b_i, τ, from these estimates of α_i, β_i, then requires a nonlinear transformation according to equation set (11).

When the estimates of α_i, β_i, have converged to their true values, this nonlinear transformation will, in general, be unique. (The pathological cases where a_i, b_i, τ, will not be unique are analysed in Section 4.) One simple way to perform this unique transformation is to obtain from the equation set (11),

through algebraic manipulation, the polynomials of the form

$$\tau^{n+1} + \sum_{i=1}^{n+1} (-1)^i \alpha_i \tau^{n+1-i} = 0 \qquad (14)$$

$$\sum_{i=0}^{m+1} \beta_i \tau^{m+1-i} = 0 \qquad (15)$$

The unique value of τ is identified as the common root between these polynomials. This common root can be readily determined by reciprocal substitution of the lower-order polynomial into the higher-order polynomial until the order of both is reduced to unity. Having deduced τ, the parameters a_i, b_i, are easily obtained from equation set (11).

When the estimates of α_i, β_i, are in transient condition, the nonlinear transformation to a_i, b_i, τ, will, in general, not even exist since all the roots of the polynomial in (14) might be different from the roots of the polynomial in (15); indeed, some or all of these roots could be complex. When this transformation happens to exist, it might well render values of a_i, b_i, τ, far from their true values even though α_i, β_i, may represent reasonable transient estimates.

The lack of availability of any estimates of a_i, b_i, τ, during transient conditions involving parameter changes is the only drawback of this estimation scheme. This will not be of serious consequence in applications that do not make use of transient parameters and rely on the old estimates until the new estimates have converged. Such is the case, for instance, in many control applications that base the controller parameters on the estimates of the process parameters, requiring a transformation that is often unreliable when the estimates are in

transience.

It is worth mentioning here that the estimates of α_i, β_i, can be used directly in equation (7) when the model expressed by the Laplace inverse of this equation would suffice for the user's purpose. Of course, in the event that the process dynamics represented by a_i, b_i, are known, the linear estimation scheme, with τ as the sole element in θ, can be readily employed to estimate the time delay.

Nonlinear estimation scheme

In this scheme the parameter vector, θ, is defined as (a_i, b_i, τ) so that equation (13) becomes nonlinear with respect to the elements of θ. A recursive nonlinear least-squares algorithm, presented by Goodwin and Sin (1984), is then applied to obtain direct estimates of a_i, b_i, τ. The algorithm takes the form

$$\hat{\theta}(k) = \hat{\theta}(k-1) + P(k-1)\Psi(k-1)[y_o(k) - \hat{y}_o(k,\hat{\theta}(k-1))] \tag{16}$$

$$P(k-1) = P(k-2) - \frac{P(k-2)\Psi(k-1)\Psi(k-1)^T P(k-2)}{\lambda + \Psi(k-1)^T P(k-2)\Psi(k-1)} \tag{17}$$

where

$$\Psi(k-1) \equiv \left. \frac{d\hat{y}_o(k,\theta)}{d\theta} \right|_{\theta = \hat{\theta}(k-1)}$$

and $\hat{y}_o(k,\theta)$ is defined as the right-hand side of equation (12) with α_i, β_i, denoted as functions of θ. Here $\hat{\theta}(k)$ represents the estimate of the vector θ at instant k, and P(k) can be recognized as the symmetric covariance matrix and λ as the

forgetting factor analogous to the linear RLS algorithm.

Again, in this case, the objective function in equation (13) has a unique minimum, except in pathological situations discussed in Section 4. Hence, if the initial values of the parameters are good enough for the algorithm to converge, it will converge to the unique values of a_i, b_i, τ.

Choice of F(s)

The motivation for choice of the operator F(s) derives from the cost function in equation (13) expressed in the frequency domain as

$$V(\theta) = \int_{-\pi/h}^{+\pi/h} \left| G^o(j\omega)\left[1 + \sum_{i=1}^{n+1}(j\omega)^i\alpha_i\right] - \sum_{i=0}^{m+1}(j\omega)^i\beta_i \right|^2 \left| F(j\omega) \right|^2 \Phi_u(\omega)d\omega \quad (18)$$

where $G^o(s)$ is the true process transfer function, and $\Phi_u(\omega)$ is the spectral density of the input signal. It is clear from equation (18) that the operator F(s) acts as a weighting function that dictates the relative importance of matching the model to the true process at any particular frequency. The estimation algorithms will identify parameters that will enable the model to describe the process accurately in the frequency range where the magnitude of $F(j\omega)$ is high, while representing the process poorly in the frequency range where this magnitude is low.

The opposing factors influencing the choice of F(s) surface clearly from this interpretation. On one hand, it is desirable to suppress high-frequency information while seeking an accurate model representation only in the low-frequency range in which τ' can assume relatively larger absolute values without leading the approximation in equation (4) to jeopardize the sought accuracy of model representation. This consideration suggests a choice of $F(j\omega)$ with low magnitude for high frequencies. On the other hand, the user may select an ambitious high-order model if the application requires identification of some fast process modes as well. This intention would be frustrated unless the magnitude of $F(j\omega)$ is chosen to be significant at the frequencies of these fast modes.

Having decided the highest frequency, ω_p, for which the model is sought to represent the process accurately, the ideal choice of $F(j\omega)$ would be uniform magnitude for frequencies upto ω_p and zero thereafter. In practice, however, F(s) is selected as a linear filter of the form $c^f/(s+c)^f$, with $f \geq n+1$, in order to enable construction of the signals $y_i(t)$, $u_i(t)$, by the multifilter technique. Thus, the magnitude of $F(j\omega)$ is constant for frequencies upto c and diminishes thereafter at a rate dependent on f. For reasonably fast suppression of the higher frequencies which are not of interest, it is advisable to choose $f>n+1$, although too high a value of f may be computationally unattractive. With f specified, the cut-off frequency, c, is selected at the smallest value that would ensure a significant magnitude of $F(j\omega)$ at the high frequency ω_p. A lower value of c would render unidentifiable any process modes close to ω_p, and a higher value of c would narrow the range of τ' that can be successfully identified.

Equation (18) also shows that even over the frequency range where the magnitude of $F(j\omega)$ is uniform, the estimated model will represent the process more accurately for frequencies at which the magnitude of $G^o(j\omega)$ or the spectral density $\Phi_u(\omega)$ are relatively high. Thus, in essence, the product $|G^o(j\omega)|^2|F(j\omega)|^2\Phi_u(\omega)$ represents the true weighting for a given frequency. Having prescribed the filter parameters, c, f, and with some idea of the amplitude of the input signal and a rough notion of the band-width of the process frequency spectrum, it should be possible to deduce an upper bound on frequency, ω_{max}, above which the amalgam of the frequency spectra of the process, the input, and the filter would be expected to have an insignificant magnitude. Then, ω_{max} will correspond to a conservative estimate of the band-width of the combined spectrum. Knowing the range of $\omega\tau'$ for which the chosen approximation in equation (4) is valid, this estimate of ω_{max} can be translated into a value τ^*, a conservative estimate of the maximum absolute τ' which would be expected to preserve the accuracy of the approximation, enabling unbiased estimation.

The value of τ^* reflects only upon the rate of time-delay variation that can be tracked successfully, and not, as will be clear later, upon the range of this variation. The value of τ^* and an experience-based notion of the algorithm's convergence rate will indicate the secure rate of time-delay variation. If the process time delay is expected to vary much faster than this rate, then either a less ambitious ω_p must be specified or a more complex approximation in equation (4) must be chosen.

Choice of the input signal

Use of a periodic input is inadvisable for estimation of the time delay. If a periodic signal is used, care must be taken to ensure that the time-period of the signal is large compared to the relevant absolute values of τ'. Moreover, the uniqueness of the minimum of the objective function in equation (13) is contingent upon choice of an input for which the required signals u(t), $\dot{u}$(t), etc., are linearly independent. For instance, a choice of $u(t) = e^{-\nu t}$, where ν is a constant, is clearly detrimental in this respect.

The amplitude of the input signal is important, too. It is best to select a signal with high magnitude for frequencies upto ω_p and low magnitude at higher frequencies. If the density spectrum of the input is more uniform over frequencies, a high amplitude will facilitate faster convergence, but will also tend to render more significant the frequencies above ω_p, leading to higher ω_{max}, and lower τ^*.

Example 1

A first-order process with time delay is simulated. The process is identified using a model of the form

$$G(s) = \frac{b_o}{a_1 s + 1} e^{-2.5s} \left[\frac{1 - 0.5\tau' s}{1 + 0.5\tau' s}\right]$$

Here, the time delay is expected to be in the vicinity of 2.5, and the first-order

Padé approximation is used for $e^{-\tau' s}$. With the intention of identifying frequencies upto $\omega_p = 2.0$, the filter parameters are chosen as c = 1.5, f = 3. The input signal is chosen to be periodic square-wave with an amplitude of 0.1 around a mean of 0.1. The time-period of the input is chosen as 10 so that it is much greater than the expected value of τ'. With these choices and the expected first-order process frequency spectrum, the value of ω_{max} is reckoned at 4.0, beyond which the overall magnitude of the combined spectrum would be expected to be insignificant. Thus, practically zero bias would be expected for absolute values of τ' upto at least $\tau^* = 0.25$, for the chosen approximation of $e^{-\tau' s}$.

The sampling period is chosen to be 0.1 so that the sampling frequency is larger than $2\omega_{max}$ ensuring insignificant overlap between the bands of the sampled frequency spectrum. Recursive nonlinear least-squares algorithm is used. The diagonal elements of the covariance matrix are initialized to 10^3 and a forgetting factor of 0.98 is specified expecting slow variations in the parameters. The initial guess for $\hat{a}_1$ is conservatively taken as 1.5. For lack of knowledge of even the sign of b_o, the initial guess of $\hat{b}_o$ is taken to be zero. Since the best guess of the time delay is 2.5, the initial value of $\hat{\tau}'$ is set to zero.

Figure 1 shows the parameter estimates obtained when the process is simulated with constant $a_1 = 1.0$ and $b_o = 0.5$, and with τ_d taking the values 1.7, 3.0, 2.2, 3.8 at times 0, 20, 40, 60, respectively. The estimates converge without bias in the third leg where $|\tau'| = 0.3$, and exhibit increasing bias for larger absolute values of τ'. In the final leg, the value of $\tau' = +1.3$ clearly seems to exert excessively the approximation of $e^{-\tau' s}$ used in the model, leading to unacceptable bias in estimates.

Figure 2 shows the estimates obtained when a_1 is changed from 1.0 to 1.5 at time 20, and b_o is changed from 0.5 to 0.3 at time 50, while τ_d stays constant at 2.0. It is observed that the estimation scheme tracks changes in the process dynamics effectively, without excessive drift in the estimate of the time delay. The slower process mode in the second leg reduces the effective value of ω_{max}', which makes the Padé approximation valid for a larger absolute τ', leading to a slight reduction of bias in the estimates.

The stepping mechanism

This mechanism enables attainment of unbiased estimates over a practically unlimited range of the slowly-varying time delay, enhancing immensely the capability of the proposed algorithm. Its philosophy lies in a reduction of the effective value of $|\tau'|$ by changing on-line the user-specified parameter τ_o.

Upon calculation of the new parameter estimates at each sampling instant, a user-specified criterion is used to test whether the parameters have converged. One such criterion requires the absolute relative change over two successive instants for each of the estimated parameters to be less than a prespecified small value, ϵ. Thus, convergence is inferred at instant k_c, if

$$\left| \frac{\hat{\theta}_i(k_c) - \hat{\theta}_i(k_c - 1)}{\hat{\theta}_i(k_c)} \right| < \epsilon \qquad (19)$$

for each of the elements, θ_i, of the parameter vector, θ. Recall that for the linear estimation scheme such a test would be performed anyway, before calculation of

$\hat{a}_i$, $\hat{b}_i$, $\hat{\tau}$, from the converged estimates $\hat{\alpha}_i$, $\hat{\beta}_i$.

When convergence is detected at instant k_c, the values of τ_o and $\hat{\tau}$ are changed as

$$\tau_o = \hat{\tau}_d(k_c) \quad , \qquad \hat{\tau}(k_c) = 0 \tag{20}$$

Thus, in the successive instants, the filtered signals u_i used in the estimation algorithm are delayed by the new value of τ_o. As discussed in Section 2, if τ_o has been specified as an integer multiple of the sampling time, h, this restriction can be continued, without loss of generality, and the new τ_o can be chosen as the integer multiple of h closest to $\hat{\tau}_d(k_c)$. In this case, $\hat{\tau}(k_c)$ will be modified to a small value such that $\tau_o + \hat{\tau}(k_c) = \hat{\tau}_d(k_c)$.

At the instant k_c, the value of $\hat{\tau}$ in the estimation algorithm is aritificially set to zero, or to the small value mentioned above. For the nonlinear estimation scheme, this is accomplished directly by changing the value of the corresponding element in the parameter vector, $\hat{\theta}$. For the linear estimation scheme, the changed value of $\hat{\tau}$ and the estimated values of $\hat{a}_i$, $\hat{b}_i$, are used in equation set (11) to obtain the values of $\hat{\alpha}_i$, $\hat{\beta}_i$, to which the elements of the parameter vector $\hat{\theta}$ are to be set.

It should be noted that the above changes are not to be accompanied by any other change in the elements of the estimation scheme. In particular, the diagonal elements of the covariance matrix should not be reset. This ensures that the estimation algorithm is not perturbed unduly and the converged state is preserved if the estimate of τ_d was close to its true value.

If, at instant k_c, the estimates were biased due to $|\tau'|$ being too large for the approximation in equation (4) to be valid, the changes in equation (20) have the effect of reducing $|\tau'|$ and making the approximation more accurate. The parameter estimates will then show another transience before converging to less biased estimates. Successive applications of equations (19) and (20) will ensure achievement of unbiased estimates as τ_o steps to the true value of τ_d.

When the process time delay undergoes a large change, the value of τ' may be much greater than the value of τ^* for which the approximation is expected to be valid. In this case, it may be inefficient to wait for the convergence in equation (19) if the convergence rate is slow. However, so long as the change in τ_d is gradual, $\hat{\tau}$ will move in the correct direction during transience. Then, improved performance can be achieved by executing the changes in equation (20) at intermediate instants prior to k_c, whenever the absolute value of $\hat{\tau}$ exceeds a prespecified multiple of τ^*. Of course, this option is not available for the linear estimation scheme, which does not yield values of $\hat{a}_i$, $\hat{b}_i$, $\hat{\tau}$, during transience.

Example 2

To demonstrate the power of the modified algorithm, the same system is used as in Example 1. In this case, the initial τ_o is specified to be 2.0. The process parameters a_1, b_o, stay constant at 1.0, 0.5, respectively, while τ_d takes the values 3.0, 4.0, 5.0, at times 0, 20, 40, respectively. All other conditions are identical to Example 1. Figure 3a shows the parameter estimates obtained with fixed τ_o. It is apparent that the estimates become progressively more biased as τ_d

drifts farther away from τ_o. Figure 3b shows the effect of utilizing the stepping mechanism with a value of 1% specified for ϵ in criterion (19). Unbiased estimates result as τ_o successively steps toward τ_d, extending the validity of the Padé approximation over a practically unlimited range. In the simulation, τ_o was updated to the integer multiple of sampling time closest to $\hat{\tau}_d$, for the reason mentioned above.

Example 3

A more complex system is simulated, represented by the transfer function

$$G^o(s) = \frac{-0.5(1 - 0.7s)}{(1.0s + 1)(0.4s + 1)} e^{-\tau_d s}$$

with τ_d taking the values 1.7, 1.3, 1.0, 1.8, 2.8, at times 0, 20, 40, 60, and 80, respectively. A model with n = 2, m = 1, and with first-order Padé approximation for $e^{-\tau' s}$, was used to identify this process. The initial τ_o was taken to be 1.5, and the estimates of all parameters were initialized to zero. The filter parameters were chosen as c = 3.0, and f = 4. The stepping mechanism was used with all other conditions being identical to those used in Example 2. The results are shown in Figure 4, with the values of τ_o omitted for sake of clarity. Excellent, unbiased estimates are achieved, even though the variations in time delay far exceed the value of τ^*. (The last leg shows no bias upon convergence after some time.)

4. SPECIAL CASES

In the previous section, it was mentioned that the estimation schemes might not converge to true parameter values in certain special cases. These cases, discussed here, reveal themselves to be rather benign. The discussion is facilitated by writing the transfer function of the true process in reduced form as

$$G^o(s) = \frac{b'(s + z_1)(s + z_2)\ldots.(s + z_{m'})}{(s + p_1)(s + p_2)\ldots.(s + p_{n'})} \; e^{-\tau' s} \; e^{-\tau_o s} \qquad (21)$$

Pole-zero cancellation

It is clear from equation (21) that the cancellation of a pole or a zero is not possible unless the factor $e^{-\tau' s}$ can be represented by a finite number of poles and zeroes analogous to equation (4). Any such representation is valid only upto a certain value of $\omega\tau'$ above which the representation is inadequate and cannot lead to any cancellation with the process poles and zeroes.

Consider, for the moment, a first-order Padé representation of $e^{-\tau' s}$ in equation (21), valid for values of $\omega\tau'$ less than 1. It would seem, then, that so long as τ' is small enough for this representation to be valid, the zero or pole of this representation could conceivably cancel a pole or a zero of the process. Reflection shows, however, that this situation cannot occur. Consider the case where the pole of the first-order Padé representation of $e^{-\tau' s}$ in equation (21)

cancels the i^{th} zero, z_i, of the process. In order for this to be possible, the Padé representation would have to be valid at least upto the frequency $|z_i|$. This, in turn, implies that $|\tau'|$ must be less than $1/|z_i|$. The pole of the Padé representation, $1/(0.5|\tau'|)$, would have to be larger than $|z_i|/0.5$ and, consequently, cannot cancel the proposed zero, z_i. Considerations of higher-order representations of $e^{-\tau's}$ in equation (21) lead to even bigger discrepancies of this nature. Hence, cancellation of a process pole or zero is not possible.

Presence of impostors

Equation (21) shows that if a combination of process zeroes and poles mimics the form of approximation (4) used in the model, then the estimation schemes will not be able to differentiate between the true value of τ and the fictitious value, τ_f, represented by that combination. For example, if the first-order Padé approximation is used in the model, then the presence of one or more pairs of zeroes and poles of the form $(1-\tau_f s)/(1+\tau_f s)$ in the process transfer function will offer that many incorrect but equally viable solutions to the estimation scheme. Then, the objective function in equation (13) will exhibit more than one minimum, and the uniqueness of solution will be lost.

Clearly, the higher the order of the approximation selected in equation (4), the less likely will be the existence of a closely resembling form in the process transfer function. Moreover, since all reasonably accurate approximations of $e^{-\tau's}$ will include at least one factor of the form $(1-c_1\tau's)/(1+c_2\tau's)$, where c_1, c_2, are positive constants, no resembling equivalent is possible in the process transfer

function for stable, minimum-phase processes.

Consider a nonminimum-phase process with transfer function containing a factor of the form $(1-\tau_f s)/(1+\tau_f s)$, $\tau_f>0$. Suppose that first-order Padé approximation is used in the model. A second solution corresponding to $\hat{\tau}' = 2\tau_f$ will exist only if the objective function in equation (18) has a significant weighting for the frequency $1/\tau_f$, i.e., only if $\omega_{max} \geq 1/\tau_f$. Now if the true solution τ' is within the range of τ^*, where $\tau^* = 1/\omega_{max}$ is the maximum absolute value of τ' for which the Padé approximation is valid, then $|\tau'| \leq \tau^* \leq \tau_f$. Thus the spurious solution, $\hat{\tau}' = 2\tau_f$, will be at least twice the value of the true solution if the latter lies within τ^*. For higher-order approximations in equation (4), this difference is even larger.

The true solution will be expected to lie within τ^* when the process parameter variations are slow and the stepping mechanism described previously is being used. Hence, if the initial parameter estimates are close to their true values, the estimation algorithm will continue to converge to the true solution and will not drift to the other minimum. For the linear estimation scheme, this will be ensured by selecting as the true solution the negative or the smaller positive root out of the two common roots that will be obtained between polynomials (14) and (15) after reducing each of them to order 2.

Hence, the possibility of identifying the wrong solution exists only for nonminimum-phase processes, when the process parameter variations are not slow or during the initial phase of estimation. Even in these singular situations, it is straightforward to force convergence to the correct solution, as outlined

below.

For the linear estimation scheme, upon convergence, the existence of two common roots between polynomials (14) and (15) will indicate presence of a spurious solution. Upon detecting this situation, the value of τ_o can be changed by an amount less than τ^*, and parameters allowed to reconverge. Of the two new common roots obtained between (14) and (15) from the newly converged parameters, the one that is almost identical to one of the previous solutions can be identified as the incorrect solution and rejected. The other new root would be different from the other previous solution by an amount roughly equal to the change made in τ_o, and this root represents the true solution.

For the nonlinear estimation scheme, upon convergence to one particular solution $(\hat{a}_i,\hat{b}_i,\hat{\tau})$, the existence and value of the other solution can be determined by substituting in polynomials (14) and (15) $\alpha_i = \hat{a}_i$, $\alpha_{n+1} = 0$, $\beta_i = \hat{b}_i$, $\beta_{m+1} = 0$, and finding the common root between them in the usual way. If a common root is found, it represents the value of τ corresponding to the other solution. The selection between the two solutions is then made, in the same manner as for the linear estimation scheme, by changing τ_o. Upon identifying the correct solution, the estimates $\hat{a}_i$, $\hat{b}_i$, $\hat{\tau}$, can be set to these values and τ_o moved closer, if necessary.

Analogous procedures can be used, if a more complex approximation has been selected for equation (4).

Underparameterized model

When the model orders are not large enough to account for all the process modes within the frequency range for which the weighting in equation (18) is significant, the identified slower modes and the identified τ' will be different from their true values. However, in spite of seemingly erroneous estimates, the identified model will give the closest possible representation of the true process, as guaranteed by equation (18). The following example best illustrates this.

Example 4

A second-order process is simulated, having the transfer function

$$G^o(s) = \frac{-0.5}{(1.0s + 1)(0.4s + 1)} e^{-2.2s}$$

A constant value of 2.0 is assumed for τ_o, so that τ' is only 0.2 and the stepping method is not employed. All other conditions of identification are identical to those in Example 3. Figure 5a shows the parameter estimates for a model with n = 2, m = 0, and first-order Padé approximation for $e^{-\tau' s}$. All parameters are seen to converge to their true values. A reduced-order model with n = 1, m = 0, was then used for identification under identical conditions. Figure 5b shows that, in this case, the converged value of $\hat{a}_1$ is 1.11, and that of $\hat{\tau}'$ is 0.52. This is exactly what one would expect, considering the well-known approximation of a second-order system as a first-order system with time delay. Thus, even though the estimated time delay and time constant are different from their true values,

the identified model represents the real process as closely as possible given its inferior structure.

Presence of noise

Uncertainties caused by the presence of measurement or process noise can cause poorer estimation. Although the development above has been presented for a deterministic process, it is clear from equation (18) that the noise will be perceived by the estimation algorithm only after passing through the filter, F(s). Thus, zero-mean noise of frequency greater than ω_{max} is expected to cause no observable deterioration in parameter estimates. Lower frequency noise will bias the parameter estimates more significantly as the weighting in equation (18) becomes larger at that frequency. The following example helps to illustrate the extent of deterioration that may result.

Example 5

Measurement noise is added to the process simulated in Example 2. Runs are made under identical conditions as for the stepping-method identification shown in Figure 3b. Discrete white noise with standard deviation equal to 10% of the amplitude of the process output was added to the sampled output before reception by the estimation algorithm. No observable change in the performance compared to the run in Figure 3b occurred when the noise was introduced at every sampling instant. Figures 6a and 6b show, omitting τ_o for clarity, the

parameter estimates when the noise was introduced every third and tenth sample, respectively. As expected, the performance deteriorates as the noise frequency decreases. The high-frequency noise at the sampling frequency is clearly beyond the range of ω_{max}, and is, therefore, essentially not perceived by the estimation scheme. The noise at the frequency of 3.3 is still well-over the filter cut-off frequency of 1.5, and is dampened significantly as a result. Thus, the performance in Figure 6a is only slightly poorer. The noise at the frequency of 1.0 interferes directly with the process mode, without any curtailment by the filter. This leads to significant deterioration of the parameter estimates, as shown in Figure 6b.

5. CONCLUSIONS

A conceptually straightforward technique is presented for on-line estimation of constant or slowly-varying time delay and continuous-time parameters. Earlier works with similar objectives, proposed in the literature, suffer from one or more of the following drawbacks--limitation to off-line identification, problems due to creation of multiple extrema of the objective function, and estimation of time delay only to an integer multiple of the sampling time. The method presented here avoids these drawbacks.

A frequency-domain transformation of the continuous-time model of a linear, deterministic, SISO system is subjected to approximation of the unknown time-delay variation by a rational transfer function. Several possible approximations are indicated. The derivative signals appearing in the time-domain transformation of the approximate model are then constructed from the process output and input by using a multifilter technique previously proposed in the literature. After sampling, a discrete model results that is nonlinear with respect to the desired parameters.

Two alternatives for estimation are proposed. One involves use of linear RLS algorithm followed by a nonlinear algebraic transformation, but suffers from the drawback that the desired parameters are not available during transient conditions of time-delay and parameter changes. The other estimation method overcomes this drawback by utilizing a previously proposed recursive nonlinear least-squares algorithm that yields directly the estimates of the desired

parameters. Both schemes lead to the true, unique solution, in general.

A stepping mechanism is then developed which permits tracking of the time delay over an unlimited range, despite the limitations imposed by the approximation used for the time-delay variation. It achieves this by modifying on-line the known part of the time delay, so that the unknown variations around the known value stay within reasonable bounds. Several simulation examples are presented to illustrate the capability of the proposed technique.

Finally, it is shown that the algorithm is immune to possibility of pole-zero cancellations, yields best possible estimates when the chosen model is structurally inferior, and is robust to the presence of high-frequency measurement or process noise. The singular case of presence of multiple solutions is shown to be possible only for nonminimum-phase processes and only under extreme conditions; simple procedures are presented for detecting and circumventing this problem if it occurs.

The method offers various choices of implementation, allowing flexibility with respect to the complexity of the process application.

Acknowledgements--We are indebted to Prof. K. J. Åström for graciously extending us the opportunity to work at LTH. Thanks are due also to our colleagues in the department for their continued interest and encouragement. Agarwal is grateful to Prof. D. E. Seborg for making leave of absence possible, with support from NSF Engineering Research Center grant to UCSB. Canudas acknowledges support from CONACYT (Mexico), CEFI and GRECO-Systemes adaptatifs (France), and his adviser Prof. I. D. Landau.

REFERENCES

Abrishamkar, F. and G. A. Bekey (1985). Estimation of time varying delays in linear stochastic systems. Proc. 7th IFAC Symposium on Identification and System Parameter Estimation, York, UK.

Åström, K. J. and B. Wittenmark (1984). Computer Controlled Systems--Theory and Design, Prentice-Hall, Englewood Cliffs, NJ.

Åström, K. J. and B. Wittenmark (1985). The self-tuning regulator revisited. Proc. 7th IFAC Symposium on Identification and System Parameter Estimation, York, UK.

Bohn, E. V. (1985). Walsh function decoupled parameter estimation equations for dynamic continuous-time models with time-delay. Proc. 7th IFAC Symposium on Identification and System Parameter Estimation, York, UK.

Bokor, J. and L. Keviczky (1985). Recursive structure, parameter and delay time estimation using ESS representations. Proc. 7th IFAC Symposium on Identification and System Parameter Estimation, York, UK.

Canudas, C. (1985). On process parameter identification. Internal report TFRT-7290, Dept. of Automatic Control, Lund Institute of Technology, Lund, Sweden.

Eykhoff, P. (1974). System Identification. Wiley-Interscience Publication.

Goodwin, G. C. and K. S. Sin (1984). Adaptive Filtering Prediction and Control. Prentice-Hall, Inc.

Kaminskas, V. (1979). Parameter estimation in systems with time delay and closed loop systems. Preprints 5th IFAC Symposium on Identification and System Parameter Estimation, Darmstadt, vol. 1, 669-677.

Kurz, H. and W. Goedecke (1981). Digital parameter-adaptive control of processes with unknown dead time. Automatica, 17(1), 245-252.

Lee, T. H. and C. C. Hang (1985). A performance study of parameter estimation schemes for systems with unknown dead time. Proc. ACC, Boston, MA.

Lilja, M. (1985) Private communication.

Marchand, M. and Fu K-h (1985). Frequency domain parameter estimation of aeronautical systems without and with time delay. Proc. 7th IFAC Symposium on Identification and System Parameter Estimation, York, UK.

Pupeikis, R. (1985). Recursive estimation of the parameters of linear systems with time delay. Proc. 7th IFAC Symposium on Identification and System Parameter Estimation, York, UK.

Rao, G. P. and L. Sivakumar (1976). Identification of deterministic time-lag systems. IEEE Trans. Auto. Control, August, 1976, 527-529.

Song, W. and X. Yu (1985). Structure and parameter identification for a kind of multivariable linear systems with unknown time delays. Proc. 7th IFAC Symposium on Identification and System Parameter Estimation, York, UK.

Young, P. C. (1965). Process Parameter Estimation and Self-adaptive Control System (Ed. Hamnond), Plenum Press, New York.

Young, P. C. (1969). Applying parameter estimation to dynamic systems. Control Engng., 16, Oct. 119-125, Nov. 118-124.

FIGURE CAPTIONS

Figure 1: Parameter estimates for Example 1 for changes in time delay

Figure 2: Parameter estimates for Example 1 for changes in process dynamics.

Figure 3: Parameter estimates for Example 2, (a) with constant τ_o, and (b) with τ_o changing by the stepping mechanism.

Figure 4: Parameter estimates for Example 3.

Figure 5: Parameter estimates for Example 4, (a) for a second-order model, and (b) for a first-order model.

Figure 6: Parameter estimates for Example 5, (a) with medium-frequency noise, and (b) with low-frequency noise.

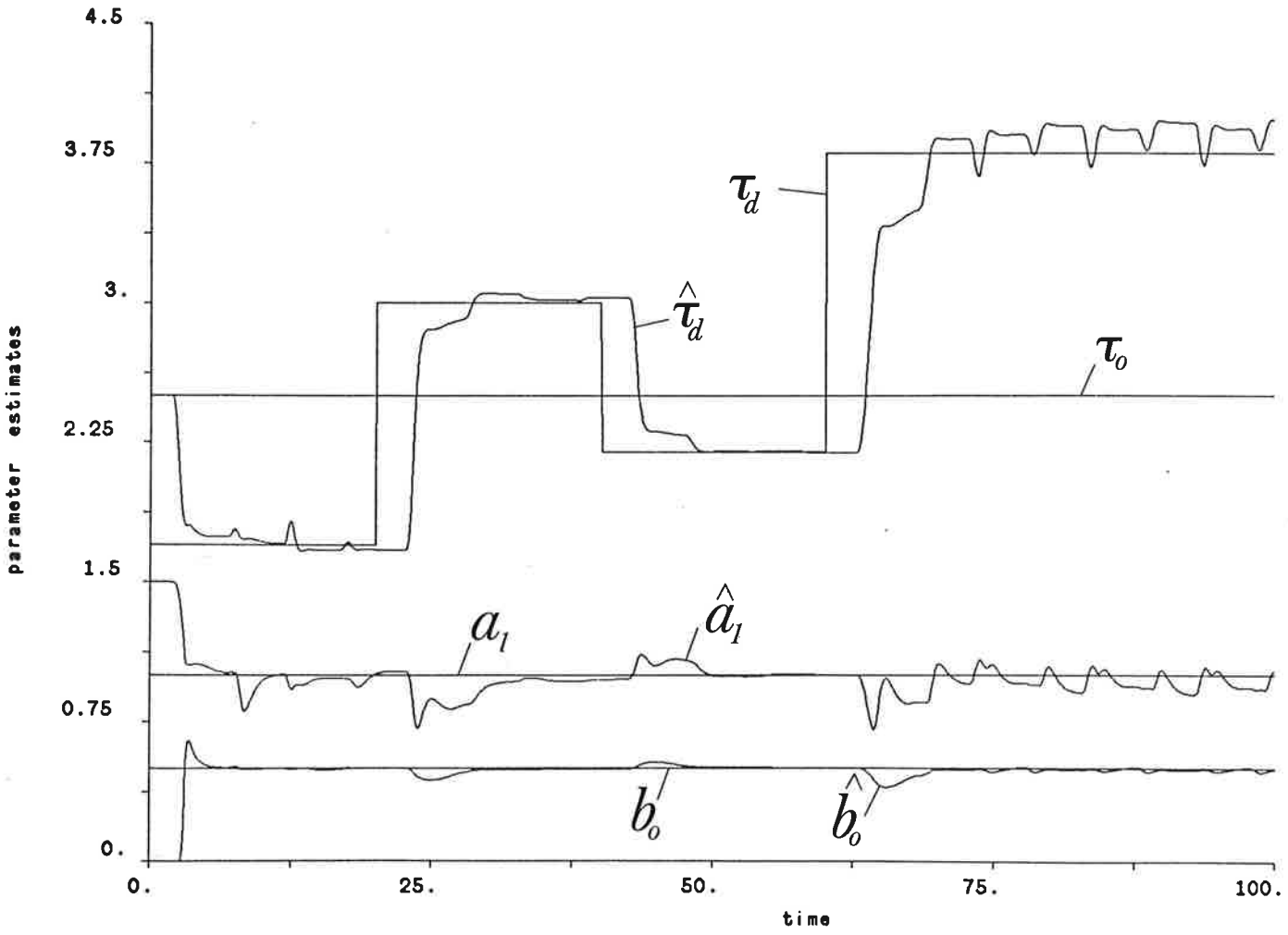

4.5
3.75
3.
2.25
1.5
0.75
0.
parameter estimates
0.
25.
50.
75.
100.
time
τ_d
$\hat{\tau}_d$
τ_o
a_1
$\hat{a}_1$
b_o
$\hat{b}_o$

Figure 1

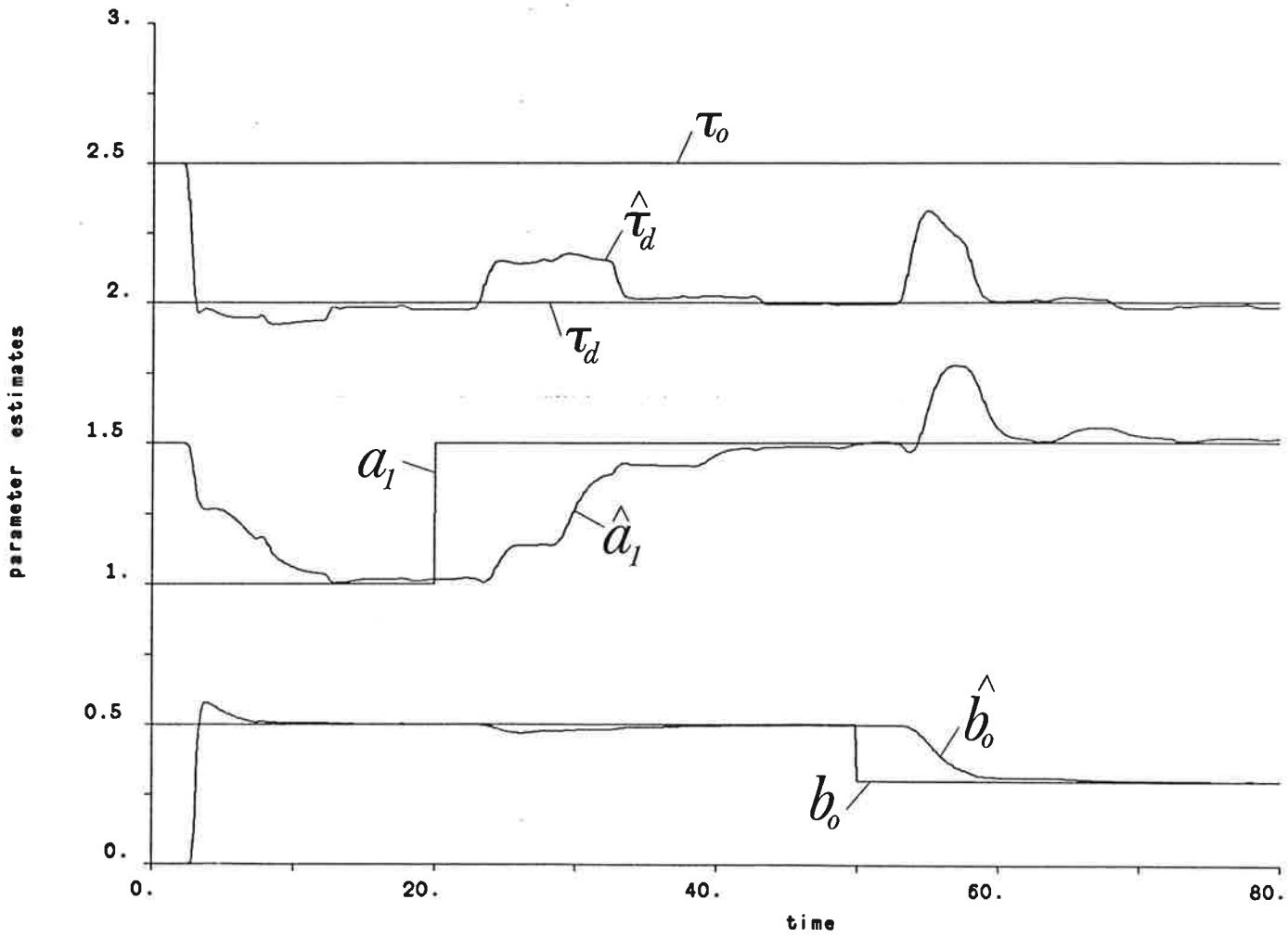

3.
2.5
2.
1.5
1.
0.5
0.
parameter estimates
τ_o
$\hat{\tau}_d$
τ_d
a_1
$\hat{a}_1$
$\hat{b}_o$
b_o
0.
20.
40.
60.
80.
time

Figure 2

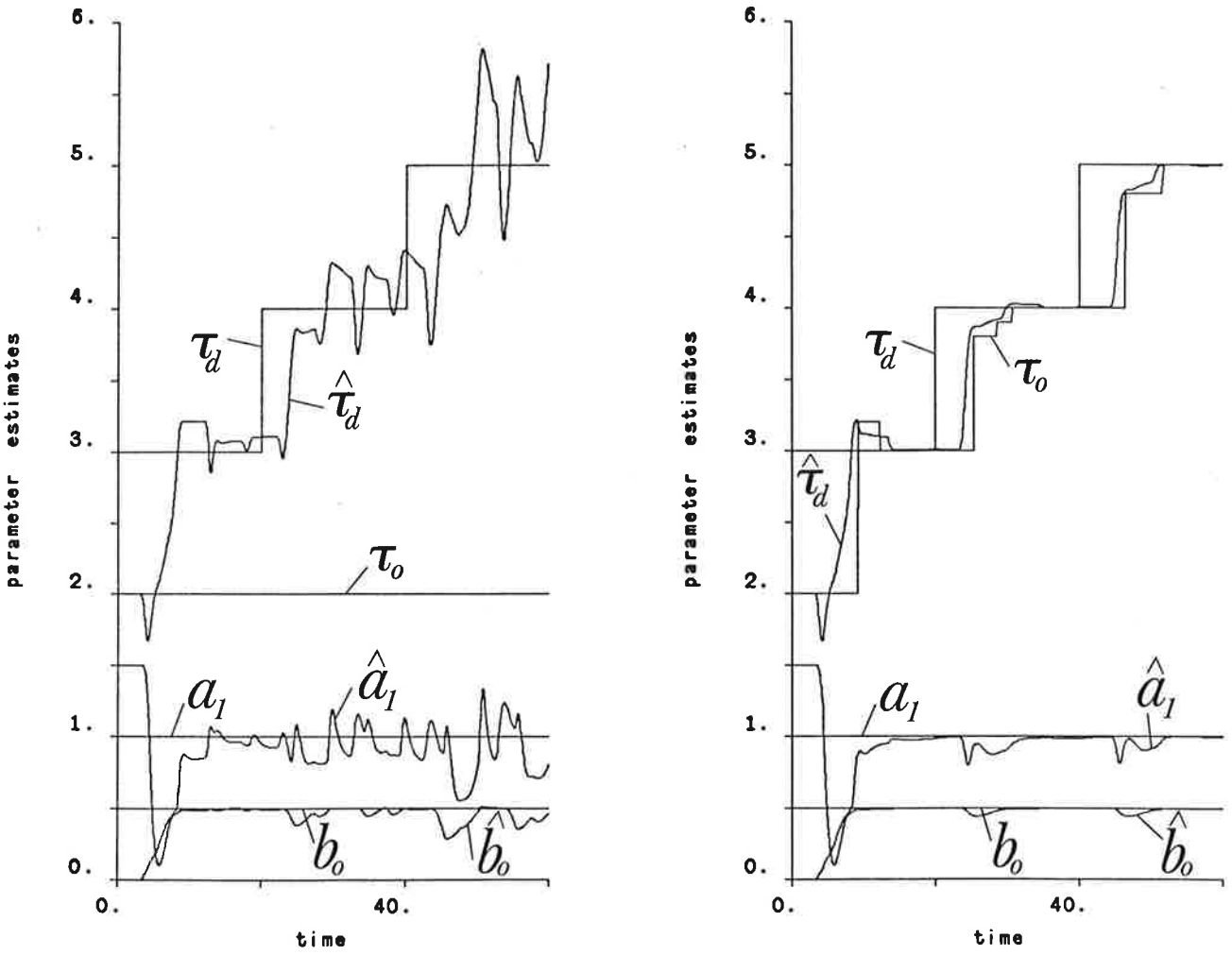

6.
5.
4.
3.
2.
1.
0.
parameter estimates
τ_d
$\hat{\tau}_d$
τ_o
a_1
$\hat{a}_1$
b_o
$\hat{b}_o$
0.
40.
time
(a)
τ_d
τ_o
$\hat{\tau}_d$
a_1
$\hat{a}_1$
b_o
$\hat{b}_o$
(b)

Figure 3

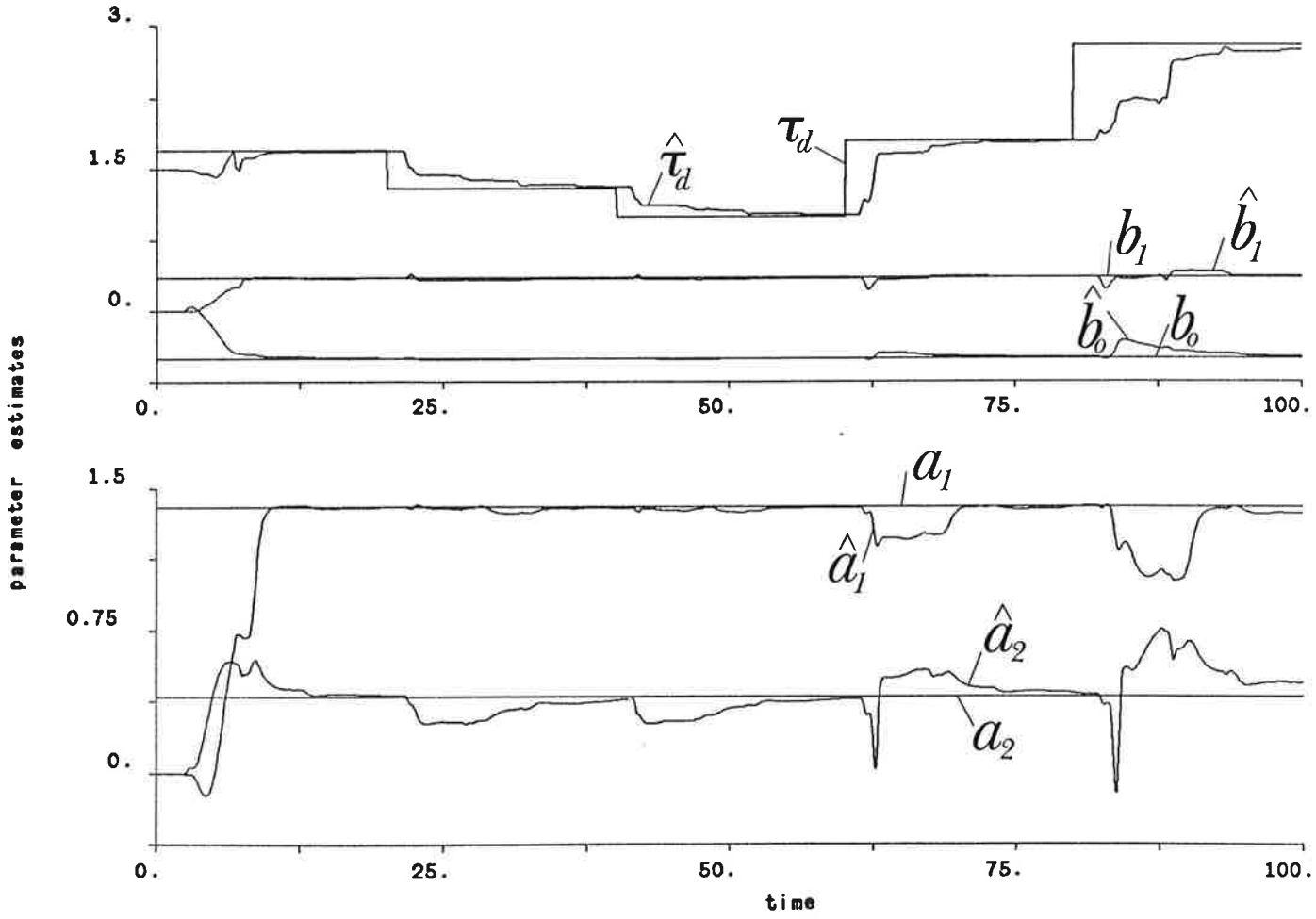

3.
1.5
0.
$\hat{\tau}_d$
τ_d
b_1
$\hat{b}_1$
$\hat{b}_o$
b_o
0.
25.
50.
75.
100.
1.5
0.75
0.
a_1
$\hat{a}_1$
$\hat{a}_2$
a_2
parameter estimates
time

Figure 4

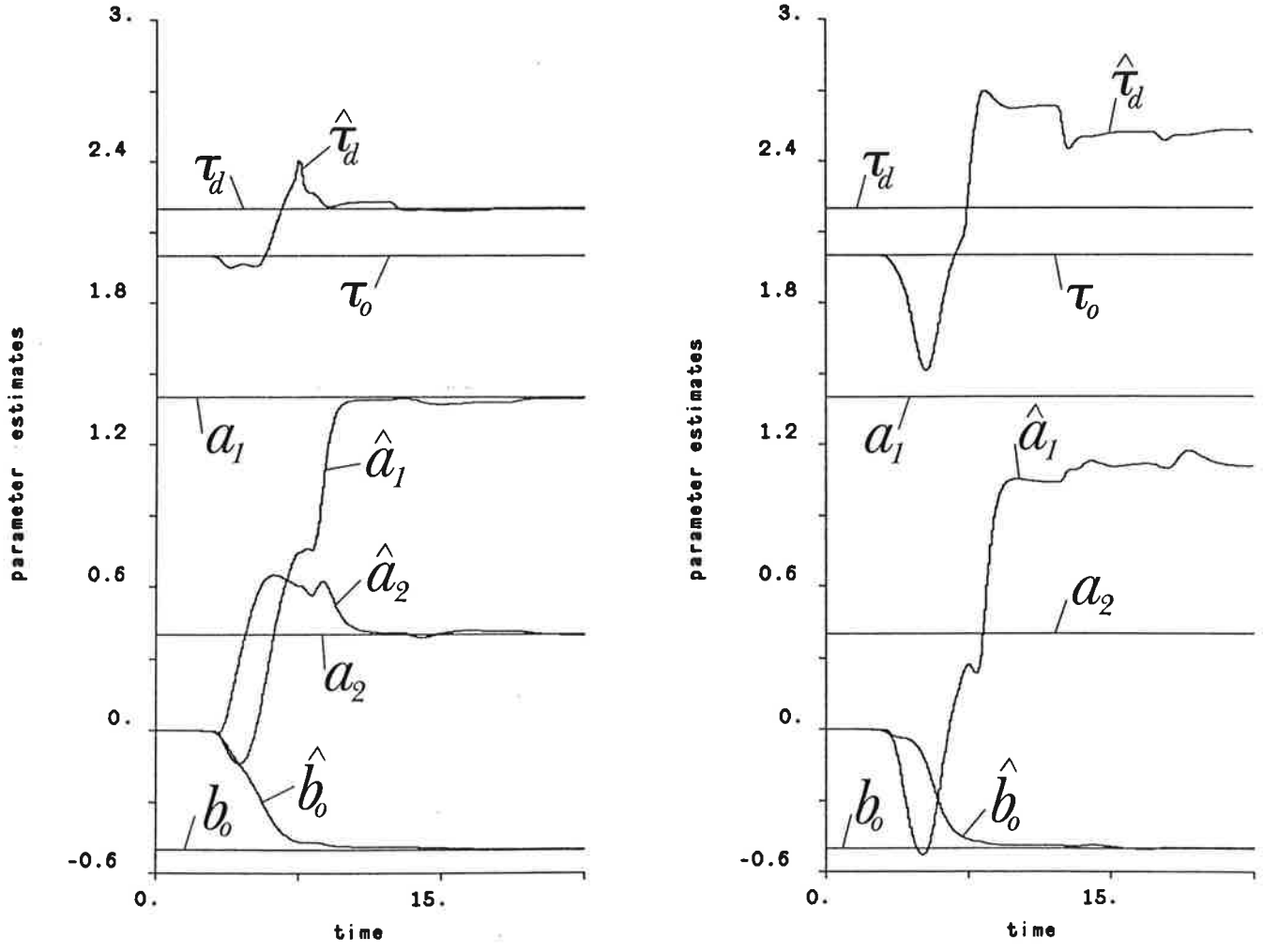
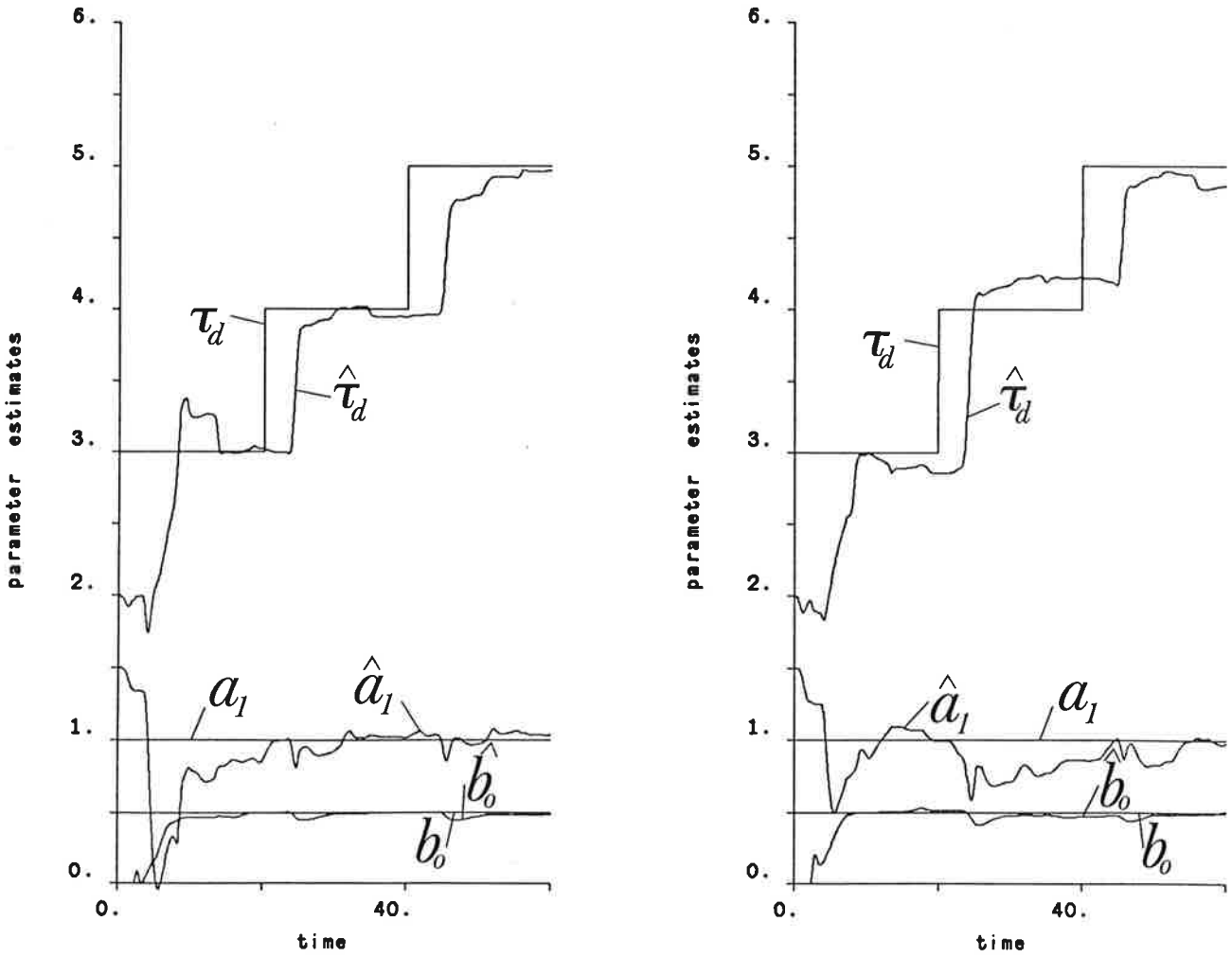

3.
2.4
1.8
1.2
0.6
0.
-0.6
parameter estimates
τ_d
$\hat{\tau}_d$
τ_o
a_1
$\hat{a}_1$
$\hat{a}_2$
a_2
b_o
$\hat{b}_o$
0.
15.
time
(a)
(b)

Figure 5


parameter estimates
6.
5.
4.
3.
2.
1.
0.
0.
40.
time
τ_d
$\hat{\tau}_d$
a_1
$\hat{a}_1$
b_o
$\hat{b}_o$

(a) (b)

Figure 6

SIMNON LISTING FOR FIRST-ORDER PROCESS IDENTIFICATION

```
CONTINUOUS SYSTEM PROC
"first order process with noise
"File called Proc

Input u nse
Output y

State x1
Der  dx1
time t

a=if t<ta then a1 else a2        "select time-constant
b=if t<tb then b1 else b2        "select gain

"states
dx1= (-x1 + b*u + nse*nsetyp)/a

"Outputs
y=x1+nse*(1-nsetyp)


"Parameters
a1:1            "time constant before ta
a2:1            "time constant after ta
ta:20           "time at which a changes
b1:.5           "gain before tb
b2:.5           "gain after tb
tb:50           "time at which b changes
nsetyp:0        "0 = measurement noise,  1 = process noise

End
```

```
DISCRETE SYSTEM REF
" reference generator, raw undelayed 'u'
"file called genref

Input nse delta
Output yr1
Time t
Tsamp ts

s=sign(mod(t,per)-per/2)              "periodic square wave with noise
yr1=if s<0 then amp+nse else amp2+nse

ts=t+delta

amp:.2      "upper value of signal u(t)
amp2:0      "lower value of signal u(t)
per:10      "time-period of signal u(t)
End
```

```
DISCRETE SYSTEM DTU
"delay signal u by amount d to generate delayed signal ud
"file called dtu
"systems dtu0 and dtu1 are identical except for first line
"delta is the smallest unit of delay.

Input delta u d
Output ud
State n1 n2 n3 n4 n5 n6 n7
State n8 n9 n10 n11 n12 n13 n14
State n15 n16 n17 n18 n19 n20 n21
State n22 n23 n24 n25 n26 n27 n28
State n29 n30 n31 n32 n33 n34 n35
State n36 n37 n38 n39 n40 n41 n42
State n43 n44 n45 n46 n47 n48 n49 n50
New nn1 nn2 nn3 nn4 nn5 nn6 nn7
New nn8 nn9 nn10 nn11 nn12 nn13 nn14
New nn15 nn16 nn17 nn18 nn19 nn20 nn21
New nn22 nn23 nn24 nn25 nn26 nn27 nn28
New nn29 nn30 nn31 nn32 nn33 nn34 nn35
New nn36 nn37 nn38 nn39 nn40 nn41 nn42
New nn43 nn44 nn45 nn46 nn47 nn48 nn49 nn50
Time t
Tsamp ts


d0=delta/2.
d1=1*delta+d0
d2=2*delta+d0
d3=3*delta+d0
d4=4*delta+d0
d5=5*delta+d0
d6=6*delta+d0
d7=7*delta+d0
d8=8*delta+d0
d9=9*delta+d0
d10=10*delta+d0
d11=11*delta+d0
d12=12*delta+d0
d13=13*delta+d0
d14=14*delta+d0
d15=15*delta+d0
d16=16*delta+d0
d17=17*delta+d0
d18=18*delta+d0
d19=19*delta+d0
d20=20*delta+d0
d21=21*delta+d0
d22=22*delta+d0
d23=23*delta+d0
d24=24*delta+d0
d25=25*delta+d0
d26=26*delta+d0
d27=27*delta+d0
d28=28*delta+d0
d29=29*delta+d0
d30=30*delta+d0
d31=31*delta+d0
d32=32*delta+d0
d33=33*delta+d0
d34=34*delta+d0
d35=35*delta+d0
d36=36*delta+d0
d37=37*delta+d0
d38=38*delta+d0
d39=39*delta+d0
d40=40*delta+d0
d41=41*delta+d0
d42=42*delta+d0
d43=43*delta+d0
d44=44*delta+d0
d45=45*delta+d0
d46=46*delta+d0
d47=47*delta+d0
d48=48*delta+d0
d49=49*delta+d0
d50=50*delta+d0
```

```
r0= if d<d0 then u else 0.
r1= if d<d1 and d>d0 then n1 else 0.
r2= if d<d2 and d>d1 then n2 else 0.
r3= if d<d3 and d>d2 then n3 else 0.
r4= if d<d4 and d>d3 then n4 else 0.
r5= if d<d5 and d>d4 then n5 else 0.
r6= if d<d6 and d>d5 then n6 else 0.
r7= if d<d7 and d>d6 then n7 else 0.
r8= if d<d8 and d>d7 then n8 else 0.
r9= if d<d9 and d>d8 then n9 else 0.
r10= if d<d10 and d>d9 then n10 else 0.
r11= if d<d11 and d>d10 then n11 else 0.
r12= if d<d12 and d>d11 then n12 else 0.
r13= if d<d13 and d>d12 then n13 else 0.
r14= if d<d14 and d>d13 then n14 else 0.
r15= if d<d15 and d>d14 then n15 else 0.
r16= if d<d16 and d>d15 then n16 else 0.
r17= if d<d17 and d>d16 then n17 else 0.
r18= if d<d18 and d>d17 then n18 else 0.
r19= if d<d19 and d>d18 then n19 else 0.
r20= if d<d20 and d>d19 then n20 else 0.
r21= if d<d21 and d>d20 then n21 else 0.
r22= if d<d22 and d>d21 then n22 else 0.
r23= if d<d23 and d>d22 then n23 else 0.
r24= if d<d24 and d>d23 then n24 else 0.
r25= if d<d25 and d>d24 then n25 else 0.
r26= if d<d26 and d>d25 then n26 else 0.
r27= if d<d27 and d>d26 then n27 else 0.
r28= if d<d28 and d>d27 then n28 else 0.
r29= if d<d29 and d>d28 then n29 else 0.
r30= if d<d30 and d>d29 then n30 else 0.
r31= if d<d31 and d>d30 then n31 else 0.
r32= if d<d32 and d>d31 then n32 else 0.
r33= if d<d33 and d>d32 then n33 else 0.
r34= if d<d34 and d>d33 then n34 else 0.
r35= if d<d35 and d>d34 then n35 else 0.
r36= if d<d36 and d>d35 then n36 else 0.
r37= if d<d37 and d>d36 then n37 else 0.
r38= if d<d38 and d>d37 then n38 else 0.
r39= if d<d39 and d>d38 then n39 else 0.
r40= if d<d40 and d>d39 then n40 else 0.
r41= if d<d41 and d>d40 then n41 else 0.
r42= if d<d42 and d>d41 then n42 else 0.
r43= if d<d43 and d>d42 then n43 else 0.
r44= if d<d44 and d>d43 then n44 else 0.
r45= if d<d45 and d>d44 then n45 else 0.
r46= if d<d46 and d>d45 then n46 else 0.
r47= if d<d47 and d>d46 then n47 else 0.
r48= if d<d48 and d>d47 then n48 else 0.
r49= if d<d49 and d>d48 then n49 else 0.
r50= if d<d50 and d>d49 then n50 else 0.

ud1=r0+r1+r2+r3+r4+r5+r6+r7+r8+r9+r10+r11+r12+r13+r14+r15+r16+r17
ud2=ud1+r18+r19+r20+r21+r22+r23+r24+r25+r26+r27+r28+r29+r30+r31+r32+r33
ud =ud2+r34+r35+r36+r37+r38+r39+r40+r41+r42+r43+r44+r45+r46+r47+r48+r49+r50

nn50=n49
nn49=n48
nn48=n47
nn47=n46
nn46=n45
nn45=n44
nn44=n43
nn43=n42
nn42=n41
nn41=n40
nn40=n39
nn39=n38
nn38=n37
nn37=n36
nn36=n35
nn35=n34
nn34=n33
nn33=n32
nn32=n31
nn31=n30
```

```
nn30=n29
nn29=n28
nn28=n27
nn27=n26
nn26=n25
nn25=n24
nn24=n23
nn23=n22
nn22=n21
nn21=n20
nn20=n19
nn19=n18
nn18=n17
nn17=n16
nn16=n15
nn15=n14
nn14=n13
nn13=n12
nn12=n11
nn11=n10
nn10=n9
nn9=n8
nn8=n7
nn7=n6
nn6=n5
nn5=n4
nn4=n3
nn3=n2
nn2=n1
nn1=u

ts=t+delta

End
```

```
DISCRETE SYSTEM DGEN
"generate the value of process time delay
"file called dgen

Input delta
Output d
Time t
Tsamp ts

dd=if t<t1 then d1 else if t<t2 then d2 else if t<t3 then d3 else 0
d=if t<t3 then dd else if t<t4 then d4 else d5

ts=t+delta

d1:1.7
d2:3.0
d3:2.2
d4:3.8
d5:3.8
t1:20
t2:40
t3:60
t4:80

end
```

```
DISCRETE SYSTEM RLS

"Sequential nonlinear RLS estimation
"ref. Goodwin and Sin pp. 310
"File called RLS
"q is the value of TAU-zero by which the filtered u's are to be delayed
"ud is the value assigned to TAU-zero by the estimation scheme

Input u0 u1 y0 y1 y2
Output delta q
State  th1 th2 th3
State p11 p12 p22
State p13 p23 p33
State ud
New   nth1 nth2 nth3
New np11 np12 np22
New np13 np23 np33
New nud
time t
Tsamp ts

"vector si(t-1)
f1=-y1-0.5*th3*y2
f2=u0-0.5*th3*u1
f3=-0.5*(th1*y2+y1+th2*u1)

"Estimation error
e=y0+th1*y1-th2*u0+0.5*th3*(th1*y2+y1+th2*u1)
e3=0  "to please macro ddelay

"Estimation gain
k1=p11*f1+p12*f2+p13*f3
k2=p12*f1+p22*f2+p23*f3
k3=p13*f1+p23*f2+p33*f3
den=lam+f1*k1+f2*k2+f3*k3

"Update estimates.
nth1=th1+k1*e/den
nth2=th2+k2*e/den
mth3=th3+k3*e/den

"Update covariance.
mp11=(p11-k1*k1/den)/lam
np12=p12-k1*k2/den
mp22=(p22-k2*k2/den)/lam
np13=p13-k1*k3/den
np23=p23-k2*k3/den
mp33=(p33-k3*k3/den)/lam

"reset covariance matrix
np11= if abs(e)>eset then pset else mp11
np22= if abs(e)>eset then pset else mp22
np33= if abs(e)>eset then pset else mp33

"logic for selecting q
q=ud
qth=q+mth3
ratio1=abs(k1*e/den/(th1+1e-20))
ratio2=abs(k2*e/den/(th2+1e-20))
ratio3=abs(k3*e/den/(th3+1e-20))
change=if ratio1<eps and ratio2<eps and ratio3<eps then 1 else 0   "convergence
update=if change>.9 and abs(mth3)>h then 1 else 0    "will q change by > h
"update ud to closest multiple of h in range 0,5
nud=if update>.9 then (min(5,max(0,h*int(qth/h)))) else ud
nth3=if update>.9 then (qth-nud) else mth3

"Update sampling time
ts=t+h
delta=h

"Parameters
h:.1                "sampling time
lam:.98             "forgetting factor
p11:1000            "covariance matrix elements
p22:1000
p33:1000
eset:1e34           "error threshold above which covariance is reset
```

```
pset:1000           "value to which covariance is reset
th1:1.5
th2:0
th3:0
ud:2.5
eps:0.0             "relative absolute change for convergence test

End
```

```
CONTINUOUS SYSTEM FILTER
"File called Filter
"Multifilter technique
"l th  order filters, l = 2 or 3

Input u y
Output u0 u1 y0 y1 y2

State x1 x2 x3
State z1 z2 z3
Der  dx1 dx2 dx3
Der  dz1 dz2 dz3

"states
dx1= -c*x1 + c*x2
dx2= -c*x2 + c*x3
dx3= -c*x3 + c*y

dz1= -c*z1 + c*z2
dz2= -c*z2 + c*z3
dz3= -c*z3 + c*u

"Outputs
y0=if l>2.9 then x1 else if l>1.9 then x2 else 0
y13=c*(x2-x1)
y12=c*(x3-x2)
y1=if l>2.9 then y13 else if l>1.9 then y12 else 0
y23=c^2*(x3-2*x2+x1)
y22=c^2*(y-2*x3+x2)
y2=if l>2.9 then y23 else if l>1.9 then y22 else 0


u0=if l>2.9 then z1 else if l>1.9 then z2 else 0
u13=c*(z2-z1)
u12=c*(z3-z2)
u1=if l>2.9 then u13 else if l>1.9 then u12 else 0


"Parameters
c:1.5          "Filter cut-off frequency [hz]
l:3            "order of filter

End
```

```
CONNECTING SYSTEM CDELAY
"connect filter, proc, rls, ref, noise1, dtu, dtu0, dtu1, dgen
"File called CDELAY

Time t
nse[ref]=e1[noise1]
nse[proc]=e2[noise1]
u[dtu]=yr1[ref]
u[proc]=ud[dtu]
u[filter]=yr1[ref]
y[filter]=y[proc]
u[dtu0]=u0[filter]
u0[rls]=ud[dtu0]
u[dtu1]=u1[filter]
u1[rls]=ud[dtu1]
y0[rls]=y0[filter]
y1[rls]=y1[filter]
y2[rls]=y2[filter]
delta[ref]=delta[rls]
delta[dgen]=delta[rls]
delta[dtu]=delta[rls]
delta[dtu0]=delta[rls]
delta[dtu1]=delta[rls]
d[dtu]=d[dgen]
d[dtu0]=q[rls]
d[dtu1]=q[rls]

end
```

```
MACRO SDELAY

"compilation of system and generation of noise
"File called  Sdelay

LET n.noise1=2
,nodd.noise1=21853
SYST NOISE1 GENREF DGEN DTU PROC FILTER DTU0 DTU1 RLS CDELAY
Par dt:.1
Par stdev1:0
Par stdev2:0
Par rect:0
Par same:1

End
```

```
MACRO DDELAY


split 2 2
store e e3 th1 th2 qth q a b d[dtu] u[filter] y[proc] u0[rls]

simu 0 100 .005
ashow e e3
text 'e'
ashow a th1
text'a'
ashow b th2
text'b'
ashow d qth q
text'd'
disp th1 th2 qth p11 p22 p33

End
```

SIMNON LISTING FOR SECOND-ORDER PROCESS IDENTIFICATION

Systems REF, DGEN, DTU, DTU0, DTU1, (and DTU2) are identical to those
for the first-order process identification

```
CONTINUOUS SYSTEM PROC
"second order process with noise
"File called Proc

Input u nse
Output y

State x1 x2
Der  dx1 dx2
Time t

ca0=1./(a1*a2)
ca1=(a1+a2)*ca0
cb0=b0*ca0
cb1=b0*b1*ca0

"states
dx1=x2+cb1*u
dx2=-ca0*x1-ca1*x2+(cb0-ca1*cb1)*u+nse*nsetyp*ca0

"Outputs
y=x1+nse*(1-nsetyp)

"for plots
pth1=a1*a2
pth2=a1+a2
pth4=b0*b1
pth5=b0

"Parameters
a1:1
a2:.4
b0:-.5
b1:-.7
nsetyp:0                "0 = meas. noise,  1 = process noise

End
```

```
DISCRETE SYSTEM RLS

"Sequential nonlinear RLS
"identifies (th4s+th5)/(th1ss+th2s+1) times (first order Pade for th3=delay)
"ref. Goodwin and Sin pp. 310
"File called RLS
"q is TAU-zero sent to dtu0,1,..
"ud is generated value of TAU-zero

Input u0 u1 u2 y0 y1 y2 y3
Output delta q
State  th1 th2 th3 th4 th5
State p11 p12 p22
State p13 p23 p33
State p14 p24 p34 p44
State p15 p25 p35 p45 p55
State ud
New nth1 nth2 nth3 nth4 nth5
New np11 np12 np22
New np13 np23 np33
New np14 np24 np34 np44
New np15 np25 np35 np45 np55
New nud
Time t
Tsamp ts

"vector si(t-1)
f1=-.5*th3*y3-y2
f2=-.5*th3*y2-y1
f3=-.5*(th1*y3+th2*y2+y1+th4*u2+th5*u1)
f4=-.5*th3*u2+u1
f5=-.5*th3*u1+u0

"Estimation error
ey=y0+(th2+.5*th3)*y1+(th1+.5*th2*th3)*y2+.5*th1*th3*y3
e=ey-th5*u0-(th4-.5*th3*th5)*u1-(-.5*th3*th4)*u2
e3=0  "to please macro ddelay

"Estimation gain
k1=p11*f1+p12*f2+p13*f3+p14*f4+p15*f5
k2=p12*f1+p22*f2+p23*f3+p24*f4+p25*f5
k3=p13*f1+p23*f2+p33*f3+p34*f4+p35*f5
k4=p14*f1+p24*f2+p34*f3+p44*f4+p45*f5
k5=p15*f1+p25*f2+p35*f3+p45*f4+p55*f5
den=lam+f1*k1+f2*k2+f3*k3+f4*k4+f5*k5

"Update estimates.
nth1=th1+k1*e/den
nth2=th2+k2*e/den
mth3=th3+k3*e/den
nth4=th4+k4*e/den
nth5=th5+k5*e/den

"Update covariance.
mp11=(p11-k1*k1/den)/lam
np12=p12-k1*k2/den
mp22=(p22-k2*k2/den)/lam
np13=p13-k1*k3/den
np23=p23-k2*k3/den
mp33=(p33-k3*k3/den)/lam
np14=p14-k1*k4/den
np24=p24-k2*k4/den
np34=p34-k3*k4/den
mp44=(p44-k4*k4/den)/lam
np15=p15-k1*k5/den
np25=p25-k2*k5/den
np35=p35-k3*k5/den
np45=p45-k4*k5/den
mp55=(p55-k5*k5/den)/lam

"update covariance matrix
np11= if abs(e)>eset then pset else mp11
np22= if abs(e)>eset then pset else mp22
np33= if abs(e)>eset then pset else mp33
np44= if abs(e)>eset then pset else mp44
np55= if abs(e)>eset then pset else mp55
```

```
"logic for selecting q
q=ud
qth=q+mth3
ratio1=abs(k1*e/den/(th1+1e-20))
ratio2=abs(k2*e/den/(th2+1e-20))
ratio3=abs(k3*e/den/(th3+1e-20))
ratio4=abs(k4*e/den/(th4+1e-20))
ratio5=abs(k5*e/den/(th5+1e-20))
chang3=if ratio1<eps and ratio2<eps and ratio3<eps then 1 else 0
change=if chang3>.9 and ratio4<eps and ratio5<eps then 1 else 0    "convergence
update=if change>.9 and abs(mth3)>h then 1 else 0
nud=if update>.9 then (min(5,max(0,h*int(qth/h)))) else ud
nth3=if update>.9 then (qth-nud) else mth3

"Update sampling time
ts=t+h
delta=h

"Parameters
h:.1                "sampling time
lam:.98             "forgetting factor
p11:1000            "covariance elements
p22:1000
p33:1000
p44:1000
p55:1000
eset:1e34           "error threshold for covariance reset
pset:1000           "reset value of covariance elements
th1:0
th2:0
th3:0
th4:0
th5:0
ud:1.5
eps:0.01            "relative absolute change for convergence test

End
```

```
CONTINUOUS SYSTEM FILTER
"File called Filter
"Multifilter technique
"3rd,4th  order filters

Input u y
Output u0 u1 u2 y0 y1 y2 y3

State x1 x2 x3 x4
State z1 z2 z3 z4
Der  dx1 dx2 dx3 dx4
Der  dz1 dz2 dz3 dz4

"states
dx1= -c*x1 + c*x2
dx2= -c*x2 + c*x3
dx3= -c*x3 + c*x4
dx4= -c*x4 + c*y

dz1= -c*z1 + c*z2
dz2= -c*z2 + c*z3
dz3= -c*z3 + c*z4
dz4= -c*z4 + c*u

"Outputs
y0=if l>3.9 then x1 else x2
y1=if l>3.9 then c*(x2-x1) else c*(x3-x2)
y2=if l>3.9 then c^2*(x3-2*x2+x1) else c^2*(x4-2*x3+x2)
y3=if l>3.9 then c^3*(x4-3*x3+3*x2-x1) else c^3*(y-3*x4+3*x3-x2)

u0=if l>3.9 then z1 else z2
u1=if l>3.9 then c*(z2-z1) else c*(z3-z2)
u2=if l>3.9 then c^2*(z3-2*z2+z1) else c^2*(z4-2*z3+z2)


"Parameters
c:3           "Filter cut-off frequency [hz]
l:4           "order of filter

End




CONNECTING SYSTEM CDELAY
"connect filter, proc, rls, ref, dgen, dtu, dtu0, dtu1, dtu2, noise1
"File called CDELAY

Time t
nse[ref]=e1[noise1]
nse[proc]=e2[noise1]
u[dtu]=yr1[ref]
u[proc]=ud[dtu]
u[filter]=yr1[ref]
y[filter]=y[proc]
u[dtu0]=u0[filter]
u0[rls]=ud[dtu0]
u[dtu1]=u1[filter]
u1[rls]=ud[dtu1]
u[dtu2]=u2[filter]
u2[rls]=ud[dtu2]
y0[rls]=y0[filter]
y1[rls]=y1[filter]
y2[rls]=y2[filter]
y3[rls]=y3[filter]
delta[ref]=delta[rls]
delta[dgen]=delta[rls]
delta[dtu]=delta[rls]
delta[dtu0]=delta[rls]
delta[dtu1]=delta[rls]
delta[dtu2]=delta[rls]
d[dtu]=d[dgen]
d[dtu0]=q[rls]
d[dtu1]=q[rls]
d[dtu2]=q[rls]

end
```

```
MACRO DDELAY


split 2 2
store e th1 th2 qth q th4 th5 pth1 pth2 d[dtu] pth4 pth5

simu 0 100 .005
ashow e
text 'e'
ashow th1 pth1 th2 pth2
text'a1,a2'
ashow th4 pth4 th5 pth5
text'b0,b1'
ashow d qth q
text'd'
disp th1 th2 qth th4 th5 p11 p22 p33 p44 p55

End
```

```
MACRO SDELAY

"compilation of system and generation of noise
"File called  sdelay

LET n.noise1=2
,nodd.noise1=21853
SYST NOISE1 GENREF DGEN DTU PROC FILTER DTU0 DTU1 DTU2 RLS CDELAY
Par dt:.1
Par stdev1:0
Par stdev2:0
Par rect:0
Par same:1

End
```