Václav Peterka

Control of uncertain processes: applied theory and algorithms

Kybernetika, Vol. 22 (1986), No. Suppl, (1),3--72,73--101

Persistent URL: http://dml.cz/dmlcz/125576

Kybernetika

CONTROL OF UNCERTAIN PROCESSES: APPLIED THEORY AND ALGORITHMS

VÁCLAV PETERKA

ACADEMIA

PRAHA

Some advances in digital control of continuous linear stochastic processes with unknown parameters are reported. The theory is developed with emphasis on algorithmic and numerical aspects. Stochastic input-output models of ARMA form, contingently multivariate and with time delay, are used to represent the process to be controlled but, for numerical reasons in cases of fast sampling rates, also the theoretically equivalent Delta models are considered in parallel. PASCAL procedures, suitable for real time computation and microprocessor implementation, are given for the main resulting algorithms.

1. INTRODUCTION

Most processes met in practice are uncertain in the sense that it is not possible to determine what the future output of the process precisely will be. When modelling such process for control-design purposes two kinds of uncertainties are encountered. The first one is due to stochastic nature of real processes. This uncertainty cannot be removed but it can be described by suitable stochastic models and thus taken into account. The second kind of uncertainty arises when it is a priori not known which stochastic model in the set of possible models is the "true" one. This second kind of uncertainty can be removed, at least partially, by identification of the stochastic model on the basis of the input-output data observed during its operation. The following text deals with direct digital control of processes subject to the both above uncertainties.

The paper is a report on some recent advances in the research of industrial process control carried out at the Institute of Information Theory and Automation of the Czechoslovak Academy of Sciences. It can be considered as a selfcontained continuation of the survey of former results presented by the author's colleagues in [3]. (For references see end of each section.)

1.1. Motivation and layout of the paper

The LQG control theory appears to be a suitable basis for designing of control algorithms. However, when trying to apply practically the available theoretical results to industrial control problems difficulties of several kinds arise. The stochastic state space models in their general form, as considered in the standard theory, are often not available and cannot be identified solely from input-output data. The problem formulation often has to be nontrivially modified to meet the practical needs. Theoretical results (e.g. the matrix differential or difference equations of Riccati type) are not suitable for numerical calculation with limited precision of algebraic operations.

Most but not all of these difficulties can be removed by considering the input-output models of regression type as described in [3] and [7]. The regression models can be easily identified even when their parameters vary in time [4, 5], and all successful industrial applications we have been involved in are based on these models.

However, the choice of a suitable sampling rate still remains to be a problem. Simple self-tuning controllers reported in [1] have been designed for relatively long sampling period which must be chosen by the user and is important for proper functioning. Faster sampling gives, in general, better quality of control but then a higher order of the employed regression model is required in order to cover a sufficiently long past history of the process. This, of course, leads to higher computational burden both for process identification and control law synthesis. In case of very fast sampling rates, approaching the continuous control, also numerical difficulties can be expected.

In order to cope with fast sampling rates attempts have been made to approach the problem from the side of optimal continuous control. Unfortunately, the continuous stochastic control theory based on state space models leads to feasibility problems and seems to be rather an interesting academical topic than a useful tool for practicing engineers. Until now only asymptotically optimal synthesis based on polynomial algebra has been found practicable in real-time computation when limited to simple low order cases [6]. It also turns out that a proper discretization of the continuous solution, as required for digital implementation, is not a simple task as it might seem.

These unsuccessful, or only partially successful attempts led the author back to discrete-time standpoint but with different form of the input-output models. It has been found that the input-output relation, when expressed through difference operators rather than as weighted sum of neighboring samples, is less sensitive with respect to rounding errors, especially in case of fast sampling. Independently, the same observation has been made by Goodwin [2] who introduced the name "Delta models" for such representation of sampled continuous processes and also gives some analyses supporting this observation. In case of fast sampling Delta models can be considered as a suitable approximation of stochastic differential equations while for slower sampling rates they are just an alternative form of standard ARMA models with no approximation involved.

In the present paper the LQG control and estimation theory is revised with emphasis on algorithmic and numerical aspects. Both ARMA and Delta models are considered in parallel. Numerically robust (insensitive with respect to rounding errors) and efficient algorithms for real-time identification and optimum control synthesis, suitable for self-tuning control, are developed. The presentation is arranged in the following manner.

In the rest of this Introduction the considered control loop with continuous stochastic process and digital controller is described, practical objectives are stated, and the main notational conventions are introduced.

Section 2 is devoted to basic tools which will be repeatedly applied throughout the paper. They are: quantitative description of uncertainties by probability distributions in Bayesian interpretation, the elementary algorithm of "dyadic reduction", and the application of the latter to decomposition and minimization of quadratic forms and to operations with multivariate normal probability distribution.

In Section 3 linear stochastic input-output models of positional and incremental types and of ARMA and Delta forms are introduced and transformed into a uniform canonical state-space form suitable for numerical solutions. Simple procedures for digital simulation of continuous stochastic processes and for tranasformation of models between ARMA and Delta forms are included.

The special state form of the stochastic input-output models makes it possible to derive the filter for real time estimation of the model state which is simpler and numerically more robust than the standard Kalman filter. This is shown in Section 4.

Optimum control synthesis is the topic of Section 5. After general discussion of the problem suitable quadratic criteria are introduced and minimized for the above stochastic models with known parameters. The special canonical form of the state-space model is exploited to obtain effective numerical algorithms yielding the optimum control law operating on the estimate of the state (more precisely, on the expected value of the model state conditioned on the known past data).

The case of unknown parameters is considered in Section 6, where the problem of simultaneous parameter estimation and state estimation (and prediction in case of time delay) is solved. It is shown that, for given or suitably chosen c-parameters of the ARMA model or its Delta form, the problem can be solved exactly within normal distributions. Algorithms are derived which update the statistic which is sufficient both for parameter and state estimation and defines the corresponding joint probability distribution.

In Section 7 the problem of dual control is discussed and a well feasible numerical solution of the LQG self-tuning control is suggested.

To support proper understanding of what the algorithms really do and what is the meaning of each number entering the numerical computation each section starts with the conceptual solution of the given problem in terms of conditional probability distributions. Then this general solution is specialized for the considered linear stochastic models, the corresponding algorithms are derived, and finally if appropriate, the PASCAL procedures for immediate use are listed.

If it will aid to simplify the exposition or to make the main ideas more plain the single-input single-output case will be considered first. However, care will be exercised so that the extension to multivariate case outlined afterwards may be straightforward.

1.2. Notational conventions

The control loop considered in the sequel is schematically sketched in Fig. 1. The process P is controlled by a digital controller C generating a sequence of numbers $u(t)$; $t = 1, 2, 3, \ldots$, vectors in multivariate case, which govern the actuator(s) used to manipulate the process. The controlled output y_c of the process is assumed to be measured and sampled with the same period as the inputs $u(t)$ are generated. The task of the controller is to generate the inputs in such a way that the controlled output within a given control horizon be as close as possible to its prescribed values w,

the command signal. To perform this task the controller can make use of an auxiliary output y_a which is not subject to control but carries an additional piece of information about the state of the process. If available for measurement, also the sampled external disturbance v can be introduced as a feedforward to improve the performance of the controller.

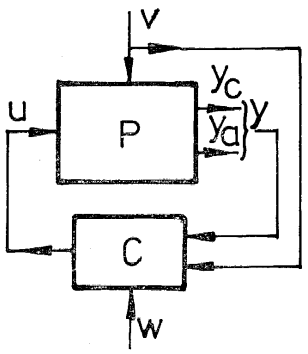

Fig. 1. Control loop with continuous stochastic process P and digital controller C.

The total process output will be denoted as $y = \{y_c, y_a\}$. All digital signals in the control loop can be, in general, multivariate. Of course, the dimension of the controlled output y_c is the same as of the command signal w.

Time indexing. The signal samples to which the same discrete time t is assigned are related to the sampling period in which the input $u(t)$ is generated. However, within one sampling period it is assumed that $u(t)$ is determined first and then the samples $v(t)$ and $y(t)$ follow. This means that when the input $u(t)$ is being decided only the signal samples observed on the process up to and including the sampling period $(t-1)$ are available, the samples $v(t)$ and $y(t)$ are not known yet.

According to the way how the command signal is made available for the controller the following operating modes will be considered.

Regulation. The command signal is a given constant, a fixed setpoint, which can be set to zero if the controlled output is measured relatively to this fixed reference value.

Program control. The command signal is preprogrammed in advance for the entire control horizon and this prior information is available for the controller. This means that the controller when generating $u(t)$ can operate also on $w(t+k)$, $k > 0$.

Positional servo. When $u(t)$ is being generated the future course of the command signal $w(t+k)$, $k > 0$, is uncertain. A suitable model for this case will be introduced in Section 5.

Some abbreviations

p.f. ... probability function
p.d.f. ... probability density function

c.p.d.f. ... conditional p.d.f.
LT-matrix ... lower triangular matrix
UT-matrix ... upper triangular matrix
monic LT-matrix ... LT-matrix with 1's on the main diagonal
LD-factorization ... factorization of a nonnegative definite matrix $M = LDL'$ where L is a monic LT-matrix and D is a diagonal matrix with nonnegative diagonal elements

Main symbols

(3) ... equation (3) in the current section
(5.3) ... equation (3) in Section 5
M' ... transposed matrix M
$\mathrm{tr}\,(M)$ ... trace of a square matrix M, the sum of diagonal elements
$\det(M)$ ... determinant of a square matrix M
$|D; Mx|$ ... nonnegative definite quadratic form $x'M'DMx$ where M is a rectangular matrix, D is a diagonal matrix with nonnegative elements, and x is a column vector
∂x ... dimension of the vector x
t ... discrete time, time index of a sampling period; the observation of the process (gathering of data, not necessarily control) starts at the sampling period indexed by $t = 1$
τ ... continuous time
$y(t)$ ... sample of a signal (here output) related to the sampling period with time index t; in general a set, in algebraic expressions assumed to be ordered into a column vector
$\Delta y(t)$ ... backwards difference, $\Delta y(t) = y(t) - y(t-1)$
$y(j \ldots k)$ ... set of all signal samples in the given time span, $j \leqq t \leqq k$; for $j > k$ the set is empty

Model-related variables and parameters

$s(t)$ ... state of the process model in canonical form
$x(t)$ ... model state extended by the process output, $x(t) = \{y(t), s(t)\}$
a_i, b_i, c_i, d_i ... coefficients of the input-output stochastic model, matrices in multivariate case
A, b, c, d ... matrix coefficients of the canonical state model
n ... order of the input-output model
θ ... set of uncertain model parameters

Probability distributions

$\mathsf{p}(a \mid b)$ ... p.d.f. or p.f., according to the type of the argument a, conditioned on b
$\mathsf{p}(y(t) \mid t-1; u(t))$ ... abbreviation for
$\mathsf{p}(y(t) \mid u(1 \ldots t), y(1 \ldots t-1), v(1 \ldots t-1))$;

$t-1$ in the condition alone means conditioning on all data observed on the given process up to and including the sampling period $t-1$, e.g.

$\mathsf{p}(\cdot \mid t-1)$ ... abbreviation for

$\mathsf{p}(\cdot \mid u(1 \cdot\cdot t-1), y(1 \cdot\cdot t-1), v(1 \cdot\cdot t-1))$

$\mathsf{E}[a]$ ... expected (mean) value of a random variable a

$\hat{y}(t \mid t-1; u(t))$ ... abbreviation for the expected value

$\mathsf{E}[y(t) \mid u(1 \cdot\cdot t), y(1 \cdot\cdot t-1), v(1 \cdot\cdot t-1)]$;

similarly for $\hat{s}(t \mid t-1)$ and $\hat{x}(t \mid t-1)$

$\mathrm{Cov}[a, b]$ ... covariance $\mathsf{E}[(a - \mathsf{E}[a])(b - \mathsf{E}[b])']$

$\mathrm{Var}[a]$ ... $\mathrm{Cov}[a, a]$

$R_x(t \mid t-1; u(t))$ ... covariance matrix $\mathrm{Var}[x(t) \mid t-1; u(t)]$

REFERENCES

[1] J. Böhm, A. Halousková, M. Kárný and V. Peterka: Simple LQ self-tuning controllers. 9th World Congress of IFAC, 1984, Budapest, Hungary, Preprints Vol. VII, 171—176.

[2] G. G. Goodwin: Some observations on robust estimation and control. 7th IFAC/IFORS Symposium on Identification and System Parameter Estimation, 1985, York, UK, Preprints Vol. 1, 851—859.

[3] M. Kárný, A. Halousková, J. Böhm, R. Kulhavý and P. Nedoma: Design of linear quadratic adaptive control: theory and algorithms for practice. Supplement to the journal Kybernetika, Vol. *21* (1985), No. 3, 4, 5, 6.

[4] R. Kulhavý and M. Kárný: Tracking of slowly varying parameters by directional forgetting. 9th World Congress of IFAC, 1984, Budapest, Hungary, Preprints Vol. X, 78—83.

[5] R. Kulhavý: Restricted exponential forgetting in real-time identification. 7th IFAC/IFORS Symposium on Identification and System Parameter Estimation, 1985, York, UK, Preprints Vol. 2, 1143—1148.

[6] I. Nagy and V. Peterka: A hybrid LQ self-tuning controller. 7th IFAC/IFORS Symposium on Identification and System Parameter Estimation, 1985, York, UK, Preprints Vol. 1, 1025—1030.

[7] V. Peterka: Predictor-based self-tuning control. Automatica *11* (1981), 39—50.

2. BASIC TOOLS

In this section the tools are introduced which will be repeatedly applied in the following sections.

First the Bayesian methodology for dealing with uncertainties and for solving statistical problems will be expounded. This will be the main tool in conceptual solutions of particular problems each section will start with.

The elementary algorithm of dyadic reduction, derived and coded in the second paragraph, forms the basis for numerical solution of optimum control synthesis as well as of state and parameter estimation. To make its later applications straightforward it is shown in the third and fourth paragraphs how the dyadic reduction

can be used to decomposition and minimization of quadratic forms and to operations with normal multivariate probability distributions.

The last of basic operations introduced in this section is the linear composition of normal probability distributions.

2.1. Uncertainty and probability, Bayesian methodology

Only the main features of the Bayesian philosophy can be briefly surveyed here. A more detailed explanation with applications to system identification is given in [10] where also further references can be found.

The Bayesian methodology as applied here rests on the fact that uncertainty has the probability structure. The meaning of this statement is that the mathematical discipline called probability theory, in which the concept of probability is defined axiomatically without any relation to reality, can be employed to operate with subjective probability distributions which are used to describe quantitatively the uncertain relation between a rationally and consistently reasoning person (e.g. a control system designer) and the external world (the process to be controlled). This can be proved on the basis of a few simple and sound principles.

In Bayesian view random means uncertain. Not only data but also uncertain constants like model parameters are random. Similarly, a hypothesis which is not known to be true is a random event and probability can be assigned to it.

Bayesian standpoint is that a random variable can take on just one true value. Act of observation changes the status of the quantity from a random variable to a number, a random event is changed, when observed, into a fact,

It should be stressed that the Bayesian interpretation of probability as a rational measure of belief is in no contradiction with initutive conception of probability as the limit of relative frequences the stationarity of which may appear to an outer observer, with a given observation ability, as an objective property of the external world. On the contrary, the idea of such limits can be very helpful when constructing stochastic models (see Section 3). Bayesian statistics can serve as a means for finding out what these, so-called 'objective', probabilities are. However, its applicability, as exploited in this paper, is much wider.

Basic operations on probability distributions

It can be said that the following two relations determine the structure of the Bayesian system of consistent reasoning.

$$\mathsf{p}(b \mid c) = \int \mathsf{p}(a, b \mid c)\, \mathrm{d}a \tag{1}$$

$$\mathsf{p}(a, b \mid c) = \mathsf{p}(a \mid b, c)\, \mathsf{p}(b \mid c) \tag{2}$$

The first relation determines the marginal distribution for b from the joint probability distribution of a and b where a is an auxiliary uncertain quantity which is to be

eliminated. The integration in (1) is taken over the entire set of possible values of a. If a is of discrete type the integration in (1) has to be replaced by the appropriate summation. The relation (2) gives the rule how a joint probability distribution can be decomposed. When read from right to left it shows how a joint probability distribution can be constructed if required. When rewritten in the following way

$$\mathrm{p}(a \mid b, c) = \frac{\mathrm{p}(a, b \mid c)}{\mathrm{p}(b \mid c)} = \frac{\mathrm{p}(a, b \mid c)}{\int \mathrm{p}(a, b \mid c)\, \mathrm{d}a} \tag{3}$$

it describes the operation of conditioning. If in (3) the decomposition (2) with interchanged roles of a and b is applied the famous Bayes formula is obtained

$$\mathrm{p}(a \mid b, c) = \frac{\mathrm{p}(b \mid a, c)\, \mathrm{p}(a \mid c)}{\int \mathrm{p}(b \mid a, c)\, \mathrm{p}(a \mid c)\, \mathrm{d}a} \tag{4}$$

which can be understood as a rule how a prior probability distribution $\mathrm{p}(a \mid c)$ can be corrected by incorporating a newly observed quantity b. Note that $\mathrm{p}(b \mid a, c)$ must be given in order to be able to perform this task. Note also that the integral in the denominator is just a normalizing factor which does not depend on a.

It is true that all practicable probability distributions are conditioned, at least on the prior information on the basis of which they have been constructed. However, it is a good usage not to state explicitly and repeatedly the conditions which are permanent during the solution of a given problem.

Actually, all conceptual solutions of particular problems we shall deal with in the following sections are nothing else than a systematic applications of the relations (1) and (2). However, to be able to apply them consistently in a closed control loop with an adaptive controller it is necessary to define conditions under which the controller operates.

Natural conditions of control

Suppose that q is a quantity on the process which is uncertain but its c.p.d.f. $\mathrm{p}(q \mid t-1)$ conditioned on all data given a priori and observed on the process up to and including the sampling period $(t-1)$ is available. The question is how this probability distribution has to be modified when a new input $u(t)$ has been generated and incorporated into the condition.

In adaptive control the controller when generating the new input $u(t)$ can make use only of that information about the uncertain quantity q which is given a priori and which is contained in the observed data. If this information is already reflected in the condition of the probability distribution for q then the mere generation of $u(t)$ cannot bring any new piece of information and it must hold

$$\mathrm{p}(q \mid t-1; u(t)) = \mathrm{p}(q \mid t-1) \tag{5}$$

The relation (5) cannot be derived mathematically, it must be introduced exogeneously as an assumption defining the "natural conditions of control" [10].

To throw more light on these conditions let us consider the joint probability distribution $\mathrm{p}(q, u(t) \mid t-1)$ and let us decompose it in the following two ways.

$$\mathrm{p}(q \mid t-1; u(t))\, \mathrm{p}(u(t) \mid t-1) = \mathrm{p}(u(t) \mid t-1; q)\, \mathrm{p}(q \mid t-1)$$

From this relation it is clear that if (5) holds then also the equality

$$\mathrm{p}(u(t) \mid t-1; q) = \mathrm{p}(u(t) \mid t-1) \tag{6}$$

must hold, and vice versa. The equality (6) reflects the fact that a realizable control law cannot operate on the quantity q which is not known.

Transformation of random variables

When operating with probabilistic models it is often suitable to transform random variables in order to obtain the probability distribution of interest. For our purposes it is sufficient to consider only linear regular transformations.

Suppose that the c.p.d.f. for a multivariate random variable e is given as a function f of e and of the condition c.

$$\mathrm{p}(e \mid c) = \mathrm{f}(e, c)$$

The question is how the c.p.d.f. of a random variable x can be determined if the following relation holds

$$e = T(c)\, x + q(c) \tag{7}$$

where $T(c)$ is a nonsingular square matrix possibly dependent on the condition c, and $q(c)$ is a vector. The answer follows from the theory of multiple integrals and is

$$\mathrm{p}(x \mid c) = \mathrm{f}(e, c)\, |\det (T(c))|$$

where e is to be substituted from (7).

2.2. Algorithm of dyadic reduction

The elementary algorithm which will be introduced now is due to the author's colleague K. Šmuk. It makes it possible to construct elegant and numerically save procedures for decompositions and minimizations of nonnegative definite quadratic forms and therefore has been chosen as the algorithmic basis for optimum control synthesis as well as for numerical operations with normal probability distributions in real-time estimation.

Consider a symmetric nonnegative definite matrix M of rank at most 2 expressed as a weighted sum of two symmetric dyads

$$M = f' D_f f + r' D_r r \tag{8}$$

where f and r are row-vectors

$$f = [1, f_1, f_2, \ldots, f_n]$$

$$r = [r_0, r_1, r_2, \ldots, r_n]$$

while D_f and D_r are nonnegative scalar weights. Note that the element f_0 is 1. Of course, such a representation of the matrix M is not unique and can be modified.

$$M = f' D_f f + r' D_r r = \tilde{f}' \tilde{D}_f \tilde{f} + \tilde{r}' \tilde{D}_r \tilde{r} \tag{9}$$

Let us consider the modification by which the element $\tilde{f}_0$ remains to be equal to 1, but the element in the same position in $\tilde{r}$ is zeroed.

$$\tilde{f}_0 = 1 \; ; \quad \tilde{r}_0 = 0$$

It will be proved that the following simple algorithm performs this modification and reduces the dimension of the nonzero part of the row-vector r by one.

Algorithm of dyadic reduction

$$\tilde{D}_f = D_f + D_r r_0^2 \tag{10}$$

$$\tilde{D}_r = (D_f / \tilde{D}_f)\, D_r \tag{11}$$

$$k_r = (D_r / \tilde{D}_f)\, r_0 \tag{12}$$

$$j = 1, 2, \ldots, n: \tilde{r}_j = r_j - r_0 f_j \tag{13}$$

$$\tilde{f}_j = f_j + k_r \tilde{r}_j \tag{14}$$

It is easily seen that in all cases when the reduction has sense $\tilde{D}_f > 0$ and the divisions in (11) and (12) can be performed. For $\tilde{D}_f = 0$ the reduction can be simply skipped as in such a case, besides $D_f = 0$, either r_0 is already zero or the entire dyad $r' D_r r$ has a zero weight D_r and can be omitted.

Proof. From (9) it is seen that the following relation must hold for any i and j.

$$f_i D_f f_j + r_i D_r r_j = \tilde{f}_i \tilde{D}_f \tilde{f}_j + \tilde{r}_i \tilde{D}_r \tilde{r}_j \tag{15}$$

By setting $i = j = 0$ and $\tilde{r}_0 = 0$ the formula (10) is obtained. For $i = 0$ and $j > 0$ it follows that

$$\tilde{f}_j = (D_f f_j + r_0 D_r r_j) / \tilde{D}_f \tag{16}$$

After substitution for $\tilde{f}_i$ and for $\tilde{f}_j$ from (16) the relation (15) for $i > 0$ and $j > 0$ can be rearranged in the following way.

$$\begin{aligned} \tilde{r}_i \tilde{D}_r \tilde{r}_j = {} & r_i D_r (1 - D_r r_0^2 / \tilde{D}_f)\, r_j + f_i D_r (1 - D_f / \tilde{D}_f)\, f_j - \\ & - f_i (D_r r_0 D_f / \tilde{D}_f)\, r_j - r_i (D_r r_0 D_f / \tilde{D}_f)\, f_j = \\ & = (r_i - r_0 f_i)\, (D_r D_f / \tilde{D}_f)\, (r_j - r_0 f_j) \end{aligned}$$

This proves (13) and (11). The proof is completed by substituting r_j from (13) into (16) which then gets the simplified form (14) with the coefficient k_r determined according to (12).

Remark (*a*). It is apparent from the derivation that instead of (14) it would be possible to use (16).

$$\tilde{f}_j = k_f f_j + k_r r_j\,; \quad k_f = D_f/\tilde{D}_f$$

However this would mean n multiplications more for one dyadic reduction.

Procedure DYDR

The PASCAL procedure *DYDR* listed below performs one dyadic reduction.
Type introduced:
TYPE *row* = ARRAY [jmin .. jmax] OF REAL;
where jmin and jmax are suitable integer constants.
Parameters:

r of type *row*	... reduced row
f of type *row*	... reducing row
Dr, *Df* of type REAL	... weights of corresponding dyads
jr of type INTEGER	... index of the element *r*[*jr*] which is to be zeroed
jl, *jh* of type INTEGER	... lower and upper bounds of the range within which the both rows are modified

```
PROCEDURE DYDR (VAR r,f: row; VAR Dr, Df: REAL; jr, jl, jh: INTEGER);
CONST mzro = 1E-20;                                   {see Remark (b)}
VAR j: INTEGER; kr, kD, r0: REAL;
BEGIN
IF Dr < mzro THEN Dr := 0;                            {see Remark (c)}
r0 := r[jr];
kD := Df;
kr := r0 * Dr;
Df := kD + r0 * kr;
IF Df > mzro
  THEN BEGIN kD := kD/Df; kr := kr/Df END
  ELSE BEGIN kD := 1; kr := 0 END;                    {see Remark (d)}
Dr := kD * Dr;
FOR j := jl TO jh DO
  BEGIN
  r[j] := r[j] - r0 * f[j];
  f[j] := f[j] + kr * r[j]
  END
END;
```

Remark (*b*). The constant mzro ("machine zero"), which is used to check very small numbers, has to be chosen with respect to the accuracy of the computing device. The value 1E-20 has been found suitable for 4 bytes floating point arithmetic.

Remark (*c*). In applications in which the procedure *DYDR* will be used it may well happen that the weight *Dr* converges to zero and the possible underflow must be checked. If it is done automatically by the compiler then the marked statement can be omitted.

Remark (*d*). As mentioned before, in all cases when the updated weight *Df* is zero the dyadic reduction could theoretically be skipped. However, for numerical reasons it is advantageous to process the rows using the same algorithm with $kD = 1$ and $kr = 0$. Then the row r is normalized so that $r[jr]$ be zero even when the row has zero or very small weight *Dr*.

2.3. Decomposition and minimization of nonnegative definite quadratic forms

A classical problem which is, in various modifications, repeatedly met in both LQ optimum control synthesis and estimation is decomposition of a nonnegative definite quadratic form in the following sense. Consider the quadratic form

$$q(x) = x'Qx = \begin{bmatrix} x_a \\ x_b \end{bmatrix}' \quad Q \begin{bmatrix} x_a \\ x_b \end{bmatrix} \tag{17}$$

where x is a vector-valued variable, x_a and x_b are its components, $x' = [x_a', x_b']$ and Q is a numerically given matrix. It is assumed that the quadratic form (17), and thus also the matrix Q, are nonnegative definite, i.e. it holds $q(x) \leqq 0$ for any x. The problem is to decompose the quadratic form into two quadratic terms in such a way that one of these terms depends only on the second component x_b of the vector x.

$$q(x) = (y_a + Fx_b)' Q_1(x_a + Fx_b) + x_b'Q_2x_b \tag{18}$$

If we succeed to perform this decomposition then, at the same time, also the problem of minimization of the quadratic form with respect to the first component x_a is solved.

If the original quadratic form (17) is nonnegative definite then, according to Sylvester's law of inertia (see e.g. [9]), the both quadratic forms on the right-hand side of (18) must be also nonnegative definite. The second one does not depend on the variable component x_a while the first one can be zeroed by its choice. Since zero is the minimum value which a nonnegative definite quadratic form can achieve it holds for any x_b

$$\min_{x_a} q(x) = x_b'Q_2x_b$$

and the minimizing argument (not necessarily unique since F and Q_1 are not necessarily unique) is

$$\arg\min_{x_a} q(x) = x_a^* = -Fx_b$$

Standard procedures of matrix algebra commonly used to determine the matrices

Q_1, Q_2 and F are not suitable for numerical computation. They require an inversion (or some generalized inversion) of a certain submatrix and, namely, do not guarantee the nonnegative definiteness of the matrices Q_1 and Q_2.

From the physical nature of the problems we are going to solve it follows that all quadratic forms we shall deal with are nonnegative definite. Quadratic criteria as well as exponents in normal probability distributions cannot be negative. When calculating numerically it is necessary to design the algorithms in such a way that this important property be maintained also when the rounding errors are present, otherwise the sense of the solved problem could be lost and the computation could collapse. For instance, when the rounding errors cause that a quadratic criterion becomes indefinite then its minimum is minus infinity and the minimizing procedure diverges accordingly if the nonnegative definiteness is not rectified. This is of extreme importance when the matrices of quadratic forms are singular or almost singular, which is often the case, and particularly when the computation is performed on microprocessors with a short word length.

Even with reduced precision of arithmetic operations the nonnegatige definiteness of quadratic forms can be quaranteed when their matrices are systematically considered in factorized forms

$$Q = M'DM \tag{19}$$

and when all numerical calculations are performed only on the factors D and M. The matrix D is diagonal and can be stored as a vector of dimension ∂D, while M is, in general, a rectangular matrix of dimensions $\partial D \times \partial x$.

If M_i is a row-vector introduced as the ith row of the matrix M then (19) can be expressed as a sum of weighted dyads

$$Q = \sum_{i=1}^{\partial D} M_i' D_i M_i \tag{20}$$

and the quadratic form (17) can be expressed as a sum of squares

$$q(x) = \sum_{i=1}^{\partial D} D_i (M_i x)^2 = \sum_{i=1}^{\partial D} D_i \Big(\sum_{j=1}^{\partial x} M_{ij} x_j \Big)^2 \tag{21}$$

If all weights D_i are nonnegative then also the quadratic form is nonnegative. The algorithms we shall use cannot produce negative weights and therefore the nonnegativity even does not need to be checked.

Note that D has been introduced as a diagonal matrix only because of the compatibility of the matrix product on the right-hand side of (19). Equally well D could be interpreted as a column-vector the element D_i of which gives the weight to the square produced by the row of M with the same index. To emphasize this interpretation and to shorten the writing the following notation will be often used for quadratic forms.

$$q(x) = |D; Mx|$$

The factorization (19) is not unique and can be modified in various ways leaving

the value of the quadratic form unchanged for any variable vector x. This opens the possibility for direct and elegant numerical solutions of our problems. As the full matrix of any nonnegative quadratic form can be expressed as a weighted sum of dyads (20) the above described algorithm of dyadic reduction is an excellent tool for this purpose.

Consider, for instance, the quadratic form

$$q(x_a, x_b) = \left| \begin{bmatrix} D_a \\ D_b \end{bmatrix}; \begin{bmatrix} M_a & M_{ab} \\ M_{ba} & M_b \end{bmatrix} \begin{bmatrix} x_a \\ x_b \end{bmatrix} \right|$$

and let us perform its decomposition (18). The algorithm of dyadic reduction, as described in the foregoing paragraph, assumes that the reducing row, called f, has the element $f_0 = 1$ in the proper place. In later applications we shall organize the computations so that this condition will allways be fulfilled. Here, when considering a general situation, we have to apply the following trick. The quadratic form does not change if it is extended by a term with zero weight.

$$q(x_a, x_b) = q(x_a, x_b) + |0; Ix_a| = \left| \begin{bmatrix} 0 \\ D_a \\ D_b \end{bmatrix}; \begin{bmatrix} I & 0 \\ M_a & M_{ab} \\ M_{ba} & M_b \end{bmatrix} \begin{bmatrix} x_a \\ x_b \end{bmatrix} \right|$$

Now it is possible to use the dyadic reduction so that the first row of the extended matrix is used to reduce to zero the first columns of submatrices M_a und M_{ba}. Using the second row the second columns of these submatrices are zeroed, and so on until the both submatrices M_a and M_{ba} are zeroed. After this repeated application of the procedure *DYDR* the quadratic form is modified as follows.

$$q(x_a, x_b) = \left| \begin{bmatrix} D_0 \\ \tilde{D}_a \\ \tilde{D}_b \end{bmatrix}; \begin{bmatrix} U & G \\ 0 & \tilde{M}_{ab} \\ 0 & \tilde{M}_b \end{bmatrix} \begin{bmatrix} x_a \\ x_b \end{bmatrix} \right| =$$

$$= |D_0; Ux_a + Gx_b| + \left| \begin{bmatrix} \tilde{D}_a \\ \tilde{D}_b \end{bmatrix}; \begin{bmatrix} \tilde{M}_{ab} \\ \tilde{M}_b \end{bmatrix} x_b \right|$$

In this way the quadratic form is decomposed as required. The second term determines the minimum with respect to x_a and the minimizing argument can be determined from the equation

$$Ux_a^* + Gx_b = 0 \tag{22}$$

which, if required, can be easily solved without using the operation of division since U is a monic UT-matrix (upper triangular with unit diagonal elements).

Note that the solution of (22) is unique even when the problem is singular and more than one minimizing arguments exist. However, in such a case one or more weights in D_0 are zeros and the corresponding equations in the system of linear equations (22) do not need to be satisfied since their residua do not influence the quadratic

form. Practically it means that when the zero weighted equation is met during the solution of (22) then the component of x_a which is being determined by this equation can be chosen arbitrarily.

2.4. Integration and conditioning in normal probability distributions

Integration and conditioning in multivariate normal probability distributions are standard steps in real-time estimation, filtering, and prediction. Again, it can be stated that the classical solutions of these steps, as given e.g. in [8] and as implicitly contained in the Riccati equation of the Kalman filter or in standard recursive least squares estimation, are not suitable for practical computation as they do not guarantee numerically the positive definiteness of computed covariance matrices. For these numerical reasons all covariance matrices will be propagated in factorized forms.

The present paragraph will be devoted to the proof of the following

Result (2A): Conditional and marginal distributions for given normal joint probability distributions.

Consider a normal p.d.f. $\mathsf{p}(x)$ of a random vector x with an expected (mean) value $\hat{x}$ and with a covariance matrix $\mathrm{Var}[x] = R$ given as the matrix product

$$R = MDM' \tag{23}$$

where D is a diagonal matrix with nonnegative diagonal elements D_i, $i = 1, 2, \ldots, \partial D$, and M is a rectangular matrix of dimensions $(\partial x \times \partial D)$, $\partial D \geqq \partial x$. Suppose that D and M are given numerically and that the dyadic reduction is applied to modify the factorization (23)

$$MDM' = L\tilde{D}L' \tag{24}$$

so that $\tilde{D}$ is again diagonal, $\partial\tilde{D} = \partial x \leqq \partial D$, but L is a monic LT-matrix.

If the random vector x is partitioned into two subvectors

$$x = \begin{bmatrix} x_a \\ x_b \end{bmatrix}, \quad \hat{x} = \begin{bmatrix} \hat{x}_a \\ \hat{x}_b \end{bmatrix}$$

and correspondingly also the matrices L and $\tilde{D}$

$$L = \begin{bmatrix} L_a & 0 \\ G & L_b \end{bmatrix}, \quad \tilde{D} = \begin{bmatrix} \tilde{D}_a & 0 \\ 0 & \tilde{D}_b \end{bmatrix} \tag{25}$$

then the c.p.d.f. $\mathsf{p}(x_b \mid x_a)$ is normal with the expected value given by

$$\mathsf{E}[x_b \mid x_a] = \hat{x}_{b|a} = \hat{x}_b + G\varepsilon\,, \quad L_a\varepsilon = x_a - \hat{x}_a \tag{26}$$

The covariance matrix of x_b conditioned on the given (observed) value of x_a is

$$\mathrm{Var}[x_b \mid x_a] = L_b\tilde{D}_bL_b' \tag{27}$$

The marginal p.d.f. $\mathrm{p}(x_a)$ has the expected value $\hat{x}_a$ and the covariance matrix

$$\mathrm{Var}\,[x_a] = L_a \tilde{D}_a L_a' \tag{28}$$

Remark (*e*). Note that the both resulting covariance matrices are obtained in factorized forms with numerically guaranteed nonnegative definiteness as the algorithm of dyadic reduction used to perform the modification (24) cannot produce negative weights $\tilde{D}$ if the original ones D are nonnegative. Since L is a monic LT-matrix ε in (26) can be calculated without employing numerical division. Frequently $\partial x_a = 1$, then $L_a = 1$ and ε is the prediction error, $\varepsilon = x_a - \hat{x}_a$.

Proof. Since the modification of the factorization (24) does not change the given matrix R (23) it is sufficient to consider only the modified factorization $R = L\tilde{D}L'$, where the monic LT-matrix L is always invertible and $\det(L) = 1$. Clearly

$$\det(R) = \prod_{i=1}^{\partial x} \tilde{D}_i$$

For the sake of simplicity the proof will be given only for the regular case when all $\tilde{D}_i$ are arbitrarily small but finite. The degenerate case when one or more $\tilde{D}_i$ are zero could be handled using characteristic functions instead of p.d.f.'s. However, the result would remain the same.

Using our notation for quadratic forms the normal p.d.f. can be rewritten in the following way.

$$\mathrm{p}(x) = (2\pi)^{-\partial x/2} (\det(R))^{-1/2} \exp\{-\tfrac{1}{2}(x - \hat{x})' R^{-1}(x - \hat{x})\} =$$
$$= (2\pi)^{-\partial x/2} \Big(\prod_{i=1}^{\partial x} \tilde{D}_i\Big)^{-1/2} \exp\{-\tfrac{1}{2}|\tilde{D}^{-1}; L^{-1}(x - \hat{x})|\}$$

It can be easily verified that for L^{-1}, when partitioned similarly to (25), it holds

$$L^{-1} = \begin{bmatrix} L_a^{-1} & 0 \\ L_b^{-1} G L_a^{-1} & L_b^{-1} \end{bmatrix}$$

Then the quadratic form in the exponent of the p.d.f. can be decomposed as follows

$$|\tilde{D}^{-1}; L^{-1}(x - \hat{x})| = |\tilde{D}_a^{-1}; L_a^{-1}(x_a - \hat{x}_a)| +$$
$$+ |\tilde{D}_b^{-1}; -L_b^{-1} G L_a^{-1}(x_a - \hat{x}_a) + L_b^{-1}(x_b - \hat{x}_b)| =$$
$$= |\tilde{D}_a^{-1}; L_a^{-1}(x_a - \hat{x}_a)| + |\tilde{D}_b^{-1}; L_b^{-1}(x_b - \hat{x}_{b|a})|$$

where in agreement with (26)

$$\hat{x}_{b|a} = \hat{x}_b + G L_a^{-1}(x_a - \hat{x}_a)$$

This decomposition is the main step in the proof as it enables the following factorization of the joint p.d.f.

$$\mathrm{p}(x_a, x_b) = (2\pi)^{-\partial x_a/2} \Big(\prod_{i=1}^{\partial x_a} \tilde{D}_{a,i}\Big)^{-1/2} \exp\{-\tfrac{1}{2}|\tilde{D}_a^{-1}; L_a^{-1}(x_a - \hat{x}_a)|\} \cdot$$
$$(2\pi)^{-\partial x_b/2} \Big(\prod_{j=1}^{\partial x_b} \tilde{D}_{b,j}\Big)^{-1/2} \exp\{-\tfrac{1}{2}|\tilde{D}_b^{-1}; L_b^{-1}(x_b - \hat{x}_{b|a})|\}$$

To prove that this factorization corresponds to the relation

$$\mathrm{p}(x_a, x_b) = \mathrm{p}(x_a)\,\mathrm{p}(x_b \mid x_a)$$

it is sufficient to perform the integration

$$\mathrm{p}(x_a) = \int \mathrm{p}(x_a, x_b)\,\mathrm{d}x_b$$

The transformation $z = L_b^{-1}(x_b - \hat{x}_{b|a})$ with the Jacobian $\det(L_b^{-1}) = 1$ reduces the multivariate integral into the product of univariate integrals.

$$\int \exp\{-\tfrac{1}{2}|\tilde{D}_b^{-1}; L_b^{-1}(x_b - \hat{x}_{b|a})|\}\,\mathrm{d}x_b = \prod_{j=1}^{\partial x_b} \int \exp\left\{-\frac{z_j^2}{2\tilde{D}_{b,j}}\right\} \mathrm{d}z_j = \prod_{j=1}^{\partial x_b} (2\pi\tilde{D}_{b,j})^{1/2}$$

To complete the proof is now trivial.

2.5. Linear composition of normal probability distributions

Suppose that the normal p.d.f.'s $\mathrm{p}(x)$ and $\mathrm{p}(y \mid x)$ are given by their expected values and covariance matrices possibly LD-factorized.

$$\mathrm{E}[x] = \hat{x}\,, \quad \mathrm{Var}\,[x] = R_x = L_x D_x L_x' \tag{29}$$

$$\mathrm{E}[y \mid x] = Mx + k\,, \quad \mathrm{Var}\,[y \mid x] = R_{y|x} = L_{y|x} D_{y|x} L_{y|x}' \tag{30}$$

One of standard steps repeatedly met when solving LQG problems is determination of the joint p.d.f.

$$\mathrm{p}(x, y) = \mathrm{p}(y \mid x)\,\mathrm{p}(x) \tag{40}$$

Result (2B): Linear composition of normal probability distributions.

The joint p.d.f. (40) is normal with the expected value

$$\mathrm{E}\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \hat{x} \\ M\hat{x} + k \end{bmatrix} \tag{41}$$

and with the covariance matrix

$$R = \begin{bmatrix} \mathrm{Var}\,[x] & \mathrm{Cov}\,[x, y] \\ \mathrm{Cov}\,[y, x] & \mathrm{Var}\,[y] \end{bmatrix} = LDL' \tag{42}$$

determined by the relations

$$\mathrm{Var}\,[y] = R_{y|x} + MR_xM' \tag{43}$$

$$\mathrm{Cov}\,[y, x] = \mathrm{Cov}\,[x, y]' = MR_x \tag{44}$$

$$L = \begin{bmatrix} L_x & 0 \\ ML_x & L_{y|x} \end{bmatrix}, \quad D = \begin{bmatrix} D_x & 0 \\ 0 & D_{y|x} \end{bmatrix} \tag{45}$$

Proof. When substituting

$$\mathrm{p}(x) = (2\pi)^{-\partial x/2} \det(D_x)^{-1/2} \exp\{-\tfrac{1}{2}|D_x^{-1}; L_x^{-1}(x - \hat{x})|\}$$

$$\mathrm{p}(y \mid x) = (2\pi)^{-\partial y/2} \det(D_{y|x})^{-1/2} \exp\{-\tfrac{1}{2}|D_{y|x}^{-1}; L_{y|x}^{-1}(y - Mx - k)|\}$$

into (40) it is clear that the quadratic forms in the exponents are added. The following rearrangement of this sum of quadratic forms yields the proof.

$$\left|D_x^{-1}; L_x^{-1}(x - \hat{x})\right| + \left|D_{y|x}^{-1}; L_{y|x}^{-1}(y - Mx - k)\right| =$$

$$= \left|D_x^{-1}; L_x^{-1}(x - \hat{x})\right| + \left|D_{y|x}^{-1}; L_{y|x}^{-1}(y - M\hat{x} - k) - L_{y|x}^{-1}M(x - \hat{x})\right| =$$

$$= \left| \begin{bmatrix} D_x^{-1} & 0 \\ 0 & D_{y|x}^{-1} \end{bmatrix}; \begin{bmatrix} L_x^{-1} & 0 \\ -L_{y|x}^{-1}M & L_{y|x}^{-1} \end{bmatrix} \begin{bmatrix} x - \hat{x} \\ y - M\hat{x} - k \end{bmatrix} \right|$$

Apparently

$$L = \begin{bmatrix} L_x^{-1} & 0 \\ -L_{y|x}^{-1}M & L_{y|x}^{-1} \end{bmatrix}^{-1} = \begin{bmatrix} L_x & 0 \\ ML_x & L_{y|x} \end{bmatrix}$$

This proves (41) and (45). The relations (43) and (44) are obtained by substituting (45) into the right-hand side of (42).

REFERENCES

[8] T. W. Anderson: An Introduction to Multivariate Statistical Analysis. John Wiley, New York 1958.

[9] A. Kurosh: Higher Algebra. MIR Publishers, Moscow 1984.

[10] V. Peterka: Bayesian approach to system identification. Chapter 8 in: Trends and Progress in System Identification (P. Eykhoff, editor), IFAC Series for Graduates, Research Workers and Practising Engineers, Pergamon Press, Oxford 1981.

3. LINEAR STOCHASTIC MODELS

In this section models are introduced which will be used to represent the process to be controlled.

The section commences by general discussion of the question: What mathematical models are needed for control-design purposes? The question is answered in terms of conditional probability distributions and on this basis the concept of a linear stochastic model of the controlled process is introduced. Also the concept of the state of the process model is discussed in terms of conditional probability distributions in the first paragraph.

In the second paragraph the nonparametric input-output model is suitably parametrized to obtain the ARMA form of the model. Both positional and incremental (ARIMA) types of the model are considered.

The Delta form of the input-output model is introduced in the third paragraph where also PASCAL procedures for transformations of the ARMA form into the Delta form and vice versa can be found.

The ARMA and Delta input-output models of both positional and incremental types are transformed into a canonical state-space model in the fourth paragraph.

In the last paragraph of this section the relation between Delta models and stochastic differential equations is discussed and a simple PASCAL procedure for digital simulation of a linear continuous stochastic process is given.

3.1. Process model in general

Suppose that, starting with $t = 1$, a control-system designer has the possibility to observe the process up to and including the sampling period indexed by $t_0 \geqq 0$. His task is to design a control strategy for the next T sampling periods in some optimal sense. Note that for $t_0 = 0$ no prior observation is made. If the designer picked a particular strategy and performed the experiment he could judge the control performance according to the actual values of the data he can observe

(1)

$$\{y(t_0 + 1 \cdot\cdot t_0 + T), v(t_0 + 1 \cdot\cdot t_0 + T), u(t_0 + 1 \cdot\cdot t_0 + T), w(t_0 + 1 \cdot\cdot t_0 + T)\}$$

However, since the control strategy has to be chosen optimally and in advance the designer must have models available which make it possible to compare all admissible strategies before the input $u(t_0 + 1)$ is applied. The choice of a suitable criterion for this comparizon will be discussed in Section 5. Here, it is important to note that for any criterion which contains only the observable data it is sufficient to be able to determine the probability distribution of the future data (1) conditioned on the data available up to the sampling period t_0 for any control strategy which might be applied. In case of program control or of a given fixed setpoint the future command signal $w(t_0 + k)$, $k > 0$, is a priori given and can be put into the condition. Here the more general servo case will be considered in which the future command signal is also uncertain.

By suitably repeated application of the decomposition rule (2.2) the joint c.p.d.f. of the future data (1) can be written as follows

(2)

$$\begin{aligned}
&\mathsf{p}(y(t_0 + 1 \cdot\cdot t_0 + T), v(t_0 + 1 \cdot\cdot t_0 + T), u(t_0 + 1 \cdot\cdot t_0 + T), w(t_0 + 1 \cdot\cdot t_0 + T) \mid \\
&\qquad \mid (y(1 \cdot\cdot t_0), v(1 \cdot\cdot t_0), u(1 \cdot\cdot t_0), w(1 \cdot\cdot t_0)) = \\
&= \prod_{t=t_0+1}^{t_0+T} \mathsf{p}(y(t) \mid y(1 \cdot\cdot t-1), v(1 \cdot\cdot t), u(1 \cdot\cdot t), w(1 \cdot\cdot t)) \\
&\qquad \mathsf{p}(v(t) \mid y(1 \cdot\cdot t-1), v(1 \cdot\cdot t-1), u(1 \cdot\cdot t), w(1 \cdot\cdot t)) \\
&\qquad \mathsf{p}(u(t) \mid y(1 \cdot\cdot t-1), v(1 \cdot\cdot t-1), u(1 \cdot\cdot t-1), w(1 \cdot\cdot t)) \\
&\qquad \mathsf{p}(w(t) \mid y(1 \cdot\cdot t-1), v(1 \cdot\cdot t-1), u(1 \cdot\cdot t-1), w(1 \cdot\cdot t-1))
\end{aligned}$$

The particular factors in this multiple product have the following interpretation.

Control strategy

The c.p.d.f.

(3) $$\mathsf{p}(u(t) \mid v(1 \cdot\cdot t-1), y(1 \cdot\cdot t-1), u(1 \cdot\cdot t-1), w(1 \cdot\cdot t))$$

represents the transformation, in general stochastic, by which each $u(t)$ for $t = t_0 + 1, \ldots, t_0 + T$ is generated on the basis of the data which are available at the given time instant. In fact, the decomposition in (2) has been made in such a way that the condition part of (3) contains all data on which an admissible control strategy can operate. Hence, the set of c.p.d.f.'s (3) is what the designer has to determine when solving the control problem. In Section 5 it will be proved that under very general conditions the optimal control strategies are deterministic, i.e. that the optimal input $u(t)$ has to be determined as a deterministic function of the available data

$$u(t) = \mathrm{f}_t(u(1 \cdot\cdot t-1), y(1 \cdot\cdot t-1), v(1 \cdot\cdot t-1), w(1 \cdot\cdot t))$$

Then the c.p.d.f. (3) degenerates into a Dirac δ-function

$$\delta(u(t) - \mathrm{f}_t u(1 \cdot\cdot t-1), y(1 \cdot\cdot t-1), v(1 \cdot\cdot t-1), w(1 \cdot\cdot t)))$$

and the control problem is reduced into determination of the function f_t for each t.

Command signal

As mentioned above, if the command signal is predetermined for the entire control horizon it can be considered as a given condition for all c.p.d.f.'s involved. However, if it is a priori uncertain its evolution within the considered time span $(t_0 + 1, \cdot\cdot t_0 + T)$ must be modelled. When formulating the control problem it is natural to assume that there does not exist any hidden feedback or feedforward from other signals in the control loop in Fig. 1 which might influence the evolution of the command signal w. This mean that

$$\mathrm{p}(w(t) \mid y(1 \cdot\cdot t-1), v(1 \cdot\cdot t-1), u(1 \cdot\cdot t-1), w(1 \cdot\cdot t-1)) = \mathrm{p}(w(t) \mid w(1 \cdot\cdot t-1)) \tag{4}$$

To define this c.p.d.f. for all t of interest a simple model suitable for the case of positional servo will be introduced when it will be needed in Section 5.

Measurable external disturbance

The digital controller in Fig. 1 can operate on the disturbance v if it is available for measurement. Assuming that the disturbance v is external means that

$$\mathrm{p}(v(t) \mid y(1 \cdot\cdot t-1), v(1 \cdot\cdot t-1), u(1 \cdot\cdot t), w(1 \cdot\cdot t)) = \mathrm{p}(v(t) \mid v(1 \cdot\cdot t-1)) \tag{5}$$

According to this prior information the evolution of the external disturbance can be considered as an autonomous process, i.e. as the measurable output of the uncontrollable part of the external world sometimes called the enviroment. A suitable model for this external process defining the c.p.d.f. (5) will be introduced in Section 5.

Controlled process

The c.p.d.f. which remains to be interpreted in the product on the right-hand side of (2) describes the stochastic transformation performed by the controlled process itself. Since there does not exist any direct connection between the command signal w and the controlled process output y it can beassumed that

$$\begin{aligned}(6)\qquad \mathsf{p}(y(t) \mid y(1 \cdot\cdot t-1), v(1 \cdot\cdot t), u(1 \cdot\cdot t), w(1 \cdot\cdot t)) &= \\ = \mathsf{p}(y(t) \mid y(1 \cdot\cdot t-1), v(1 \cdot\cdot t), u(1 \cdot\cdot t)) &= \\ = \mathsf{p}(y(t) \mid t-1; v(t), u(t))\end{aligned}$$

The first equality says that the conditioning on $w(1 \cdot\cdot t)$ is superfluous as the given history of the command signal w can influence the output $y(t)$ only through the data which are already present in the condition. The second equality in (6) just recalls the abbreviated notation introduced in the paragraph 1.2. By $t-1$ in the condition part the conditioning on all data observed on the controlled process up to and including the sampling period $t-1$ is indicated.

Process model. Any mathematical model which defines the family of the c.p.d.f.'s

$$\mathsf{p}(y(t) \mid t-1; v(t), u(t)) \quad \text{for} \quad t = t_0 + 1, t_0 + 2, \ldots, t_0 + T$$

will be called the process model.

Linear process model. The process model is called linear if the mean value of the c.p.d.f. (6), i.e. the expected value of $y(t)$, can be expressed as a linear function of the data in the condition, and if the variance of $y(t)$ does not depend on these data.

$$\begin{aligned}(7)\qquad \mathsf{E}[y(t) \mid t-1; v(t), u(t)] = \hat{y}(t \mid t-1; v(t), u(t)) &= \\ = k_y(t) + g_0(t)\, u(t) + h_0(t)\, v(t) + &\\ + \sum_{k=1}^{t-1} [-f_k(t)\, y(t-k) + g_k(t)\, u(t-k) + h_k(t)\, v(t-k)]&\end{aligned}$$

$$(8)\qquad \mathrm{Var}\,[y(t) \mid t-1; v(t), u(t)] = R_y(t)$$

The linear model will be called normal if in addition to (7) and (8) it is also assumed that the c.p.d.f. (6) is Gaussian (normal).

If $e(t)$ is introduced as the difference between the true value $y(t)$ and its expected value (7)

$$e(t) = y(t) - \hat{y}(t \mid t-1; v(t), u(t))$$

then the model can be written in the form

$$(9)\qquad \sum_{k=0}^{t-1} f_k(t)\, y(t-k) = \sum_{k=0}^{t-1} [g_k(t)\, u(t-k) + h_k(t)\, v(t-k)] + k_y(t) + e(t)$$

where $f_0(t) = I$. The stochastic term $e(t)$ has the variance (8) and its expected value is, as readily seen, zero.

Multi-output case. The general form of a linear model (9) can hold for a single-input single-output process as well as for a process with more outputs and more inputs. The only difference is that in the latter case the coefficients (in general time varying) are matrices of appropriate dimensions. However, in multivariate case it will appear convenient to modify the model in the following way.

Suppose that the covariance matrix (8), which must be nonnegative definite, is available in the factorized form

$$R_y(t) = L_y(t)\, D_e(t)\, L_y'(t)$$

where $L_y(t)$ is a monic LT-matrix (always and simply invertible) and $D_e(t)$ is diagonal with nonnegative diagonal entries. Let us multiply by $L_y^{-1}(t)$ both sides of the model equation (9). This, in general, changes all the coefficients. We shall not change the notation but from now on the coefficient $f_0(t)$ will be a monic LT-matrix

$$f_0(t) = L_y^{-1}(t) \tag{10}$$

which is reduced into $f_0(t) = 1$ only in the single-output case. The advantage of this modification is that the stochastic term $e(t)$, now redefined as

$$e(t) = L_y^{-1}(t)\,(y(t) - \hat{y}(t \mid t-1; v(t), u(t))) \tag{11}$$

has not only the zero mean, both conditional and unconditional

$$\mathsf{E}[e(t) \mid t-1; v(t), u(t)] = \mathsf{E}[e(t)] = 0 \tag{12}$$

but also uncorrelated components

$$\mathrm{Var}\,[e(t)] = \mathsf{E}[e(t)\, e'(t)] = D_e(t) \tag{13}$$

It can be easily proved (see e.g. [10] § 3.1) that it also holds

$$\mathsf{E}[e(t)\, e'(t-k)] = 0 \quad \text{for} \quad k \neq 0 \tag{14}$$

$$\mathsf{E}[e(t)\, y'(t-k)] = 0 \quad \text{for} \quad k > 0 \tag{15}$$

$$\mathsf{E}[e(t)\, u'(t-k)] = 0 \quad \text{for} \quad k \geqq 0 \tag{16}$$

$$\mathsf{E}[e(t)\, v'(t-k)] = 0 \quad \text{for} \quad k \geqq 0 \tag{17}$$

Process delay (dead time, transport lag). In many practical cases the response to a change of the input $u(t)$ does not influence the successive sample $y(t)$ but it appears at the output only after T_u sampling periods. Then it holds

$$\mathsf{p}(y(t) \mid t-1; v(t), u(t)) = \mathsf{p}(y(t) \mid y(1 \cdots t-1), v(1 \cdots t), u(1 \cdots t-T_u))$$

In the linear model (9) this means that the leading coefficients $g_k(t)$, $k < T_u$, are zero. Similar delay T_v can exist also in the channel from the measurable external disturbance v.

State-space model

If there exist a quantity $s(t)$ of finite and fixed dimension (not necessarily accessible to measurement) such that

$$\mathrm{p}(y(t), s(t) \mid y(1..t-1), v(1..t), u(..t), s(t-1)) = \\ = \mathrm{p}(y(t), s(t) \mid v(t), u(t), s(t-1)) \tag{18}$$

and

$$\mathrm{p}(s(t) \mid t; v(t+1), u(t+1)) = \mathrm{p}(s(t) \mid t) \tag{19}$$

then $s(t)$ is called the state of the controlled process.

The condition (19) restricts the general definition of a state (18) to the state of just the controlled process itself. This can be shown using the definition of the external measurable disturbance (5) and the natural conditions of control discussed in paragraph 2.1 (see (2.5)). The possible state of the generator of the external disturbance is not included in $s(t)$.

By integrating out the output $y(t)$ in (18) the c.p.d.f.

$$\mathrm{p}(s(t) \mid v(t), u(t), s(t-1)) \tag{20}$$

is obtained which describes the evolution of the state itself. The c.p.d.f.

$$\mathrm{p}(y(t) \mid v(t), u(t), s(t-1)) \tag{21}$$

obtainable by integrating out $s(t)$ in (18) relates the process output to the state of the model.

Introduction of a suitable state, if it exists, can reduce the computational load significantly. Later on a linear state-space model of special (canonical) form will be constructed which defines the joint c.p.d.f. (18).

Given the state-space model the c.p.d.f. (6) is determined by the formula

$$\mathrm{p}(y(t) \mid t-1; v(t), u(t)) = \\ = \int \mathrm{p}(y(t) \mid v(t), u(t), s(t-1))\, \mathrm{p}(s(t-1) \mid t-1)\, \mathrm{d}s(t-1) \tag{22}$$

which shows that when operating with a state-space model it is necessary to evolve $\mathrm{p}(s(t) \mid t)$. This is the problem of state estimation which will be solved in Section 4.

3.2. Regression models and ARMA models

The general form of a linear input-output model (9) is not of much practical value. To make it practicable it is necessary to express its increasing number of coefficients, in general time varying, through a finite and possibly low number of constant (or at least temporarily constant) parameters. This parametrization can be done in different ways and is subject to some additional assumptions. Since any mathematical model can be only a simplified image of the modelled reality the same process can be described, more or less accurately, by different models. The point of the

modeling effort is to choose the model structure so that the model be as simple as possible, easy to handle (including parameter estimation), and at the same time it should cover sufficiently broad class of practical cases. These requirements are, of course, contradictory and some compromise has to be accepted. We shall proceed so that simple and intuitively well understandable regression models of positional and incremental type will be introduced first and then they will be extended to the more general model ARMA.

Regression model of positional type

Often it can be assumed that only a finite and fixed length of the past input-output history is significant for the prediction of the process output $y(t)$. Then

$$\mathrm{p}(y(t) \mid t-1; v(t), u(t)) = \mathrm{p}(y(t) \mid y(t-N..t-1), v(t-N..t), u(t-N..t))$$

where N determines the past history considered. The corresponding linear time invariant process model (9) can be written for $t > N$ in the form

$$\sum_{k=0}^{N} f_k y(t-k) = \sum_{k=0}^{N} g_k u(t-k) + \sum_{k=0}^{N} h_k v(t-k) + k_y + e(t) \tag{23}$$

It is important to note that in practical applications, when the regression model is identified from real data, a sufficiently long past history must be incorporated into the process model, i.e. N must be chosen sufficiently large (dependent on the sampling period). Only under this condition it can be assumed that $\mathrm{p}(e(t) \mid t-1; v(t), u(t)) = \mathrm{p}(e(t))$ and the model can define $\mathrm{p}(y(t) \mid t-1; v(t), u(t))$ for $t > N$ as, for the given condition, the transformation between $e(t)$ and $y(t)$ is one-to-one. (See Section 2 for transformation of random variables.)

Recall that in multi-output case f_0 is a monic LT-matrix (10) and the components of $e(t)$ are mutually uncorrelated, $\mathrm{Var}\,[e(t)] = D_e$. When the both sides of (23) are multiplied by $f_0^{-1} = L_y$ the model can be rewritten into the standard regression form

$$y(t) = -\sum_{i=1}^{N} a_i\, y(t-i) + \sum_{i=0}^{N} b_i\, u(t-i) + \sum_{i=0}^{N} d_i\, v(t-i) + \bar{k}_y + \varepsilon(t) \tag{24}$$

where

$$a_i = L_y f_i\,, \quad b_i = L_y g_i\,, \quad d_i = L_y h_i\,, \quad \bar{k}_y = L_y k_y$$

and

$$\mathrm{Var}\,[\varepsilon(t) = R_y = L_y D_e L_y' \tag{25}$$

Incremental regression model

Many practical processes are contaminated by stochastic disturbances which are nonstationary like drifts, unpredictable and unmeasurable load changes. Then the reference level in the stochastic input-output relation (24) cannot be considered as a constant k_y. In such cases it is more appropriate to relate the predicted output

$y(t)$ to the previous, already known, output $y(t-1)$ and to consider the incremental regression model

$$y(t) = y(t-1) + \sum_{i=1}^{N} a_i \Delta y(t-i) + \sum_{i=0}^{N} b_i \Delta u(t-i) + \sum_{i=0}^{N} d_i \Delta v(t-i) + \varepsilon(t) \tag{26}$$

Then, instead of (23) we have

$$\sum_{k=0}^{N} f_k \Delta y(t-k) = \sum_{k=0}^{N} g_k \Delta u(t-k) + \sum_{k=0}^{N} h_k \Delta v(t-k) + e(t) \tag{27}$$

Model ARMA

It is apparent that the larger is the memory size N of the regression models (23) or (27) the broader is the class of processes which can be described by these models. However, the larger is also the number of parameters which have to be determined. This may be critical in cases of very fast sampling rates when N must be chosen relatively large in order to incorporate a sufficiently long history of the process into the model and to guarantee the white noise properties of $e(t)$. To reduce the number of model parameters in such cases it is possible to proceed as follows.

Consider $N \to \infty$ in the regression model (23) and introduce, for generality, also the possible process delays T_u and T_v. Then, with obvious reindexing of the coefficients, the model can be written as follows.

$$\sum_{k=0}^{\infty} f_k y(t-k) = \sum_{k=0}^{\infty} g_k u(t-T_u-k) + \sum_{k=0}^{\infty} h_k v(t-T_v-k) + k_y + e(t) \tag{28}$$

In order to meet certain regularity conditions the coefficients in (28) have to satisfy the relations

$$\sum_{k=0}^{\infty} f_k^2 < \infty\,, \quad \sum_{k=0}^{\infty} g_k^2 < \infty\,, \quad \sum_{k=0}^{\infty} h_k^2 < \infty$$

To express these infinitely many coefficients through a finite number of parameters suppose that only the first $m+1$ cofficients f_k, g_k, h_k $(k = 0, \ldots, m)$ can be arbitrary while the rest of them for $k > m$ (the "tails") can be approximately described as a weighted sum of n_c suitably chosen exponentials so that for $k > m$

$$f_k = \sum_{j=1}^{n_c} \Phi_j \zeta_j^{-k}\,, \quad g_k = \sum_{j=1}^{n_c} \Gamma_j \zeta_j^{-k}\,, \quad h_k = \sum_{j=1}^{n_c} \Psi_j \zeta_j^{-k} \tag{29}$$

where $|\zeta_j| > 1$ for all j. Note that in multivariate case the weights Φ_j, Γ_j and Ψ_j are matrices of appropriate dimensions while ζ_j are scalars determining the base of exponentials.

Consider further the following weighted sum of neighboring cofficients f_k for $k > m + n_c$

$$f_k + \sum_{i=1}^{n_c} c_i f_{k-i} = \sum_{j=1}^{n_c} \Phi_j \zeta_j^{-k} \left(1 + \sum_{i=1}^{n_c} c_i \zeta_j^{i}\right)$$

The rearrangement on the right-hand side shows that the weighted sum can be made equal to zero if the scalar weights c_i are chosen so that ζ_i $(j = 1, 2, .., n_c)$ are roots of the polynomial

$$c(\zeta) = 1 + \sum_{i=1}^{n_c} c_i \zeta^i = \prod_{j=1}^{n_c} (1 - \zeta_j^{-1} \zeta) \tag{30}$$

Hence with this choice and $c_0 = 1$ it holds for $k > m + n_c$

$$\sum_{i=0}^{n_c} c_i f_{k-i} = 0\,, \quad \sum_{i=0}^{n_c} c_i g_{k-i} = 0\,, \quad \sum_{i=0}^{n_c} c_i h_{k-i} = 0 \tag{31}$$

To simplify the exposition it has been tacitly assumed that the roots ζ_j are distinct and real. However, from the theory of linear difference equations it is well known that the base of functions which satisfy the relations (31) and which are used to approximate f_k, g_k, h_k for $k > m$ can be made somewhat more general if also complex and multiple roots are admitted.

The important point is that if the coefficients of the linear nonparametric input-output model (28) satisfy the difference equations (31) then the moving average with scalar weights c_i taken over (28) for t, $t - 1, .., t - n_c$ has only $n + 1$ terms, $n = m + n_c$. For instance

$$\sum_{j=0}^{n_c} c_j \sum_{k=0}^{\infty} f_k y(t - j - k) = \sum_{j=0}^{n_c} c_j \sum_{i=j}^{\infty} f_{i-j} y(t - i) =$$

$$= \sum_{i=0}^{n_c} \Big(\sum_{j=0}^{i} c_j f_{i-j} \Big) y(t - i) + \sum_{i=n_c+1}^{n} \Big(\sum_{j=0}^{n_c} c_j f_{i-j} \Big) y(t - i) = \sum_{i=0}^{n} a_i\, y(t - i)$$

For the entire input-output relation (28) it is obtained

$$\sum_{i=0}^{n} a_i y(t - i) = \tag{32}$$

$$= \sum_{i=0}^{n} b_i\, u(t - T_u - i) + \sum_{i=0}^{n} d_i\, v(t - T_u - i) + \sum_{i=0}^{n_c} c_i\, e(t - i) + k_c$$

where, with $i_{nc} = \min(i, n_c)$,

$$a_i = \sum_{j=0}^{i_{nc}} c_j f_{i-j}\,, \quad b_i = \sum_{j=0}^{i_{nc}} c_j g_{i-j}\,, \quad d_i = \sum_{j=0}^{i_{nc}} c_j h_{i-j} \tag{33}$$

$$k_c = \Big(\sum_{i=0}^{n_c} c_i \Big) k_y \tag{34}$$

The model of this form was given the name ARMA [12] indicating that autoregressive (AR) and moving average (MA) terms are present. The way how the model has been introduced here was chosen to support interpretation of c-parameters suitable for our purposes. Later on it will be seen that these parameters are difficult to estimate in real time and a proper understanding of their role can facilitate their suitable prior choice.

In a similar way also the incremental ARMA model (ARIMA) can be introduced

as an extension of the incremental regression model.

$$\sum_{i=0}^{n} a_i \Delta y(t-i) = \tag{35}$$

$$= \sum_{i=0}^{n} b_i \Delta u(t - T_u - i) + \sum_{i=0}^{n} d_i \Delta v(t - T_v - i) + \sum_{i=0}^{n_c} c_i e(t-i)$$

If ζ is interpreted as backwards shift operator

$$\zeta f(t) = f(t-1) \tag{36}$$

then the positional ARMA model can be written in the form

$$a(\zeta)\, y(t) = b(\zeta)\, u(t - T_u) + d(\zeta)\, v(t - T_v) + k_c + c(\zeta)\, e(t) \tag{37}$$

and the incremental ARMA model in the form

$$a(\zeta)\, \Delta y(t) = b(\zeta)\, \Delta u(t - T_u) + d(\zeta)\, \Delta v(t - T_v) + c(\zeta)\, e(t) \tag{38}$$

where in both cases

$$a(\zeta) = \sum_{i=0}^{n} a_i \zeta^i, \quad b(\zeta) = \sum_{i=0}^{n} b_i \zeta^i, \quad d(\zeta) = \sum_{i=0}^{n} d_i \zeta^i \tag{39}$$

and $c(\zeta)$ is the polynomial (30) of order $n_c \leqq n$.

It is important to bear in mind that in multivariate cases a_i, b_i, and d_i are matrices but c_i are scalars. Recall also that the covariance matrix of $e(t)$ is diagonal

$$\mathrm{Var}\,[e(t)] = D_e \tag{40}$$

and a_0, according to (33), is a monic LT-matrix

$$a_0 = f_0 = L_y^{-1} \tag{41}$$

while $c_0 = 1$.

We have constructed the ARMA models as an extension of regression models for $N \to \infty$ assuming that an infinitely long past history of the process was available for observation. This is the reason why the condition of stability had to be imposed on the polynomial $c(\zeta)$, viz. $|\zeta_j| > 1$ in (30). However, if the ARMA model is understood as a generator of the process driven by a white noise $e(t)$ then the stability of $c(\zeta)$ does not need to be required. In Section 5 it will be shown that for a finite and growing length of observation the c-parameters have to be recalculated and updated in real time so that the truly applied c-parameters are time varying and converge to the coefficients of a stable polynomial (reflection of $c(\zeta)$) even when the original polynomial $c(\zeta)$ is unstable. Hence the assumption on stability of $c(\zeta)$ can be relaxed and a root of $c(\zeta)$ can lie also at the stability boundary. This favorable fact makes it posible to get rid of the constant k_y in the positional ARMA (37) by taking the difference

$$a(\zeta)\, \Delta y(t) = b(\zeta)\, \Delta u(t - T_u) + d(\zeta)\, \Delta v(t - T_v) + (1 - \zeta)\, c(\zeta)\, e(t) \tag{42}$$

Thus the positional ARMA can be considered as an incremental ARMA with the c-polynomial having one unit root.

Remark (*a*). Strictly taken the two above interpretations of the model ARMA are conceptually rather different namely in the meaning of the random variable $e(t)$. In the interpretation taken as the basis for our treatment $e(t)$ is understood as the difference between the true and expected value of $y(t)$ (suitable transformed in multivariate case) which can be eventually achieved after an infinitely long observation of the process. In the other interpretation $e(t)$ is understood as a fictitions, unobservable, and actually nonexisting driving white noise introduced as a useful modeling tool. Unlike $e(t)$ defined by (11) the driving white noise can never be reconstructed from the observed data (neither asymptotically) if the c-polynomial in the ARMA model is unstable. In the sequel we shall handle the ARMA models in such a way that these two cases do not need to be distinguished. We based our construction of the ARMA model on the former interpretation in order to show that a prior choice of scalar c-parameters actually means a choice of a base of exponentials used to approximate the "tails" in the general nonparametric linear time-invariant input-output model (28).

3.3. Delta models

Delta models, we are going to introduce, are theoretically equivalent to ARMA models. They are just an other form of ARMA models which appears to be numerically more robust especially in cases of fast sampling rates. We shall start the discussion by a very simple illustrative example.

Consider a deterministic continuous system of first order

$$\mathrm{d}y/\mathrm{d}\tau + \alpha y = \beta u$$

Suppose that the input u is manipulated digitally with a zero order hold as shown in Fig. 2. Let the continuous output y be sampled with the sampling period T_s in such

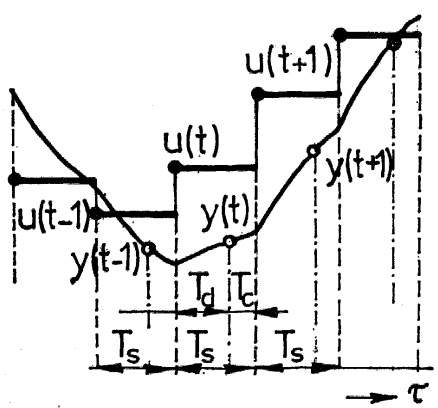

Fig. 2. Continuous process digitally controlled.

a way that T_d is the time interval between the time instants when the digital input $u(t)$ is applied and when the sample $y(t)$ is taken. Note that the fraction of the sampling period $T_c = T_s - T_d$ is available for computing the next digital input $u(t+1)$. Clearly, $0 < T_c \leqq T_s$. For such a simple example it is easy to derive that the sampled output and the digital input are related by the difference equation

$$y(t) + a_1 y(t-1) = b_0 u(t) + b_1 u(t-1)$$

where

$$a_1 = -\exp(-\alpha T_s), \quad b_0 = \frac{\beta}{\alpha}(1 - \exp(-\alpha T_d)), \quad b_1 = \frac{\beta}{\alpha}(\exp(-\alpha T_d) - \exp(-\alpha T_s))$$

In case of fast sampling $\exp(-\alpha T_s) \approx 1 - \alpha T_s$ and

$$a_1 \approx -(1 - \alpha T_s), \quad b_0 \approx \beta T_d, \quad b_1 \approx \beta T_c$$

This clearly shows that the system dynamics is encoded in the small difference between a_1 and 1 and that for a short sampling period a_1 must be numerically represented with a high relative accuracy in order to distinguish different first order systems.

To obtain the Delta form of the same model rewrite it in the following way

$$y(t) - y(t-1) + (a_1 + 1)\, y(t-1) = b_0(u(t) - u(t-1)) + \\ + (b_0 + b_1)\, u(t-1)$$

and equivalently

$$\Delta y(t) + a_1^* y(t-1) = b_0^* \Delta u(t) + b_1^* u(t-1)$$

where

$$a_1^* = a_1 + 1, \quad b_0^* = b_0, \quad b_1^* = b_0 + b_1$$

For fast sampling the parameters of the Delta model are

$$a_1^* \approx \alpha T_s, \quad b_0^* \approx \beta T_d, \quad b_1^* \approx \beta T_s.$$

In contrast to a_1 the relative accuracy of a_1^* in case of fast sampling does not need to be high. Note also that b_1^*/a_1^* is the static gain of the system.

To proceed towards higher-order and stochastic cases consider the following equality

$$\sum_{i=0}^{n} a_i\, y(t-i) = \sum_{i=0}^{n} a_i^*\, \Delta^{n-i} y(t-i) \tag{43}$$

It is important to note that the lower is the order of the difference on the right-hand side of (43) the more it is shifted backwards in time. Only under this condition the equality (43) can be fulfilled for any parameter values and the mapping between $\{a_i : i = 1..n\}$ and $\{a_i^* : i = 1..n\}$ is one-to-one. To establish this mapping it is convenient to employ the algebra of operator polynomials. However, due to the important time shift it is not possible to express the right-hand side of (43) using a polynomial in operator Δ. Therefore it is suitable to introduce the forward-differ-

ence operator.

$$\delta f(t) = f(t+1) - f(t), \quad \delta^i f(t) = \delta^{i-1}(f(t+1) - \delta^{i-1} f(t) \tag{44}$$

$$\delta = \zeta^{-1} - 1, \quad \Delta = 1 - \zeta = \zeta\delta \tag{45}$$

and to rewrite the right-hand side of (43) as follows

$$\sum_{i=0}^{n} a_i^* \, \Delta^{n-i} y(t-i) = \zeta^n \Big(\sum_{i=0}^{n} a_i^* \, \delta^{n-i} \Big) \, y(t) .$$

Hence, instead of (43) we now can consider

$$\sum_{i=0}^{n} a_i \, \zeta^{-(n-i)} = \sum_{i=0}^{n} a_i^* \, \delta^{n-i} \tag{46}$$

Substitution $\zeta^{-1} = \delta + 1$ gives the identity

$$\sum_{i=0}^{n} a_i (\delta + 1)^{n-i} = \sum_{i=0}^{n} a_i^* \, \delta^{n-i}$$

which determines $\{a_i^* : i = 0..n\}$ for given $\{a_i : i = 0..n\}$

$$a_i^* = \sum_{j=0}^{i} B_n(i,j) \, a_j \tag{47}$$

where $B_n(i,j)$ are the binomial coefficients

$$B_n(i,j) = \binom{n-j}{i-j} \tag{48}$$

which can be easily calculated using the recursion

$$B_n(i,j) = B_n(i+1, j+1) + B_n(i, j+1), \quad i > j \tag{49}$$

which starts with $B_n(k,k) = 1$ and $B_n(n,k) = 1$ for $k = 0, \ldots, n$.

Similarly the substitution $\delta = \zeta^{-1} - 1$ into the right-hand side of (46) determines the inverse transformation

$$a_i = \sum_{j=0}^{i} B_n^*(i,j) \, a_j^*, \quad B_n^*(i,j) = (-1)^{i-j} \binom{n-j}{i-j} \tag{50}$$

$$B_n^*(i,j) = B_n^*(i+1, j+1) - B_n^*(i, j+1), \quad i > j \tag{51}$$

starting with $B_n^*(k,k) = 1$ and $B_n^*(n,k) = (-1)^{n-k}$ for $k = 0, \ldots, n$. For example, for $n = 4$ the transformation matrices are

$$B_4 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 4 & 1 & 0 & 0 & 0 \\ 6 & 3 & 1 & 0 & 0 \\ 4 & 3 & 2 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}, \quad B_4^* = B_4^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ -4 & 1 & 0 & 0 & 0 \\ 6 & -3 & 1 & 0 & 0 \\ -4 & 3 & -2 & 1 & 0 \\ 1 & -1 & 1 & -1 & 1 \end{bmatrix}$$

From the way how the transformations (47) and (50) have been derived it is evident that in multivariate cases each matrix position of the model parameters is transformed

separately. Note that $a_0^* = a_0$ and remains to be a monic LT-matrix. Of course, the c-parameters are again scalars and $c_0^* = c_0 = 1$. Note also that the coefficients of the transformations are integers which can be calculated exactly with no rounding errors involved. However, in case of fast sampling, subtraction of numbers of very different orders can be critical namely in (47). PASCAL procedures performing these transformations will be listed below.

In this way any ARMA model can be recalculated into a Delta form. Using for brevity the operator notation

$$p^*(\delta) = \sum_{i=0}^{n} p_{n-i}^* \delta^i \tag{52}$$

where n is the order of the corresponding backwards-shift operator polynomial $p(\zeta)$, we obtain: (Note the time shifts ensuring exact matching.)

– Positional Delta model replacing the ARMA form (37)

$$a^*(\delta)\, y(t-n) = b^*(\delta)\, u(t - T_u - n) + d^*(\delta)\, v(t - T_v - n) + \\ + c^*(\delta)\, e(t - n_c) + k_c \tag{53}$$

where

$$k_c = c_{n_c}^* k_y \tag{54}$$

or its modification according to (42)

$$a^*(\delta)\, \Delta y(t-n) = \\ b^*(\delta)\, \Delta u(t - T_u - n) + d^*(\delta)\, \Delta v(t - T_v - n) + \delta c^*(\delta)\, e(t - n_c - 1) \tag{55}$$

– Incremental Delta model replacing the ARMA form (38)

$$a^*(\delta)\, \Delta y(t-n) = \\ = b^*(\delta)\, \Delta u(t - T_u - n) + d^*(\delta)\, \Delta v(t - T_v - n) + c^*(\delta)\, e(t - n_c) \tag{56}$$

Extension of the δ-polynomial c

If the random variable $e(t)$ is interpreted as the unpredictable part of $y(t)$ in the sense of (11) for $t \to \infty$ then it is natural and suitable (but not necessary) to rearrange the Delta model so that the highest difference of $e(t)$ be the same as that of $y(t)$ (or of $\Delta y(t)$ in incremental models). This extension of the polynomial $c^*(\delta)$ can be done in the following way.

From the definition of the δ-operator (45) it is easily seen that

$$\zeta(\delta + 1) = 1$$

Hence

$$c^*(\delta)\, e(t - n_c) = \zeta^{n-n_c}(\delta + 1)^{n-n_c}\, c^*(\delta)\, e(t - n_c) = \\ = (\delta + 1)^{n-n_c}\, c^*(\delta)\, e(t - n) \tag{57}$$

Applying this extension (see PASCAL procedure EXTEND below) we can assume from now on that all δ-polynomials in a Delta model are of the same order n. The

same effect is achieved if the polynomial $c(\zeta)$ in the ARMA form of the model is extended to order n by adding zero term

$$c(\zeta) = 1 + c_1\zeta + \ldots + c_{n_c}\zeta^{n_c} + 0\,\zeta^{n_c+1} + \ldots + 0\,\zeta^{n} \tag{58}$$

and then transformed, using (47) or PASCAL procedure ARMAtoDELTA listed below, into a δ-polynomial.

Remark (*b*). The extension of the *c*-polynomial is, actually, not necessary. However, if it is not done beforehand it will be performed automatically by the algorithm for the evolution of the predictive c.p.d.f. $\mathrm{p}(y(t) \mid t-1; u(t))$ which will be derived in Section 4.

PASCAL procedures

The following three PASCAL procedures operate on a polynomial, maximally of order (CONST) *n*max, represented by the variable parameter *p* (VAR) of TYPE *poly* = ARRAY [0.. *n*max] OF REAL and perform the transformation indicated by the procedure identifier. The result is returned in the place of the original polynomial.

```
PROCEDURE ARMAtoDELTA (VAR p:poly; {of order} n:INTEGER);
VAR i, j :INTEGER;
    B :ARRAY [0..nmax, 0..nmax] OF INTEGER;
BEGIN
  FOR i := 0 TO n DO BEGIN B[i, i] := 1; B[n, i] := 1 END;
  FOR j := n - 2 DOWNTO 0 DO
    FOR i := j + 1 TO n - 1 DO B[i, j] := B[i + 1, j + 1] + B[i, j + 1];
  FOR i := n DOWNTO 0 DO
    FOR j := 0 TO i - 1 DO p[i] := p[i] + B[i, j] *p[j]
END;

PROCEDURE DELTAtoARMA (VAR p:poly; {of order} n:INTEGER);
VAR i, j :INTEGER;
      B :ARRAY [0..nmax, 0..nmax] OF INTEGER;
BEGIN
  B[n, n] := 1;
  FOR i := n - 1 DOWNTO 0 DO BEGIN B[i, i] := 1;
                                   B[n, i] := -B[n, i + 1] END;
  FOR j := n - 2 DOWNTO 0 DO
    FOR i := j + 1 TO n - 1 DO B[i, j] := B[i + 1, j + 1] - B[i, j + 1];
  FOR i = nDOWNTO 0 DO
    FOR j := 0 TO i - 1 DO p[i] := p[i] + B[i, j] *p[j]
END;
```

```
PROCEDURE EXTEND (VAR p:poly; {of order} np, {to order} n:INTEGER);
VAR i, j, jl, ju, Dn :INTEGER;
      B :ARRAY [0..nmax] OF INTEGER;
      sum :REAL;
BEGIN
  Dn := n - np;
  FOR i := 0 TO Dn DO B[i] := 1;
  FOR i := 1 TO Dn - 1 DO
    FOR j := i DOWNTO 1 DO B[j] := B[j] + B[j - 1];
  FOR i := n DOWNTO 0 DO
  BEGIN
    sum := 0;
    jl := i - Dn; IF jl < 0 THEN jl := 0;
    ju := i; IF ju > np THEN ju := np;
    FOR j := jl TO ju DO sum := sum + p[j] *B[i - j];
    p[i] := sum
  END
END;
```

3.4. Canonical state models

In this paragraph both ARMA and Delta models will be transformed into canonical state forms. The main reason for this representation of input-output models is that it makes the algorithms for control, prediction and system identification compact and uniform for both univariate and multivariate cases and, moreover, saves the computer storage as well as the number of arithmetic operations required.

State representation of ARMA models

Consider the positional ARMA model (32) with c-part extended, for uniformity, to order n by adding zero terms according to (58). The model can be written in the following way

$$a_0\, y(t) = b_0\, u(t - T_u) + d_0\, v(t - T_v) + e(t) + s_1(t - 1) \tag{59}$$

where we denoted

$$s_1(t-1) = \sum_{j=1}^{n} [-a_j\, y(t-j) + b_j\, u(t - T_u - j) + d_j\, v(t - T_v - j) + c_j\, e(t-j)] + k_c$$

Shifting the time index forwards by one the relation for $s_1(t)$ can be expressed as follows

$$a_1\, y(t) + s_1(t) = b_1\, u(t - T_u) + d_1\, v(t - T_v) + c_1\, e(t) + s_2(t - 1)$$

where

$$s_2(t-1) = \sum_{j=2}^{n} [-a_j\, y(t+1-j) + b_j\, u(t+1-T_u-j) + \\ + d_j\, v(t+1-T_v-j) + c_j\, e(t+1-j)] + k_c$$

Continuing in this way we obtain for $i < n$

$$a_i\, y(t) + s_i(t) = b_i\, u(t-T_u) + d_i\, v(t-T_v) + c_i\, e(t) + s_{i+1}(t-1) \tag{60}$$

$$s_i(t) = \sum_{j=i}^{n} [-a_j\, y(t+i-j) + b_j\, u(t-T_u+i-j) + \\ + d_j\, v(t-T_v+i-j) + c_j\, e(t+i-j)] + k_c \tag{61}$$

and finally for $i = n$

$$a_n\, y(t) + s_n(t) = b_n\, u(t-T_u) + d_n\, v(t-T_v) + c_n\, e(t) + k_c \tag{62}$$

Summing up it is seen that the system of equations (59), (60) for $i = 1..n-1$, and (62) can be written in the following matrix form which will be called the canonical state model.

$$A\, x(t) = b\, u(t-T_u) + d\, v(t-T_v) + c\, e(t) + H\, s(t-1) + k_x \tag{63}$$

where

$$x'(t) = [y'(t), s'(t)], \quad s'(t) = [s_1'(t), s_2'(t), \ldots, s_n'(t)] \tag{64}$$

$$H = \begin{bmatrix} I & 0 & 0 & . & 0 \\ 0 & I & 0 & . & 0 \\ 0 & 0 & I & . & 0 \\ . & . & . & . & . \\ 0 & 0 & 0 & . & I \\ 0 & 0 & 0 & . & 0 \end{bmatrix} \tag{65}$$

$$A = \begin{bmatrix} a_0 & & & & & \\ a_1 & I & & & & \\ a_2 & & I & & & \\ & & & I & & \\ & & & & I & \\ a_{n-1} & & & & & I \\ a_n & & & & & \end{bmatrix}, \quad b = \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ . \\ . \\ b_{n-1} \\ b_n \end{bmatrix}, \quad d = \begin{bmatrix} d_0 \\ d_1 \\ d_2 \\ . \\ . \\ d_{n-1} \\ d_n \end{bmatrix}, \quad k_x = \begin{bmatrix} 0 \\ 0 \\ 0 \\ . \\ . \\ 0 \\ k_c \end{bmatrix} \tag{66}$$

and, since in multivariate case the parameters c_i remain scalars

$$c = \begin{bmatrix} I \\ c_1 I \\ c_2 I \\ . \\ . \\ c_n I \end{bmatrix} \tag{67}$$

where I is a unit matrix of dimension ∂y. Recall that in multivariate case a_0 is a monic LT-matrix. Consequently, also the matrix A, where only the structurally nonzero elements are indicated, is a monic LT-matrix of dimension $(n+1)\,\partial y$. The a-parameters of the model are placed in the first ∂y columns of this matrix.

For zero time delays T_u and T_v the model (63) defines the conditional mean value

$$\mathrm{E}[x(t) \mid v(t), u(t), s(t-1)] = A^{-1}(b\,u(t) + d\,v(t) + H\,s(t-1) + k_x) \tag{68}$$

and the conditional covariance matrix (singular)

$$\mathrm{Var}\,[x(t) \mid v(t), u(t), s(t-1)] = A^{-1} c D_e\, c'(A^{-1})' \tag{69}$$

If, in addition, $e(t)$ is assumed to be normal then the entire c.p.d.f. $\mathrm{p}(x(t) \mid v(t), u(t), s(t-1)) = \mathrm{p}(y(t), s(t) \mid v(t), u(t), s(t-1))$ (18) is defined. Hence, $s(t)$ is a state of the model. If the delays T_u and/or T_v are present then the state must be extended by $\{u(t-i): i = 1..T_u\}$ and $\{v(t-i): i = 1..T_v\}$. However, this part of the total state can be handled separately and when we shall speak about a state we shall mean just $s(t)$.

In the same way the canonical state form can be obtained for the incremental ARMA model.

$$A\,x(t) = b\,\Delta u(t - T_u) + d\,\Delta v(t - T_v) + H\,s(t-1) + c\,e(t) \tag{70}$$

where now the absolute term k_x is missing and

$$x'(t) = [\Delta y'(t), s'(t)] \tag{71}$$

Recall that this form can represent also the positional ARMA model if a unit root is inserted into the polynomial $c(\zeta)$.

State representation of Delta models

Consider the positional Delta model (53) but expressed in backward differences.

$$\sum_{i=0}^{n} a_i\,\Delta^{n-i} y(t-i) = \tag{71}$$

$$= \sum_{i=0}^{n} [b_i\,\Delta^{n-i} u(t - T_u - i) + d_i\,\Delta^{n-i} v(t - T_v - i) + c_i\,\Delta^{n-i} e(t-i)] + k_c$$

To simplify writting we omit the stars distinguishing the parameters of ARMA and of Delta models as no confusion can occur any more. However, the reader has to baer in his mind that they are different. We also assume that the c-part of the model has been extended to order n (see Remark (c) below).

In order to transform (71) into a canonical state model it is suitable to introduce the operation which is inversed with respect to the backward difference Δ. Clearly

$$\Delta f(t) = f(t) - f(t-1)$$
$$f(t) = \Delta f(t) + f(t-1)$$

and recursively

$$f(t) = \Delta f(t) + \Delta f(t-1) + f(t-2) = \sum_{k=0}^{t-1} \Delta f(t-i) + f(0)$$

where $f(0)$ is an arbitrary constant.
Hence, it is possible to write

$$\Delta^{-1} f(t) = \sum_{k=0}^{t-1} f(t-k) + \Delta^{-1} f(0) \tag{73}$$

$$\Delta^{-i} f(t) = \sum_{k=0}^{t-1} \Delta^{-(i-1)} f(t-k) + \Delta^{-i} f(0)$$

Application of Δ^{-n} to the relation (71) gives

$$\sum_{i=0}^{n} a_i \Delta^{-i} y(t-i) = \tag{74}$$

$$= \sum_{i=0}^{n} \Delta^{-i} [b_i u(t - T_u - i) + d_i v(t - T_v - i) + c_i e(t-i)] + f_0(t)$$

where $f_0(t)$ is a function reflecting, according to (73), the effect of posibly nonzero initial conditions. For our purpose it is important to know about this function that its nth difference is

$$\Delta^n f_0(t) = k_c, \quad t > 0$$

The relation (74) can be written similarly to (59)

$$a_0 y(t) = b_0 u(t - T_u) + d_0 v(t - T_v) + e(t) + s_1(t-1) \tag{75}$$

where

$$s_1(t-1) = \sum_{j=1}^{n} \Delta^{-j} [-a_j y(t-j) + b_j u(t - T_u - j) +$$

$$+ d_j v(t - T_v - j) + c_j e(t-j)] + f_0(t)$$

Shifting the time index by one ahead and taking the difference we have

$$\Delta s_1(t) = s_1(t) - s_1(t-1) =$$

$$= \sum_{j=1}^{n} \Delta^{-j+1} [-a_j y(t+1-j) + b_j u(t - T_u + 1 - j) + d_j v(t - T_v + 1 - j) +$$

$$+ c_j e(t+1-j)] + \Delta f_0(t+1)$$

$$a_1 y(t) + \Delta s_1(t) = b_1 u(t - T_u) + d_1 v(t - T_v) + c_1 e(t) + s_2(t-1)$$

This shows that it is possible to continue similarly to ARMA case if in each step also the difference is applied. In this way it is obtained for $i < n$

$$a_i y(t) + \Delta s_i(t) = b_i u(t - T_u) + d_i v(t - T_v) + c_i e(t) + s_{i+1}(t-1) \tag{76}$$

and for $i = n$

$$a_n y(t) + \Delta s_n(t) = b_n u(t - T_u) + d_n v(t - T_v) + c_n e(t) + k_c \tag{77}$$

The matrix form of the canonical state model (75), (76), (77) can be left the same

as (63)

$$A\,x(t) = b\,u(t - T_u) + d\,v(t - T_v) + c\,e(t) + H\,s(t - 1) + k_x \tag{78}$$

with matrices introduced by (65), (66), and (67) but in this positional Delta case

$$x'(t) = [y'(t), \Delta s'(t)] \tag{79}$$

In the same way the canonical state representation of an incremental Delta model can be obtained in the form (70)

$$A\,x(t) = b\,\Delta u(t - T_u) + d\,\Delta v(t - T_v) + c\,e(t) + H\,s(t - 1) \tag{80}$$

but with

$$x'(t) = [\Delta y'(t), \Delta s'(t)] \tag{81}$$

As discussed before, by this incremental model also the positional Delta model can be represented if instead of the original polynomial $c(\delta)$ of order n_c the polynomial $\delta c(\delta)$ is considered and then extended to order n according to (57).

Remark (c). If the extension of the c-part of the Delta model has not been done beforehand (see Remark (b)) then the leading coefficients $\{c_i: i = 0..n - n_c - 1\}$ in (72) must be set to zero, $c_{n-nc} = 1$, and the time index of e should be shifted accordingly, i.e. by $n - n_c$ steps ahead. In a state representation this means that the first $n - n_c$ elements of the matrix c (67) are zero and $e(t + n - n_c)$ should stay instead of $e(t)$. Note that $e(t + n - n_c)$ will influence the output only after $n - n_c$ steps, i.e. it will meet the output $y(t + n - n_c)$ with the same time index. However, if $e(t)$ is interpreted, or simulated on a digital computer, as an external driving white noise then its time shift is insignificant.

3.5. Digital simulation of continuous stochastic processes

Digital simulation of continuous stochastic processes is not the main topic of this paper. However, it is believed that an interested reader might appreciate a simple tool which makes it possible for him to experiment with the presented algorithms and to investigate their sensitivity with respect to violation of theoretical assumptions on which they are based. The purpose of this paragraph is to show that Delta models can serve this purpose, and to give a simple and fast PASCAL function performing this task.

To avoid the theory of stochastic differential equations of higher order consider first the ordinary differential equation

$$\frac{\mathrm{d}^n y}{\mathrm{d}\tau^n} + \sum_{i=1}^{n} \alpha_i \frac{\mathrm{d}^{n-i} y}{\mathrm{d}\tau^{n-i}} = \sum_{i=0}^{n} \beta_i \frac{\mathrm{d}^{n-i} u}{\mathrm{d}\tau^{n-i}} \tag{82}$$

When simulating a continuously operating system described by such an equation on a digital computer it is necessary to admit some kind of approximate discretization. Since our interest is to simulate the evolution of the output y in natural time, i.e.

in the direction of positive τ, it is suitable and convenient to consider the following discretization

$$\frac{\Delta^n y(t)}{(\Delta\tau)^n} + \sum_{i=1}^{n} \alpha_i \frac{\Delta^{n-i} y(t-i)}{(\Delta\tau)^{n-i}} = \sum_{i=0}^{n} \beta_i \frac{\Delta^{n-i} u(t-i)}{(\Delta\tau)^{n-i}} \tag{83}$$

where $\Delta\tau$ is a sufficiently small increment of time τ. The time shift of lower order differences guarantees that $y(t)$ appears in the relation only once and that, given $\{y(t-i): i = 1..n\}$ and $\{u(t-i): i = 0..n\}$, $y(t)$ is determined uniquely for any values of the coefficients. For $\Delta\tau \to 0$ this time shift disappears and in the limit the relation (83) approaches the original differential equation (82). Multiplying the relation (83) by $(\Delta\tau)^n > 0$ the following Delta model is obtained.

$$\sum_{i=0}^{n} a_i \Delta^{n-i} y(t-i) = \sum_{i=0}^{n} b_i \Delta^{n-i} u(t-i)$$

where

$$a_i = (\Delta\tau)^i \alpha_i\,, \quad b_i = (\Delta\tau)^i \beta_i \tag{84}$$

Now consider the state-space representation of a single-output Delta model of positional type (78)

$$A \begin{bmatrix} y(t) \\ \Delta s(t) \end{bmatrix} = b\,u(t) + c\,e(t) + H\,s(t-1) \tag{85}$$

It is easy to verify that

$$A^{-1} = \begin{bmatrix} 1 & & & & \\ a_1 & 1 & & & \\ a_2 & & 1 & & \\ . & & & 1 & \\ . & & & & 1 \\ a_n & & & & \end{bmatrix}^{-1} = \begin{bmatrix} 1 & & & & \\ -a_1 & 1 & & & \\ -a_2 & & 1 & & \\ . & & & 1 & \\ . & & & & 1 \\ -a_n & & & & \end{bmatrix} \tag{86}$$

Applying this inverse

$$\begin{bmatrix} y(t) \\ \Delta s(t) \end{bmatrix} = A^{-1} H\,s(t-1) + A^{-1} b\,u(t) + A^{-1} c\,e(t)$$

the canonical state model can be written as the state equation

$$\Delta s(t) = A_s\,s(t-1) + b_s\,u(t) + c_s\,e(t) \tag{87}$$

and the output equation

$$y(t) = s_1(t-1) + b_0\,u(t) + e(t) \tag{88}$$

where

$$A_s = \begin{bmatrix} -a_1 & 1 & & \\ -a_2 & & 1 & \\ . & & & 1 \\ . & & & \\ -a_n & & & 0 \end{bmatrix}, \quad b_s = \begin{bmatrix} b_1 - a_1 b_0 \\ b_2 - a_2 b_0 \\ . \\ . \\ b_n - a_n b_0 \end{bmatrix}, \quad c_s = \begin{bmatrix} c_1 - a_1 \\ c_2 - a_2 \\ . \\ . \\ c_n - a_n \end{bmatrix} \tag{89}$$

The ith row in (87) is

$$\Delta s_i(t) = -a_i\, s_1(t-1) + s_{i+1}(t-1) + (b_i - a_i b_0)\, u(t) + (c_i - a_i)\, e(t) \tag{90}$$

If the state components are rescaled so that

$$\sigma_i(t) = s_i(t)/(\Delta\tau)^{i-1}\,, \quad \sigma_1(t) = s_1(t) \tag{91}$$

and if the parameters are expressed according to (84)

$$a_i = \alpha_i(\Delta\tau)^i\,, \quad b_i = \beta_i(\Delta\tau)^i\,, \quad c_i = \gamma_i(\Delta\tau)^i \tag{92}$$

then we have instead of (90)

$$\frac{\Delta\sigma_i(t)}{\Delta\tau} = -\alpha_i\,\sigma_1(t-1) + \sigma_{i+1}(t-1) + (\beta_i - \alpha_i\beta_0)\, u(t) + (\gamma_i - \alpha_i)\, e(t)$$

and the state-space model (87)–(88) now is

$$\frac{\Delta\sigma(t)}{\Delta\tau} = \Phi\,\sigma(t-1) + \Psi\, u(t) + \Gamma\, e(t) \tag{93}$$

$$y(t) = \sigma_1(t-1) + \beta_0\, u(t) + e(t) \tag{94}$$

where

$$\Phi = \begin{bmatrix} -\alpha_1 & 1 & & \\ -\alpha_2 & & 1 & \\ . & & & 1 \\ . & & & \\ -\alpha_n & & & 0 \end{bmatrix}, \quad \Psi = \begin{bmatrix} \beta_1 - \alpha_1\beta_0 \\ \beta_2 - \alpha_2\beta_0 \\ . \\ . \\ \beta_n - \alpha_n\beta_0 \end{bmatrix}, \quad \Gamma = \begin{bmatrix} \gamma_1 - \alpha_1 \\ \gamma_2 - \alpha_2 \\ . \\ . \\ \gamma_n - \alpha_n \end{bmatrix} \tag{95}$$

Comparing (93) and (94) with the continuous-time stochastic model (in the innovations representation [11])

$$\mathrm{d}\sigma/\mathrm{d}\tau = \Phi\,\sigma(\tau) + \Psi\, u(\tau) + \Gamma\, e(\tau) \tag{96}$$

$$y(\tau) = \sigma_1(\tau) + \beta_0\, u(\tau) + e(\tau) \tag{97}$$

it is seen that the Delta model (85) can be used to approximate the continuous-time model if the continuous time τ is scaled so that its chosen increment $\Delta\tau$ (small but finite) is a time unit. This scaling of time appears suitable both algoritmically and numerically.

In a similar way it is possible to show that the incremental Delta model (80) can be used to simulate the continuous-time process described by the model [11]

$$\mathrm{d}\sigma = \Phi\sigma\,\mathrm{d}\tau + \Psi\,\mathrm{d}u + \Gamma\,\mathrm{d}w \tag{98}$$

$$\mathrm{d}y = \sigma_1\,\mathrm{d}\tau + \beta_0\,\mathrm{d}u + \mathrm{d}w \tag{99}$$

where $\mathrm{d}w = e\,\mathrm{d}t$ is the increment of the Wiener process.

PASCAL procedure and function

The following PASCAL procedure CONTbyDELTA recalculates, according to (92), the parameters of a continuous-time linear process model to the parameters of the Delta model by which it can be approximated. The function PROCESS generates the process output performing one step of its digital simulation. The nonstandard types of their parameters are

```
TYPE
poly = ARRAY [0..nmax] OF REAL;
system = RECORD
          n : INTEGER {order} ;
          a, b, c, s : poly {parameters and state}
          END;

PROCEDURE CONTbyDELTA (VAR p:poly; n:INTEGER; dt:REAL);
VAR i, j:INTEGER;
BEGIN
  FOR i := 1 TO n DO
    FOR j := i TO n DO p[j] := p[j] * dt
END;

FUNCTION PROCESS (VAR S: system; u, e:REAL): REAL;
VAR i:INTEGER;
BEGIN
  WITH S DO
  BEGIN
    s[0] := s[1] + b[0] * u + c[0] * e;
    FOR i := 1 TO n - 1 DO s[i] := s[i] + s[i + 1] - a[i] * s[0] + b[i] * u +
                                                              + c[i] * e;
    s[n] := s[n] - a[n] * s[0] + b[n] * u + c[n] * e;
    PROCESS := s[0]
  END
END;
```

Remark (*d*). The discrete-time process, by which the continuous-time process is approximated, is slightly less stable than the original one. This is due to the fact that the region of stability for the roots of a δ-polynomial is a disk of radius 1 centered on the point $(-1, 0)$ in δ-plane. (The relation (45) maps the region outside the unit circle in ζ-plane on the above disk in δ-plane.) In the original continuous-time scale this disk has radius $1/\Delta\tau$ and the center on the point $(-1/\Delta\tau, 0)$, see Fig. 3. This means that stable continuous-time systems with poles of their Laplace transfer function $\beta(s)/\alpha(s)$ lying between the circle and the vertical axis in Fig. 3 are simulated

as unstable. To see that this defect is, as a rule, well negligible the reader is recommended to simulate the system described by the differential equation $d^2y/d\tau^2 + (2\pi)^2 y = (2\pi)^2 u$ which, for zero initial state and $u(\tau) = 1$ for $\tau > 0$, should

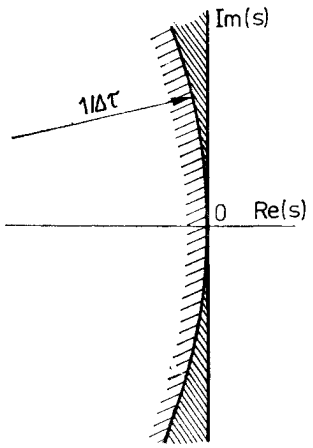

Fig. 3. Stability region in complex s-plane for Delta approximation of continuous systems.

produce the output $y(\tau) = 1 - \cos \tau$. Choosing $\Delta\tau = 0{\cdot}001$ the approximating Delta model has the parameters according to (84):

$$a_0 = 1\,, \quad a_1 = 0\,, \quad a_2 = (2\pi)^2 \,.\, 10^{-6}\,, \quad b_0 = 0\,, \quad b_1 = 0\,, \quad b_2 = (2\pi)^2 \,.\, 10^{-6}$$

The instability of the discrete model starts to be visible on the computer screen only after about 4 periods of the output (after 4000 integration steps realized by the function PROCESS). Otherwise, both the amplitude and the phase of the output are simulated very precisely.

REFERENCES

[11] K. J. Åström: Introduction to Stochastic Control Theory. Academic Press, New York, 1970.

[12] G. E. P. Box and G. M. Jenkins: Time Series Analysis, Forecasting and Control. Holden-Day, San Francisco, Cambridge, London, Amsterdam 1970.

4. STATE ESTIMATION AND OUTPUT PREDICTION

In order to be able to control a given process in a rational way it is necessary to be able to predict its future motion. If the process is described by a state model then the prediction requires to propagate the conditional probability distribution $p(s(t) \mid t)$ by which the uncertainty of the model state $s(t)$ is described. This can be seen from the formula (3.22).

In this section the solution of the classical problem of state estimation and output

prediction will be revised with emphasis on algorithmic and numerical aspects. To provide a general view on the problem the first paragraph is devoted to its conceptual solution in terms of conditional probability distributions. Such a solution does not yield practical algorithms but facilitates a deeper rooted understanding of the problem. In the succeeding paragraphs the conceptual solution is specialized for the linear models introduced in Section 3.

In the second paragraph it will be shown that for canonical state forms of input-output models it is possible to derive algorithms which are simpler and numerically more robust than the standard Kalman filter. They are uniform for both ARMA and Delta models and can solve also continuous cases with good approximation. To make the main ideas of the algorithmic solution as transparent as possible the single-output case is considered first.

The kernel of the presented solution is the algorithm for propagation of the state covariance matrix which replaces the Riccati equation of the Kalman filter. This algorithm and the corresponding PASCAL procedure are discussed in detail in the third paragraph.

In the fourth paragraph it is shown that the effectivity and numerical reliability of the algorithmic solution can be well maintained also in multi-output case.

In the last paragraph the prediction of the process output and of the model state for more than one steps ahead is considered. It will appear useful in control synthesis for processes with time delay.

In the given problem the external measurable disturbance v does not need to be considered explicitly. It can be regarded as an additional process input and as such can be easily incorporated into the final results. The fact that this external input cannot be manipulated is not essential here. However, it is essential that $v(t)$ as well as $u(t)$ influence the model state $s(t)$ but, under natural conditions of control (see paragraphs 2.1 and 3.1), they alone do not bring a new piece of information about $s(t-1)$ in addition to the information contained in the past input-output history. Recall that $s(t-1)$ is the state of the controlled process itself, not of the generator of $v(t)$.

Throughout this section it is assumed that the parameters of the model are known. Since this condition is permanent for all probability distributions involved it is not explicitly stated. The simultaneous estimation of the model parameters and of the state will be considered in Section 6.

4.1. Conceptual solution

Suppose that the inputs $u(k)$ and the outputs $y(k)$ of a particular process, possibly multivariate, have been observed for $k = 1, 2, \ldots, t-1$ and that the c.p.d.f. $\mathsf{p}(s(t-1) \mid t-1)$ has been determined. The problem is: Given the state space model defining the c.p.d.f. (3.18) $\mathsf{p}(y(t), s(t) \mid s(t-1), u(t))$ find the predictive

c.p.d.f. for the next output $p(y(t) \mid t-1; u(t))$ and, after the output is observed, determine $p(s(t) \mid t)$ to prepare the next step of the recursion.

The solution of the given problem can be decomposed into three stages. In the first stage the joint probability distribution for $y(t)$ and $s(t)$, given $u(t)$ and the past input-output history, is determined. Employing just the elementary operations with c.p.d.f.'s (2.1) and (2.2) it is possible to write

$$p(y(t), s(t) \mid t-1; u(t)) = \int p(y(t), s(t), s(t-1) \mid t-1; u(t))\, ds(t-1) =$$

$$= \int p(y(t), s(t) \mid t-1; s(t-1), u(t))\, p(s(t-1) \mid t-1; u(t))\, ds(t-1)$$

From the definition (3.18) of the state we have

$$p(y(t), s(t) \mid t-1; s(t-1), u(t)) = p(y(t), s(t) \mid s(t-1), u(t))$$

and under natural conditions of control it also holds according to (3.19)

$$p(s(t-1) \mid t-1; u(t)) = p(s(t-1) \mid t-1)$$

Hence

(1)
$$p(y(t), s(t) \mid t-1; u(t)) =$$

$$= \int p(y(t), s(t) \mid s(t-1), u(t))\, p(s(t-1) \mid t-1)\, ds(t-1)$$

In the second stage the marginal distribution for the prediction of the process output is determined.

(2)
$$p(y(t) \mid t-1; u(t)) = \int p(y(t), s(t) \mid t-1; u(t))\, ds(t)$$

In the third stage the recursion is concluded by conditioning the probability distribution for the state $s(t)$ with respect to the newly observed output $y(t)$.

(3)
$$p(s(t) \mid t) = \frac{p(y(t), s(t) \mid t-1; u(t))}{p(y(t) \mid t-1; u(t))}$$

Thus the problem of state estimation and output prediction is conceptually solved.

The inspection of the relations (1) to (3) shows that if the model defining $p(y(t), s(t) \mid s(t-1), u(t))$ is linear and normal, and if $p(s(t-1) \mid t-1)$ is assumed to be normal then the predictive c.p.d.f.'s (1) and (2) are normal and also $p(s(t) \mid t)$, updated according to (3), is normal, i.e. the normality is reproduced. This means that for a linear normal state-space model it is sufficient to express the prior uncertainty of the initial state $s(0)$ by a normal c.p.d.f. $p(s(0))$ and the problem can be solved considering only first and second moments (expected values and covariances). This observation is of fundamental importance for further applications of this conceptual solution.

4.2. Algorithmic solution for canonical state representation of linear normal input-output models

In order to cover all models discussed in Section 3 at the same time and by the same algorithms it is suitable to rearrange slightely the state representation of Delta models. The relation (3.76) for the ith component of the canonical state can be rewritten in the following way

$$a_i\, y(t) + s_i(t) = s_i(t-1) + s_{i+1}(t-1) + b_i\, u(t-T_u) + c_i\, e(t) \tag{4}$$

where the term with $v(t)$ has been omitted for brevity. (As discussed above this simplification does not restrict the generality.) Recall that in multi-output case each component of the state $s_i(t)$ is a vector of dimension ∂y. Comparing (4) with (3.60) it is easily seen that both Delta and ARMA models can be described by the relation

$$A\begin{bmatrix} y(t) \\ s(t) \end{bmatrix} = H\, s(t-1) + b\, u(t-T_u) + k_x + c\, e(t) \tag{5}$$

if the matrix H of dimensions $(n+1)\,\partial y \times n\,\partial y$ introduced by (3.65) is redefined in the following way

$$H = \begin{bmatrix} I \\ 0 \end{bmatrix} + \mu \begin{bmatrix} 0 \\ I \end{bmatrix} \tag{6}$$

where I is a unit matrix of dimension $n\,\partial y$ and μ is the model-type indicator

$$\begin{aligned} \mu &= 1 \quad \text{for Delta models}, \\ \mu &= 0 \quad \text{for ARMA models}. \end{aligned} \tag{7}$$

The matrix coefficients A, b, c in (5) and the possible absolute term k_x have the same structure (3.66–67) for both cases but, as discussed in detail in paragraph 3.3, their parameter entries are, in general, different.

By comparing (5) with (3.70–71) and with (3·80–81) it is seen that if $y(t)$ and $u(t-T_u)$ are replaced by their increments $\Delta y(t)$ and $\Delta u(t-T_u)$, respectively, then (5) with $k_x = 0$ covers also incremental models. As discussed in connection with equations (3.70) and (3.80) by incremental form also positional models can be represented if the corresponding root (1 in ARMA case and 0 in Delta case) is inserted into the c-polynomial. If also the Remark (3c), concerning the c-part of a Delta model, is recalled then we are prepared to solve our problem with rather broad generality. However, in order not to hide the main ideas of the algorithmic solution in technicalities the single-output case will be considered first.

Single-output case

The white-noise component $e(t)$ of linear input-output models introduced in Section 3 has the same dimension as the output $y(t)$. Thus in single-output case the diagonal covariance matrix D_e (3.40) is reduced to a nonnegative scalar which will be denoted here as ϱ.

$$\mathrm{Var}\,[e(t)] = \varrho \tag{8}$$

Suppose that the c.p.d.f. $p(s(t-1) \mid t-1)$ is normal with the mean values $\hat{s}(t-1 \mid t-1)$ and with the covariance matrix factorized as follows

$$\text{Var}\,[s(t-1) \mid t-1] = \varrho\, L_s(t-1)\, D_s(t-1)\, L_s'(t-1) \tag{9}$$

where $L_s(t-1)$ is a monic LT-matrix and $D_s(t-1)$ is a diagonal matrix with nonnegative diagonal entries. These factors are supposed to be given numerically. Note that the scalar ϱ is extracted in (9) and does not need to be given numerically.

Taking the expectation conditioned on the observed input-output data up to $t-1$ and on $u(t)$ the model (5) yields

$$A\begin{bmatrix}\hat{y}(t \mid t-1; u(t)) \\ \hat{s}(t \mid t-1; u(t))\end{bmatrix} = H\,\hat{s}(t-1 \mid t-1) + b\,u(t-T_u) + k_x \tag{10}$$

The first row in (10) determines the expected value (the prediction) of the output

$$\hat{y}(t \mid t-1; u(t)) = \hat{s}_1(t-1 \mid t-1) + b_0\,u(t-T_u) \tag{11}$$

and the remaining rows in (10) determine the prediction of the state

$$\hat{s}_i(t \mid t-1; u(t)) = -a_i\,\hat{y}(t \mid t-1; u(t)) + \mu\,\hat{s}_i(t-1 \mid t-1) + \\ + \hat{s}_{i+1}(t-1 \mid t-1) + b_i\,u(t-T_u), \quad i < n \tag{12}$$

$$\hat{s}_n(t \mid t-1; u(t)) = -a_n\,\hat{y}(t \mid t-1; u(t)) + \mu\,\hat{s}_n(t-1 \mid t-1) + \\ + b_n\,u(t-T_u) + k_c \tag{12'}$$

From (5) and (10) we also have

$$A\begin{bmatrix}y(t) - \hat{y}(t \mid t-1; u(t)) \\ s(t) - \hat{s}(t \mid t-1; u(t))\end{bmatrix} = H(s(t-1) - \hat{s}(t-1 \mid t-1)) + c\,e(t)$$

Hence the joint covariance matrix is

$$\text{Var}\begin{bmatrix}y(t) \\ s(t)\end{bmatrix} \Big| t-1; u(t)\Big] = \tag{13}$$

$$= \varrho A^{-1}[H\,L_s(t-1)\,D_s(t-1)\,L_s'(t-1)\,H' + cc'](A^{-1})' =$$

$$= \varrho A^{-1}[c,\, H\,L_s]\begin{bmatrix}1 & 0 \\ 0 & D_s\end{bmatrix}\begin{bmatrix}c' \\ L_s'H'\end{bmatrix}(A^{-1})'$$

where the time argument has been omitted for brevity

The mean values (11) and (12) and the covariance matrix (13) determine the normal joint probability distribution (1). The remaining stages (2) and (3) of the general recursion can be accomplished at the same time by applying the Result (2A). For this purpose it is sufficient to modify the factorization of the joint covariance matrix (13) so that it gets the form

$$\text{Var}\begin{bmatrix}y(t) \\ s(t)\end{bmatrix} \Big| t-1; u(t)\Big] = \varrho L\tilde{D}L' \tag{14}$$

where L is a monic LT-matrix and D is diagonal. Since, according to (3.86), A^{-1} is already a monic LT-matrix

$$A^{-1} = \begin{bmatrix} 1 & 0 \\ -\bar{a} & I \end{bmatrix}, \quad \bar{a}' = [a_1, a_2, \ldots, a_n] \tag{15}$$

it suffices to modify only the inner matrix product on the right-hand side of (13).

$$[c, H L_s] \begin{bmatrix} 1 & 0 \\ 0 & D_s \end{bmatrix} \begin{bmatrix} c' \\ L_s' H' \end{bmatrix} = \tilde{L} \tilde{D} \tilde{L}' \tag{16}$$

Since the left-hand side of (16) can be considered as a weignted sum of dyads with the weights given by the diagonal nonnegative entries of the inner diagonal matrix, the modification can be performed by a suitable application of dyadic reduction. The algorithm performing this task will be described in detail in the following paragraph 4.3. At this place let us assume that the modification has been performed and let us partition the resulting monic LT-matrix $\tilde{L}$ and the diagonal matrix $\tilde{D}$ in the following way

$$\tilde{L} = \begin{bmatrix} 1 & 0 \\ \tilde{c} & \tilde{L}_s \end{bmatrix}, \quad \tilde{D} = \begin{bmatrix} \tilde{d}_y & 0 \\ 0 & \tilde{D}_s \end{bmatrix} \tag{17}$$

where $\tilde{c}$ is a column vector of dimension n, L_s is a monic LT-matrix, and $\tilde{d}_y$ is, in this single-output case, a nonnegative scalar number.

After substituting the right-hand side of (16) into (13) it is seen that the monic LT-matrix L in (14) is

$$L = A^{-1}\tilde{L} = \begin{bmatrix} 1 & 0 \\ -\bar{a} & I \end{bmatrix} \begin{bmatrix} 1 & 0 \\ \tilde{c} & \tilde{L}_s \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ \tilde{c} - \bar{a} & \tilde{L}_s \end{bmatrix} \tag{18}$$

The application of the Result (2A) is now straightforward. Restoring the time argument

$$\tilde{c}(t) = \tilde{c}, \quad L_s(t) = \tilde{L}_s, \quad D_s(t) = \tilde{D}_s, \quad d_y(t) = \tilde{d}_y \tag{19}$$

we obtain

$$\hat{s}(t \mid t) = \hat{s}(t \mid t-1; u(t)) + (\tilde{c}(t) - \bar{a})(y(t) - \hat{y}(t \mid t-1; u(t))) \tag{20}$$

$$\mathrm{Var}[s(t) \mid t] = \varrho L_s(t) D_s(t) L_s'(t) \tag{21}$$

$$\mathrm{Var}[y(t) \mid t-1; u(t)] = \varrho d_y(t) \tag{22}$$

This concludes the recursion.

It is important to note that the multiplication by A^{-1} in (18) does not influence $\tilde{L}_s = L_s(t)$. This means that the evolution of the state covariance matrix (21), of the prediction variance (22), and of the time varying component $\tilde{c}(t)$ of the Kalman gain $(\tilde{c}(t) - \bar{a})$ is determined solely by the parameters c of the model, by the observation time t, and by the initial state covariance matrix which us used to characterize the prior uncertainty of the state $s(0)$ before any data are observed. No other parameters neither data enter this evolution.

Instead of calculating the estimate (the conditional mean) of the state $s(t)$ according to (11), (12), and (20) it is more suitable to proceed as follows. For the ith component

of the state the relation (20) gives

$$\hat{s}_i(t \mid t) = \hat{s}_i(t \mid t-1; u(t)) + (\tilde{c}_i(t) - a_i)\,(y(t) - \hat{y}(t \mid t-1; u(t)))$$

When $\hat{s}_i(t \mid t-1; u(\,))$ is substituted from (12) the term $a_i\,\hat{y}(t \mid t-1; u(t))$ is cancelled. Making use also of (11) it is finally obtained

$$\hat{s}_i(t \mid t) = \mu\,\hat{s}_i(t-1 \mid t-1) + \hat{s}_{i+1}(t-1 \mid t-1) - \tag{23}$$
$$- \tilde{c}_i(t)\,\hat{s}_1(t-1 \mid t-1) - (a_i - \tilde{c}_i(t))\,y(t) + (b_i - \tilde{c}_i(t)\,b_0)\,u(t - T_u)\,,$$
$$i < n$$

$$\hat{s}_n(t \mid t) = \mu\,\hat{s}_n(t-1 \mid t-1) - \tilde{c}_n(t)\,s_1(t-1 \mid t-1) - \tag{23'}$$
$$- (a_n - \tilde{c}_n(t))\,y(t) + (b_n - \tilde{c}_n(t)\,b_0)\,u(t - T_u) + k_c$$

Summing up we come to the following

Result (4A): State estimation in single-output case.

If the white-noise component $e(t)$ of the canonical state-space model (5) is normally distributed with zero mean and with the variance ϱ (8), and if the probability distribution $p(s(0))$ describing the prior uncertainty of the initial state $s(0)$ is also normal then the c.p.d.f. $p(s(t) \mid t)$ is normal for all $t > 0$ and the evolution of its mean value is governed by the difference equation

$$\hat{s}(t \mid t) = (\mu I + C(t))\,\hat{s}(t-1 \mid t-1) - (\bar{a} - \tilde{c}(t))\,y(t) + \tag{24}$$
$$+ (\bar{b} - \tilde{c}(t)\,b_0)\,u(t - T_u) + k_s$$

where

$$C(t) = \begin{bmatrix} \tilde{c}_1(t) & 1 & 0 & 0 & . & 0 \\ \tilde{c}_2(t) & 0 & 1 & 0 & . & 0 \\ \tilde{c}_3(t) & 0 & 0 & 1 & . & 0 \\ . & . & . & . & . & . \\ \tilde{c}_{n-1}(t) & 0 & 0 & 0 & . & 1 \\ \tilde{c}_n(t) & 0 & 0 & 0 & . & 0 \end{bmatrix} \tag{25}$$

$$\tilde{c}'(t) = [\tilde{c}_1(t), \tilde{c}_2(t), \ldots, \tilde{c}_n(t)] \tag{26}$$

$$\bar{a}' = [a_1, a_2, \ldots, a_n]\,, \quad \bar{b}' = [b_1, b_2, \ldots, b_n] \tag{27}$$

$$k_s' = [0, 0, \ldots, 0, k_c] \tag{28}$$

The coefficients $\tilde{c}_i(t)$, in general time varying, as well as the factors $L_s(t)$ and $D_s(t)$ of the state covariance matrix (21), and the variance of the output prediction (22) can be propagated simultaneously by the modification of the matrix factorization (16) which, with partitioning (17) and time indexing (19), reads

$$[c, H\,L_s(t-1)] \begin{bmatrix} 1 & 0 \\ 0 & D_s(t-1) \end{bmatrix} \begin{bmatrix} c' \\ L_s'(t-1)\,H' \end{bmatrix} = \tag{29}$$
$$= \begin{bmatrix} 1 & 0 \\ \tilde{c}(t) & L_s(t) \end{bmatrix} \begin{bmatrix} d_y(t) & 0 \\ 0 & D_s(t) \end{bmatrix} \begin{bmatrix} 1 & \tilde{c}'(t) \\ 0 & L_s'(t) \end{bmatrix}$$

The one-step-ahead prediction of the output $\hat{y}(t \mid t-1; u(t))$ is given by (11) and the prediction of the state $\hat{s}(t \mid t-1; u(t))$, if required, can be calculated according to (12).

Remark (*a*). Note that the variance ϱ (8) of the white-noise component of the model does not enter the evolution of conditional means and of time-varying factors of covariances. Therefore it does not need to be known if the prior covariance matrix of the initial state $s(0)$ is chosen in proportion to this possibly unknown parameter.

$$\text{Var}\,[s(0)] = \varrho\, L_s(0)\, D_s(0)\, L_s'(0) \tag{30}$$

Remark (*b*). The vector difference equation (24) clearly shows that the dynamics of the filter generating the state estimate $\hat{s}(t \mid t)$ is given by the matrix $(\mu I + C(t))$ and by the evolution of the coefficients $\tilde{c}(t)$ (26) which determine this matrix. To get a deeper insight it is advantageous to consider the stochastic difference equation by which the difference between the true but unknown state and its estimate

$$\tilde{s}(t) = s(t) - \hat{s}(t \mid t) \tag{35}$$

is governed. By simple algebraic manipulation with the model (5) and the difference equation (24) of the estimation filter it is possible to derive

$$\tilde{s}(t) = (\mu I + C(t))\, \tilde{s}(t-1) + (\bar{c} - \tilde{c}(t))\, e(t) \tag{36}$$

where

$$\bar{c}' = [c_1, c_2, \ldots, c_n] \tag{37}$$

Two important observations can be drawn from this stochastic difference equation:

(i) Suppose that the algorithm performing the modification (29), converges for $t \to \infty$ producing $\tilde{c}(\infty) = \tilde{c}$, $L_s(\infty) = L_s$, $D_s(\infty) = D_s$, and $C(\infty) = C$. Since the estimation of the state is optimal (in the sense that it extracts all relevant information about the state contained in the observed data) the stochastic difference equation (36) cannot be unstable for $t \to \infty$. This means that none of the roots $\{\lambda_i: i = 1, \ldots, n\}$ of the characteristic equation $\det((\lambda - \mu) I - C) = 0$ can lie outside the unit circle. The characteristic equation with the matrix C of structure (25) reads

$$(\lambda - \mu)^n + \sum_{i=1}^{n} \tilde{c}_i (\lambda - \mu)^{n-i} = 0$$

Hence, in ARMA case ($\mu = 0$) the polynomial

$$\tilde{c}(\lambda) = \lambda^n + \sum_{i=1}^{n} \tilde{c}_i \lambda^{n-i}$$

has no roots outside the unit circle even when the polynomial $c(\zeta)$ of the ARMA model is unstable. Similarly, in Delta case ($\mu = 1$) the polynomial

$$\tilde{c}(\delta) = \delta^n + \sum_{i=1}^{n} \tilde{c}_i\, \delta^{n-i}$$

has no roots outside the circle of radius 1 centered on the point $(-1, 0)$ in δ-plane.

(ii) If the c-polynomial of the model is stable and $\tilde{c} = \bar{c}$ then, according to (36), $\tilde{s}(t)$ converges to zero. Asymptotically, the state of the model can be reconstructed exactly and the state covariance matrix converges to a zero matrix, $D_s(\infty) = 0$.

These observations will be supported by examples in the following paragraph where the algorithm performing the modification (29) will be described in detail.

4.3. Algorithm for propagating the state covariance matrix and generating $\tilde{c}(t)$

An algorithm will be designed which performs the modification of the matrix factorization (16) and in this way, according to (29), propagates the factors $L_s(t)$ and $D_s(t)$ of the state covariance matrix (21), and simultaneously generates $\tilde{c}(t)$ (26) and the factor $d_y(t)$ of the prediction variance (22). This algorithm replaces, for the canonical state model (5), the Riccati equation of the standard Kalman filter. Unlike the Riccati equation it guarantees numerically the nonnegative definitness of the propagated covariance matrix. This is of extraordinary importance for practical computation particularly in often met cases when the state covariance matrix converges to a zero matrix (see Remark (b)). The basic idea of the algorithm rests on the fact that the left-hand side of (16) can be considered as a sum of weighted dyads and therefore can be modified using the dyadic reduction.

To design the algorithm it is sufficient to consider only the rows of the right-hand factor on the left-hand side of (16) and their weights as shown in the scheme Fig. 4 where each row represents one dayd and its weight. Note that the last row,

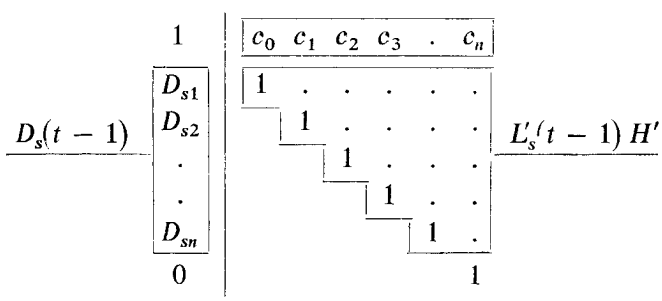

Fig. 4. First stage of the algorithm CGEN.

containing just a single nonzero entry (one), is added but the zero weight is assigned to it. This trick makes the algorithm compact. In most cases $c_0 = 1$, but as discussed in Remarks (b) and (c) in Section 3, it can also be $c_0 = 0$ if the c-polynomial of a Delta model is not extended to the full order n.

The goal is to perform the modification (16) so that the right-hand factor depicted in Fig. 4 be a monic upper triangular matrix $\tilde{L}'$. This can be achieved when the first row $c_0, c_1, \ldots, c_n$ is totally zeroed by sequential application of the dyadic reduction. The row having the weight D_{s1} is used to zero the first entry c_0, the next row is used

to zero the second entry (c_1 modified by the first step), and so on until the situation

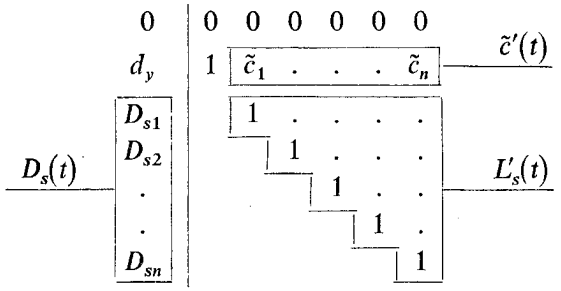

Fig. 5. Final stage of the algorithm CGEN.

depicted in Fig. 5 is reached. Note that $\tilde{c}'$ appears in the second row and that both $L'_s(t)$ and $D_s(t)$ are shifted by one row lower. This must be respected when coding the algorithm.

PASCAL PROCEDURE CGEN

The above described algorithm is realized by the following procedure *CGEN*. The nonstandard types of its parameters are

TYPE *row* = ARRAY [0..nmax] OF REAL;
matrix = ARRAY [0..nmax] OF *row*;

where nmax is a suitably chosen integer constant. Since *CGEN* makes use of the procedure *DYDR* (see paragraph 2.2) the type *row* must be the same for the both procedures.

Parameters:

L ... type *matrix*; factor L'_s of the state covariance matrix to be updated, $L[i, j] = L_{s,ji}$; must be initialized including $L[i, i] = 1$; the part under the main diagonal not used; *L* is extended by row $L[0]$ which can be arbitrary when the procedure is called.

D ... type *row*; factor D_s (its diagonal elements) of the updated state covariance matrix, $D[i] = D_{s,i}$; $D[0]$ can be arbitrary when the procedure is called.

c ... type *row*; parameters *c* of the model, $c[i] = c_i$ $(i = 0, \ldots, n)$.

n ... type INTEGER; model order.

mti ... type INTEGER; model-type indicator (in the text denoted by μ); set *mti* = 0 for ARMA, *mti* = 1 for Delta model.

The generated $\tilde{c}$ appears in the row $L[0]$, $\tilde{c}_i = L[0, i]$; d_y appears in $D[0]$,

```
PROCEDURE CGEN (VAR L:matrix; VAR D:row; c:row; n,mti. INTEGER);
VAR i, j: INTEGER;
  De:REAL;
BEGIN
De := 1;
FOR i := 1 TO n DO
```

```
  BEGIN
  FOR j := i + 1 TO n DO L[i − 1, j − 1] := L[i, j] + mti * L[i, j − 1];
  L[i − 1, n] := mti * L[i, n];
  D[i − 1] := D[i]
  END;
D[n] := 0;
FOR i := 0 TO n DO DYDR(c, L[i], De, D[i], i, i + 1, n)
END;
```

Example (4.1). For a first-order process model ($n = 1, L = 1$) simple formulae for the evolution of the scalars $D_s(t)$ and $\tilde{c}_1(t)$ can be derived. They can help to understand what the algorithm actually does. Going through the algorithm step by step it is found that for $c_0 = 1$

$$D_s(t) = \frac{D_s(t-1)}{D_s(t-1)+1}(c_1 - \mu)^2 \tag{38}$$

$$\tilde{c}_1(t) = \mu + \frac{c_1 - \mu}{D_s(t-1)+1} \tag{39}$$

$$d_y(t) = D_s(t-1) + 1 \tag{40}$$

If, for $D_s(0) > 0$, $D_s^{-1}(t)$ is considered as the dependent variable then the difference equation (38) is transformed into the following linear difference equation of first order.

$$D_s^{-1}(t) = (c_1 - \mu)^{-2} D_s^{-1}(t-1) + (c_1 - \mu)^{-2}$$

Hence, for $(c_1 - \mu)^2 \neq 1$ we have

$$D_s^{-1}(t) = D_s^{-1}(0)(c_1 - \mu)^{-2t} + ((c_1 - \mu)^2 - 1)^{-1} \tag{41}$$

and for the interesting special case when the c-polynomial has its root at the stability boundary, $(c_1 - \mu)^2 = 1$

$$D_s^{-1}(t) = D_s^{-1}(0) + t \tag{42}$$

The explicit solutions (41) or (42) determine $\tilde{c}_1(t)$ and $d_y(t)$ for any $t > 0$ according to (39) and (40). Of particular interest are the values to which the algorithm converges for growing t. From (41) and (42) it is seen that
for $(c_1 - \mu)^2 \leqq 1$

$$D_s^{-1}(\infty) = \infty\,, \quad D_s(\infty) = 0\,, \quad \tilde{c}_1(\infty) = c_1\,, \quad d_y(\infty) = 1 \tag{43}$$

while for $(c_1 - \mu)^2 > 1$

$$D_s(\infty) = (c_1 - \mu)^2 - 1\,, \quad \tilde{c}_1(\infty) - \mu = (c_1 - \mu)^{-1}\,, \quad d_y(\infty) = (c_1 - \mu)^2 \tag{44}$$

Hence, for $t \to \infty$ the algorithm returns the polynomial $\tilde{c}$ which is always stable. Note also that according to (39) for all finite t

$$|\tilde{c}_1(t) - \mu| < |c_1 - \mu|$$

For an ARMA model ($\mu = 0$) with $c_1 = -1$ the relations (39) and (42) give for $\mu = 0$, $c_1 = -1$

$$\tilde{c}_1(t) = -1 + \frac{1}{D_s^{-1}(0) + t}$$

which clearly shows that $\tilde{c}_1(t)$ is approaching its asymptotic value $\tilde{c}_1(\infty) = -1$ from the stable side. If such a first order model is transformed into the Delta form ($\mu = 1$) then $c_1 = 0$ and

for $\mu = 1$, $c_1 = 0$

$$\tilde{c}_1(t) = \frac{1}{D_s^{-1}(0) + t}$$

which again shows that also in the Delta case the stability boundary (the pure summation in the state estimator (24)) is approached in a cautious way.

Example (4.2). To demonstrate that also in higher order cases, which cannot be analysed so easily, the algorithm behaves in a similar way let us consider an ARMA model ($\mu = 0$) of order $n = 3$ with the polynomial $c(\zeta)$ having two roots outside the stability region

$$c(\zeta) = 1 - 3{\cdot}35\zeta + 2{\cdot}825\zeta^2 - 0{\cdot}25\zeta^3 = (1 - 1{\cdot}25\zeta)(1 - 2\zeta)(1 - 0{\cdot}1\zeta)$$

Since all roots lie rather far from the stability boundary the procedure *CGEN* reaches the stationary solution very fast. With $D(0) = [1, 1, 1]$ and $L(0) = I$ only after about 30 steps (for $t = 30$) it is obtained with precision of 4 significant digits

$$\tilde{c}(\zeta) = 1 - 1{\cdot}400\zeta + 0{\cdot}5300\zeta^2 - 0{\cdot}0400\zeta^3 = (1 - 0{\cdot}8\zeta)(1 - 0{\cdot}5\zeta)(1 - 0{\cdot}1\zeta)$$

Thus the unstable roots are reflected into the stability region. However, the prediction variance is $d_y(\infty) = 6{\cdot}250$ times larger than the variance of the white-noise component e of the model.

If the ARMA model is tranformed into the Delta form (using, for instance, the procedure *ARMAtoDELTA* listed in Section 3) the equivalent δ-polynomial c is

$$c(\delta) = \delta^3 - 0{\cdot}35\,\delta^2 - 0{\cdot}875\,\delta + 0{\cdot}225$$

and the procedure *CGEN* for $mti = \mu = 1$ returns with a similar speed

$$c(\delta) = \delta^3 - 1{\cdot}600\,\delta^2 + 0{\cdot}7300\,\delta + 0{\cdot}09000$$

and the same $d_y(\infty) = 6{\cdot}250$, of course.

Example (4.3). In this example we will demonstrate the fact mentioned in Remarks (b) and (c) in Section 3, namely that the procedure *CGEN* performs the extension of the c-polynomial of a Delta model to the full order if it has not been done beforehand. Consider, for instance, a regression model of order 4. If this model is transfor-

med into the Delta form then the δ-polynomial $c(\delta)$, when not extended, is

$$c(\delta) = 0\,\delta^4 + 0\,\delta^3 + 0\,\delta^2 + 0\,\delta + 1$$

and, as discussed in Remark (c) in Section 3, the vector c in the canonical state model (5) is $c' = [0, 0, 0, 0, 1]$. Applying the procedure *CGEN* the reader can verify that after 4 steps, i.e. for $t \geqq 4$

$$c(\delta) = \delta^4 + 4\delta^3 + 6\delta^2 + 4\delta + 1$$

Note that the matrix $(I + C(t))$ in (36) has, for $t > 4$, four eigenvalues equal to zero.

Example (4.4). It may be interesting to see how the procedure *CGEN* can manage the continuous process model if it is approximated by a Delta model simply replacing $d\tau$ by a small but finite time increment $\Delta\tau$. For this purpose consider a continuous process described by the model

$$y(\tau) = \frac{\beta(s)}{\alpha(s)}\,u(\tau) + \frac{\gamma(s)}{\alpha(s)}\,e(\tau)$$

where $\alpha(s)$, $\beta(s)$, and $\gamma(s)$ are polynomials in the differential operator s, and $e(\tau)$ is white noise. For demonstration let us choose $\gamma(s)$ with one positive (unstable) root.

$$\gamma(s) = s^2 + 0{\cdot}4\,s - 0{\cdot}6 = (s + 1)(s - 0{\cdot}6)$$

First, let us suppose that the order of the model, given by the order of the polynomial $\alpha(s)$, is $n = 2$, i.e. the same as the order of $\gamma(s)$. Apparently, the choice $\Delta\tau = 0{\cdot}005$ sec should give a good discrete approximation of the continuous model. With this choice the c-polynomial of the approximating Delta model is (use formulae (3.92) or the procedure *CONTbyDELTA* from Section 3)

$$c(\delta) = \delta^2 + 2\,.\,10^{-3}\,\delta - 1{\cdot}5\,.\,10^{-5}$$

If the procedure *CGEN* is started with $L(0) = I$ and $D(0) = [1, 1]$ then after 1600 steps (compare $\tau = 1600\,(\Delta\tau) = 8$ sec with the time constants of $\gamma(s)$) the stationarity is approach width

$$c(\delta) = \delta^2 + 7{\cdot}992\,.\,10^{-3}\,\delta + 1{\cdot}495\,.\,10^{-5}, \quad d_y = 1{\cdot}006$$

In the real-time scale this corresponds to (recall formulae (3.92))

$$\tilde{\gamma}(s) = s^2 + 1{\cdot}605\,s + 0{\cdot}6049$$

which well approximates the correct solution

$$\tilde{\gamma}(s) = s^2 + 1{\cdot}6\,s + 0{\cdot}6 = (s + 1)(s + 0{\cdot}6)$$

Even more interesting is the case when the order of $\gamma(s)$ is lower than the model order, i.e. than the order of $\alpha(s)$. To demonstrate such a case let us consider the same $\gamma(s)$ but formally written as the third-order polynomial

$$\gamma(s) = 0\,s^3 + s^2 + 0{\cdot}4\,s - 0{\cdot}6$$

Correspondingly, with the same $\Delta\tau = 0{\cdot}005$ sec, we now have

$$c(\delta) = 0\,\delta^3 + 5\,.\,10^{-3}\,\delta^2 + 1\,.\,10^{-5}\,\delta - 7{\cdot}5\,.\,10^{-8}$$

Starting with $L(0) = I$ and $D(0) = [1, 1, 1]$ the procedure *CGEN* gives for $t = 1600$ ($\tau = 8$ sec)

$$\tilde{c}(\delta) = \delta^3 + 1{\cdot}008\,\delta^2 + 8{\cdot}024\,.\,10^{-3}\,\delta + 1{\cdot}509\,.\,10^{-5}, \quad d_y = 2{\cdot}515\,.\,10^{-5}$$

In order to see what this result means in the real time scale it is suitable to proceed as follows. When inspecting the factorization (29) as modified by the procedure *CGEN* it is seen that the result does not change when d_y is divided by $(\Delta\tau)^2$ and at the same time $c(\delta)$ is multiplied by $(\Delta\tau)$. Hence, the couple $\{\tilde{c}(\delta), d_y\}$ can be equally written as

$$\tilde{c}(\delta) = 5\,.\,10^{-3}\,\delta^3 + 5{\cdot}040\,.\,10^{-3}\,\delta^2 + 4{\cdot}012\,.\,10^{-5}\,\delta + 7{\cdot}545\,.\,10^{-8},$$
$$d_y = 1{\cdot}006$$

Now when the transformation (3.92) back to the continuous time is applied it is obtained

$$\tilde{\gamma}(s) = 0{\cdot}005\,s^3 + 1{\cdot}008\,s^2 + 1{\cdot}6048\,s + 0{\cdot}6036$$

Compare this result with the polynomial obtained by exact reflection

$$\tilde{\gamma}(s) = 0\,s^3 + s^2 + 1{\cdot}6\,s + 0{\cdot}6 = (s + 1)(s + 0{\cdot}6)$$

It should be mentioned that these examples were calculated using the precision of only 4 bytes floating-point arithmetic.

4.4. Multi-output process

When trying to extend the derivation from paragraph 4.2 to multi-output case the first difficulty met is that Var $[e(t)]$ is no more a single number ϱ but a diagonal matrix

$$\text{Var}\,[e(t)] = D_e = \begin{bmatrix} \varrho_1 & 0 & . & 0 \\ 0 & \varrho_2 & . & 0 \\ . & . & . & . \\ 0 & 0 & 0 & \varrho_{\partial y} \end{bmatrix} \tag{45}$$

which cannot be simply extracted from all expressions for covariance matrices. Hence, to be able to proceed further we have to set $\varrho = 1$ in (9) leaving D_e incorporated in the matrix factors L_s and D_s which are now of dimensions $n\,\partial y \times n\,\partial y$. Instead of (11) it is obtained

$$a_0\,\hat{y}(t \mid t-1; u(t)) = \hat{s}_1(t-1 \mid t-1) + b_0\,u(t - T_u) \tag{46}$$

while (12) and (12′) are still valid with the only difference that now the coefficients a_i are matrices of dimensions $\partial y \times \partial y$. Note that the prediction $\hat{y}(t \mid t-1; u(t))$ can be calculated from (46) very easily, without any division, as a_0 is a monic LT-matrix.

Continuing in the revision of the single-output case we arrive to the relation (13) which now reads

$$\text{(47)} \qquad \text{Var}\left[\begin{array}{c|c} y(t) & \\ s'(t) & t-1; u(t) \end{array}\right] = A^{-1}[c, H L_s]\begin{bmatrix} D_e & 0 \\ 0 & D_s \end{bmatrix}\begin{bmatrix} c' \\ L_s' H' \end{bmatrix}(A^{-1})'$$

Also here A^{-1} is a monic LT-matrix

$$\text{(48)} \qquad A^{-1} = \begin{bmatrix} a_0^{-1} & 0 & 0 & . & 0 \\ -a_1 a_0^{-1} & I & 0 & . & 0 \\ -a_2 a_0^{-1} & 0 & I & . & 0 \\ . & & & . & . \\ -a_n a_0^{-1} & 0 & 0 & . & I \end{bmatrix}$$

Similarly to the single-output case, it is sufficient to modify only the inner matrix product in (47) to perform conditioning according to Result (2A).

$$\text{(49)} \qquad [c, H L_s]\begin{bmatrix} D_e & 0 \\ 0 & D_s \end{bmatrix}\begin{bmatrix} c' \\ L_s' H_s' \end{bmatrix} = \tilde{L}\tilde{D}\tilde{L}'$$

Recall that D_e is a diagonal matrix and that, according to (3.67), c' is a block-row with all matrix entries of diagonal form

$$c' = [c_0 I, c_1 I, c_2 I, \ldots, c_n I]$$

where all c_i are scalars.

The following observation is of primary importance for simple and well feasible numerical solution of multivariate cases. The matrix $L_s' = L_s'(t-1)$ is a monic UT-matrix of dimensions $n\,\partial y \times n\,\partial y$ which can be partitioned into $n \times n$ matrix entries of dimensions $\partial y \times \partial y$. Only the matrix entries on and above the main diagonal are, in general, nonzero. Suppose that all these nonzero matrix entries have the diagonal form. To be able to express ourselves shortly and clearly we shall say that such a matrix has an internally diagonal structure. If L_s' with the internally diagonal structure is multiplied by the matrix H' (6) then the structure of the product $L_s' H'$ is also internally diagonal. Thus all factors on the left-hand side of (46), which are to be modified, have the internally diagonal structure. When inspecting the algorithm of dyadic reduction which is used to perform the modification it is found that it preserves this structure. The matrix $\tilde{L}$, when partitioned similarly to (17)

$$\text{(50)} \qquad \tilde{L} = \begin{bmatrix} I & 0 \\ \tilde{c} & \tilde{L}_s \end{bmatrix}, \quad \tilde{D} = \begin{bmatrix} \tilde{D}_y & 0 \\ 0 & \tilde{D}_s \end{bmatrix}$$

yields $\tilde{L}_s = L_s(t)$ and the internally diagonal structure is reproduced. Practically this means that the overall algorithm can be decomposed into ∂y independent algorithms of reduced dimension, and, moreover, if these algorithms are started in the same way then a single algorithm of the same complexity as in single-output case can perform the task, and the submatrices $\tilde{c}$ and D_y in (50) get the forms

$$\text{(51)} \qquad \tilde{c}' = [\tilde{c}_1 I, \tilde{c}_2 I, \ldots, \tilde{c}_n I], \quad \tilde{D}_y = \tilde{d}_y D_e$$

where $\tilde{d}_y$, $\tilde{c}_1, \tilde{c}_2, \ldots, \tilde{c}_n$ are scalars generated by the procedure *CGEN* described in the previous paragraph.

The above favourable fact, discovered by carefull investigation of the algorithmic solution, has the following probabilistic interpretation. In the multi-output case each component of the state $s_i(t)$ has ∂y subcomponents

$$s_i'(t) = [s_{i(1)}(t), s_{i(2)}(t), \ldots, s_{i(\partial y)}(t)], \quad i = 1, 2, \ldots, n \tag{52}$$

Let us rearrange the entire set of n ∂y state subcomponents into ∂y vectors of dimension n in the following way.

$$s'_{(k)}(t) = [s_{1(k)}(t), s_{2(k)}(t), \ldots, s_{n(k)}(t)], \quad k = 1, 2, \ldots, \partial y \tag{53}$$

If the prior uncertainty of the state $s(0)$ is described by a normal probability distribution with

$$\mathrm{Cov}\,[s_{(k)}(0), s_{(j)}(0)] = 0, \quad k \neq j \tag{54}$$

then also for all $t > 0$

$$\mathrm{Cov}\,[s_{(k)}(t), s_{(j)}(t) \mid t] = 0, \quad k \neq j \tag{55}$$

If, in addition, it is assumed

$$\mathrm{Var}\,[s_{(k)}(0)] = \varrho_k L_s(0) D_s(0) L_s'(0), \quad k = 1, 2, \ldots, \partial y \tag{56}$$

then

$$\mathrm{Var}\,[s_{(k)}(t) \mid t] = \varrho_k L_s(t) D_s(t) L_s'(t), \quad k = 1, 2, \ldots, \partial y \tag{57}$$

and

$$\mathrm{Var}\,[a_0 y(t) \mid t - 1; u(t)] = d_y(t) D_e \tag{58}$$

where the monic LT-matrix $L_s(t)$, the diagonal matrix $D_s(t)$, and the scalar $d_y(t)$ are common for all k and can be generated by the procedure *CGEN*. Note that here, and from now on, the matrix factors $L_s(t)$ and $D_s(t)$ are only fo dimensions $n \times n$.

It should be emphasized that this simplification, which means a considerable reduction of the computational load, could be achieved only thanks to the introduction of the model parameter a_0 as a monic LT-matrix (3.41). This made it possible to introduce the white-noise term $e(t)$ of the model as a random vector with uncorrelated components, i.e. with the diagonal covariance matrix (45).

Equipped with this knowledge we can complete the extension for multi-output case in a straightforward way. Continuing in the revision of the single-output case in paragraph 4.2 we come to the generalizing (23)

$$\hat{s}_i(t \mid t) =$$

$$= \mu \hat{s}_i(t-1 \mid t-1) + \hat{s}_{i+1}(t-1 \mid t-1) - \tilde{c}_i(t) \hat{s}_1(t-1 \mid t-1) -$$

$$- (a_i - \tilde{c}_i(t) a_0) y(t) + (b_i - \tilde{c}_i(t) b_0) u(t - T_u), \quad i < n \tag{59}$$

$$\hat{s}_n(t \mid t) = \mu \hat{s}_n(t-1 \mid t-1) - \tilde{c}_n(t) \hat{s}_1(t-1 \mid t-1) -$$

$$- (a_n - \tilde{c}_n(t) a_0) y(t) + (b_n - \tilde{c}_n(t) b_0) u(t - T_u) + k_c \tag{59'}$$

If these relations are considered row-wise then the obtained result can be formulated as follows.

Results (4*B*) State estimation in multi-output case.

Let $s_{(k)}(t)$ be the components of the state introduced by (53). Let $a_{i(k)}$, $b_{i(k)}$, and $k_{c(k)}$ denote the kth rows of the model parameters a_i, b_i, and k_c, respectively, and let

$$\bar{a}_{(k)} = \begin{bmatrix} a_{(k)1} \\ a_{(k)2} \\ . \\ a_{(k)n} \end{bmatrix}, \quad \bar{b}_{(k)} = \begin{bmatrix} b_{(k)1} \\ b_{(k)2} \\ . \\ b_{(k)n} \end{bmatrix}, \quad k_{s(k)} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ k_{c(k)} \end{bmatrix} \tag{60}$$

Assume that

(i) the white-noise term $e(t)$ of the canonical state-space model (5) is normally distributed with zero mean and with the diagonal covariance matrix (45),

(ii) the probability distribution $\mathrm{p}(s(0))$ used to describe the prior uncertainty of the state $s(0)$ is chosen to be normal with uncorrelated components $s_{(k)}(0)$,

(iii) the initial covariance matrices (56) are chosen so that they differ only in the factors ϱ_k.

Then the probability distribution $\mathrm{p}(s(t) \mid t)$ is normal for all $t > 0$, the state components $s_{(k)}(t)$ remain mutually uncorrelated, and their conditional mean values are determined by ∂y separate filters ($k = 1, 2, \ldots, \partial y$) as follows.

$$\begin{aligned} \hat{s}_{(k)}(t \mid t) = (\mu I + C(t))\, \hat{s}_{(k)}(t-1 \mid t-1) - \\ - (\bar{a}_{(k)} - \tilde{c}(t)\, a_{0(k)})\, y(t) + (\bar{b}_{(k)} - \tilde{c}(t)\, b_{0(k)})\, u(t - T_u) + k_{s(k)} \end{aligned} \tag{61}$$

where the matrix $C(t)$ (25) and the vector $\tilde{c}(t)$ (26) are common for all k and are supplied by the procedure *CGEN* together with $L_s(t)$, $D_s(t)$ and $d_y(t)$ by which the variances (57) and (58) are determined. The one-step-ahead prediction of the output and of the state can be calculated recursively according to (46) and (12).

Remark (*c*). Note that, similarly to single-output case, the variances ϱ_k ($k = 1, 2, \ldots, \partial y$) of the mutually uncorrelated components of the discrete white noise $e(t)$ do not need to be known if only the conditional means (the point estimates) are of interest. In Section 6 we shall see that this favourable fact holds also when the model parameters a and b are estimated jointly with the state.

4.5. Prediction in case of process delay

Hitherto only one-step-ahead prediction of the process output $y(t)$ and of the state $s(t)$ of the process model has been considered. Namely, given $\hat{s}(t-1 \mid t-1)$ we are able to determine $\hat{y}(t \mid t-1; u(t))$ and $\hat{s}(t \mid t-1; u(t))$ from the relation (10) which can be solved recursively using (11), or (46) in multi-output case, and (12). If the process has a time delay $T_u > 0$ then the predicted $y(t)$ and $s(t)$ do not depend on

the already applied inputs $u(t-k)$ $k = 1, 2, .. T_u - 1$, but only on $u(t - T_u)$ and the inputs applied previously. However, when controlling such a process it is necessary to be able to predict the effect of the generated input $u(t - T_u)$ on the output $y(t)$ and on the state $s(t)$ only on the basis of the data which are available at the moment when this input is generated. Hence, it is necessary to predict the process by T_u steps more ahead. We shall show two possible ways how to proceed. Since the delay discussed here concerns only the manipulated input the external measurable disturbance v will be considered separately.

The first possibility is to increase the order of the process model by T_u and to shift the b-parameters correspondingly. Then the canonical state form (5) of the model will have no explicit time delay T_u

$$A\begin{bmatrix} y(t) \\ s(t) \end{bmatrix} = H\, s(t-1) + b\, u(t) + d\, v(t - T_v) + k_x + c\, e(t) \tag{62}$$

but the matrix b, now of dimensions $(n + T_u)\,\partial y \times \partial u$ has T_u leading (block-) entries equal to zero.

$$b' = [0, 0, \ldots, 0, b'_0, b'_1, \ldots, b'_n] \tag{63}$$

Now it is possible to proceed as if no delay T_u were present. However, it should be emphasized that in case of a Delta model ($\mu = 1$) the first (block-) column in the matrix A (3.66), i.e. the δ-polynomial $a(\delta)$, as well as the matrix d, the δ-polynomial $d(\delta)$, must be extended to the full order $n + T_u$ as described in Section 3.

The second method does not require the extension of the model state. Suppose that the input-output data up to $t - 1$ are available and that the input $u(t)$ is to be generated. When the time index is shifted by T_u steps ahead and when the mean value conditioned on the data up to $t - 1$ is taken the model equation (5) gives

$$\begin{bmatrix} \hat{y}(t + T_u \mid t - 1; u(t)) \\ \hat{s}(t + T_u \mid t - 1; u(t)) \end{bmatrix} = \tag{64}$$

$$= A^{-1}[H\, \hat{s}(t + T_u - 1 \mid t - 1) + b\, u(t) + d\, \hat{v}(t + T_u - T_v \mid t - 1) + k_x]$$

This relation shows that it is necessary to predict $s(t + T_u - 1)$ and $v(t + T_u - T_v)$ in order to estimate the effect of $u(t)$, which is to be decided, on the future motion of the process. Suppose that the prediction of the external disturbance is available. A model suitable for this purpose will be constructed in Section 5. Now it will be shown how the estimate $\hat{s}(t + T_u - 1 \mid t - 1)$ can be effectively calculated.

For j within the range $0 \leqq j < T_u$ the model equation (5) yields

$$A\begin{bmatrix} \hat{y}(t + j \mid t - 1) \\ \hat{s}(t + j \mid t - 1) \end{bmatrix} = \tag{65}$$

$$= H\hat{s}(t + j - 1 \mid t - 1) + b\, u(t + j - T_u) + d\, \hat{v}(t + j - T_v \mid t - 1) + k_x$$

Hence $\hat{s}(t + T_u - 1 \mid t - 1)$ can be calculated recursively, for $j = 0, 1, \ldots, T_u - 1$,

using the relations

$$a_0\,\hat{y}(t+j\,|\,t-1) = \tag{66}$$

$$= \hat{s}_1(t+j-1\,|\,t-1) + b_0\,u(t+j-T_u) + d_0\,\hat{v}(t+j-T_v\,|\,t-1)$$

$$\hat{s}_i(t+j\,|\,t-1) = -a_i\,\hat{y}(t+j\,|\,t-1) + \mu\,\hat{s}_i(t+j-1\,|\,t-1) + \tag{67}$$

$$+ \hat{s}_{i+1}(t+j-1\,|\,t-1) + b_i\,u(t+j-T_u) + d_i\,\hat{v}(t+j-T_v\,|\,t-1),$$

$$i = 1, \ldots, n-1$$

$$\hat{s}_n(t+j\,|\,t-1) = -a_n\,\hat{y}(t+j\,|\,t-1) + \hat{s}_n(t+j-1\,|\,t-1) + \tag{67'}$$

$$+ b_n\,u(t+j-T_u) + d_n\,\hat{v}(t+j-T_v\,|\,t-1) + k_c$$

Note that all inputs employed are available at the moment when the calculation is required, however, T_u past inputs must be stored in the memory of the computing device. Note also that, as a_0 is a monic LT-matrix, $\hat{y}(t+j\,|\,t-1)$ can be calculated from (66) very easily, also recursively.

5. CONTROL SYNTHESIS

The problem of optimal control can be formulated in different ways depending on the criterion used to measure the control performance and on the set of admissible strategies among which the optimal one is to be chosen. Since the control strategy must be chosen in advance, before it is applied and before its true effect can be observed, the criterion cannot be anything else than a single-valued probabilistic characteristic of the future motion of the process by which an ordering in the set of admissible strategies is introduced. In this section the expected value of a suitably chosen loss function, covering an arbitrary long but finite control horizon, will be considered as the criterion and the strategy minimizing this criterion will be chosen among the strategies which make use of all data available at the moment when the particular input is generated.

In the first paragraph, using dynamic programming in the form suggested in [17], a general functional recursion is derived which solves the problem conceptually. It is shown that this functional recursion can be reduced to a special kind of Bellman equation if certain sufficient statistics exist.

In the following paragraphs the loss function is restricted to be quadratic. Its choice is made in the second paragraph where also suitable models for the evolution of the external signals (the command signal and the external measurable disturbance) are introduced. When compared with the ordinary LQG control theory, as it can be found in standard engineering textbooks [11, 13, 15, 16], the problem formulation applied here is somewhat restricted and at the same time somewhat more general. The restriction concerns the loss function which is allowed to be a function only of data which can be observed on the process. No attempt is made to control internal

process variables which are not accessible to measurement. The reason why we accept such a restriction is that it makes it possible to operate only with input-output process models which can be identified from observed data. It also means that the model state, we make use of to reduce the computational burden, does not need to have physical interpretation. On the other hand the quadratic loss function is chosen to better reflect the engineering needs in industrial process control and to cover a broader class of operating modes of the controller (regulation, servocontrol, program control).

In the algorithmic solution of the optimum control synthesis it is suitable to consider the positional and the incremental forms of process models separately. Since the incremental form can cover also positional models (see Sections 3 and 4) and since the algorithm of control synthesis is simpler and more compact for this form, it will be considered first in the third paragraph.

In the fourth paragraph the algorithm of the optimum control synthesis for positional models is derived.

5.1. Control optimal in the mean — conceptual solution

Suppose that a given process has been observed and possibly somehow controlled up to and including the sampling period $t_0 \geqq 0$. Starting with the input $u(t_0 + 1)$ the process has to be controlled for $t = t_0 + 1, t_0 + 2, \ldots, t_0 + T$ optimally in the sense we are going to define.

In order to make the writting shorter and more transparent it is suitable to shift the time indexing backwards by t_0. We shall write $u(k)$ instead of $u(t_0 + k)$ and similarly for all other signal samples. Thus $u(1)$ is the first input which is to be decided and the observation of the process started for $k = -t_0$.

It is also suitable to introduce the following sets of data:
Data which are not known before the control starts

$$\mathscr{X}_k = \{y(1..k), u(1..k), v(1..k), w(1..k)\} \tag{1}$$

All data up to and including the sampling interval k

$$\mathscr{D}_k = \{y(-t_0..k), u(-t_0..k), v(-t_0..k), w(1..k)\} = \{\mathscr{D}_0, \mathscr{X}_k\} \tag{2}$$

$$\mathscr{D}_k = \{y(k), u(k), v(k), w(k), \mathscr{D}_{k-1}\}, \quad k > 0$$

History of the external disturbance

$$\mathscr{V}_k = v(-t_0..k) = \{v(k), \mathscr{V}_{k-1}\} \tag{3}$$

History of the command signal

$$\mathscr{W}_k = w(1..k) = \{w(k), \mathscr{W}_{k-1}\}, \quad k > 0 \tag{4}$$

Let $l(\mathscr{D}_T)$ be a scalar nonnegative function of the observed data which will be used to evaluate the control performance after the control task is accomplished. It is

supposed that the smaller the true value of this function will be the better is the control performance: $l(\mathscr{D}_T)$ is a loss function. Since the future data entering the loss function are not available when the first input $u(1)$ has to be decided, it is appropriate to design the control strategy using the expected value of the loss function as a criterion

$$J = \mathsf{E}[l(\mathscr{D}_T) \mid \mathscr{D}_0] = \int l(\mathscr{D}_T)\, \mathsf{p}(\mathscr{X}_T \mid \mathscr{D}_0)\, \mathrm{d}\mathscr{X}_T$$

Of course, it is assumed that there exists an admissible control strategy which makes this expectation finite.

To define the admissible control strategies it is important to consider how the command signal is made available for the controller. As discussed in paragraph 1.2, in case of program control the command signal is preprogrammed in advance for the entire control horizon T and the controller can operate also on the future values of this signal. Conceptually this is not a very interesting case as the command signal enters the loss function only as a set of fixed parameters $\mathscr{W}_T$. Here the more general case of servocontrol will be considered. It will be assumed that when $u(k)$ is being decided the command signal is known only up to $w(k)$ (the desired value of the output $y(k)$ following $u(k)$).

It will appear advantageous to consider the following decomposition of the loss function into T nonnegative terms

$$l(\mathscr{D}_T) = l_T(\mathscr{D}_T) + l_{T-1}(\mathscr{D}_{T-1}) + \ldots + l_1(\mathscr{D}_1)$$

Note that such a decomposition is not unique. For instance, we could simply choose $l(\mathscr{D}_T) = l_T(\mathscr{D}_T)$ and $l_k(\mathscr{D}_k) = 0$ for $k < T$. A more suitable choice will be made later on.

The problem of optimum control synthesis can be formulated on the conceptual level as follows: Given the c.p.d.f.'s

$$\mathsf{p}(y(k) \mid k-1; v(k), u(k)),\ \mathsf{p}(v(k) \mid \mathscr{V}_{k-1}),\ \mathsf{p}(w(k) \mid \mathscr{W}_{k-1}),\quad k = 1, 2, \ldots, T$$

determine the c.p.d.f.'s

$$\mathsf{p}(u(k) \mid w(k), \mathscr{D}_{k-1}),\quad k = 1, 2, \ldots, T$$

minimizing the criterion

$$J = \mathsf{E}\Big[\sum_{k=1}^{T} l_k(\mathscr{D}_k) \mid \mathscr{D}_0\Big] = \sum_{k=1}^{T} \int l_k(\mathscr{D}_k)\, \mathsf{p}(\mathscr{X}_k \mid \mathscr{D}_0)\, \mathrm{d}\mathscr{X}_k \tag{5}$$

It is advantageous to solve the problem recursively. Assume that the way of generating the inputs $u(k)$ for $k = 1, \ldots, T-1$ have been somehow determined and that only the control law for the last one $u(T)$ remains to be chosen. To choose it optimally it is necessary to determine the optimal c.p.d.f.

$$\mathsf{p}(u(T) \mid w(T), \mathscr{D}_{T-1}) = \mathsf{p}(u(T) \mid w(T), \mathscr{X}_{T-1}, \mathscr{D}_0) \tag{6}$$

as a function of the data $\mathscr{X}_{T-1}$ in its condition which are not known at the moment

when the choice must be made. To perform this task recall (3.4), (3.5), (3.6), and consider that

$$\mathsf{p}(\mathscr{X}_T \mid \mathscr{D}_0) = \mathsf{p}(y(T), v(T), u(T), w(T) \mid \mathscr{D}_{T-1})\, \mathsf{p}(\mathscr{X}_{T-1} \mid \mathscr{D}_0) =$$
$$= \mathsf{p}(y(T) \mid T-1; v(T), u(T))\, \mathsf{p}(v(T) \mid \mathscr{V}_{T-1})\, \mathsf{p}(u(T) \mid w(T), \mathscr{D}_{T-1})$$
$$\mathsf{p}(w(T) \mid \mathscr{W}_{T-1})\, \mathsf{p}(\mathscr{X}_{T-1} \mid \mathscr{D}_0)$$

This makes it possible to rewrite the last term of the criterion (5) with $k = T$, the only one which depends on the c.p.d.f. (6), in the following way

$$\mathsf{E}[l_T(\mathscr{D}_T) \mid \mathscr{D}_0] = \tag{7}$$
$$= \iiint F_T(u(T), w(T), \mathscr{D}_{T-1})\, \mathsf{p}(u(T) \mid w(T), \mathscr{D}_{T-1})\, \mathrm{d}u(T)$$
$$\mathsf{p}(w(T) \mid \mathscr{W}_{T-1})\, \mathrm{d}w(T)\, \mathsf{p}(\mathscr{X}_{T-1} \mid \mathscr{D}_0)\, \mathrm{d}\mathscr{X}_{T-1}$$

where

$$F_T(u(T), w(T), \mathscr{D}_{T-1}) = \tag{8}$$
$$= \int l_T(\mathscr{D}_T)\, \mathsf{p}(y(t) \mid T-1; v(T), u(T))\, \mathrm{d}y(T)\, \mathsf{p}(v(t) \mid \mathscr{V}_{T-1})\, \mathrm{d}v(T)$$

Since both $\mathsf{p}(w(T) \mid \mathscr{W}_{T-1})$ and $\mathsf{p}(\mathscr{X}_{T-1} \mid \mathscr{D}_0)$ are nonnegative functions normalized so that their integrals are equal to 1, the term (7) will be minimal if we succeed to choose the c.p.d.f. (6) in such a way that the integral

$$\int F_T(u(T), w(T), \mathscr{D}_{T-1})\, \mathsf{p}(u(T) \mid w(T), \mathscr{D}_{T-1})\, \mathrm{d}u(T) \tag{9}$$

be minimal for all possible $w(T)$ and $\mathscr{D}_{T-1}$.

The function $F_T(u(T), w(T), \mathscr{D}_{T-1})$ is nonnegative, and when plotted with respect to $u(T)$ for some arbitrary but fixed $w(T)$ and $\mathscr{D}_{T-1}$ it might look, for instance, like

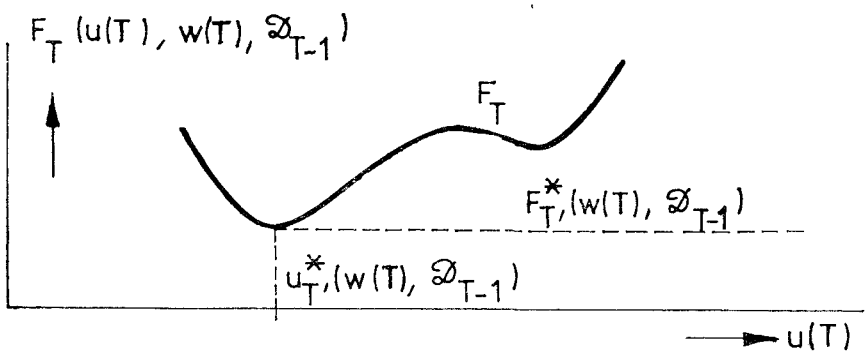

Fig. 6. Illustration to the optimal choice of $\mathsf{p}(u(T)\, w(T), \mathscr{D}_{T-1})$.

the curve F_T in Fig. 6. The problem to determine the c.p.d.f. (6) minimizing the integral (9) is equivalent to the problem how to distribute one unit of the probability mass along the curve F_T in Fig. 6 so that its moment with respect to the $u(T)$-axis

be as small as possible, i.e. so that its centre of gravity be as low as possible. It is obvious that this will be achieved if all the probability mass is placed at the lowest point of the curve F_T. This means that the optimal choice is

$$\mathrm{p}(u(T) \mid w(T), \mathscr{D}_{T-1}) = \delta(u(T) - u_T^*(w(T), \mathscr{D}_{T-1})) \tag{10}$$

where $\delta(\cdot)$ is a Dirac δ-function and

$$u_T^*(w(T), \mathscr{D}_{T-1}) = \arg\min_{u(T)} F_T(u(T), w(T), \mathscr{D}_{T-1}) \tag{11}$$

If the minimum

$$F_T^*(w(T), \mathscr{D}_{T-1}) = \min_{u(T)} F_T(u(T), w(T), \mathscr{D}_{T-1}) \tag{12}$$

can be achieved for more than one $u(T)$ then any of the minimizing arguments (11) can be chosen as optimal. However, the value of the minimum (12) is always unique.

If we denote

$$B_{T-1}(\mathscr{D}_{T-1}) = \int F_T^*(w(T), \mathscr{D}_{T-1})\, \mathrm{p}(w(T) \mid \mathscr{W}_{T-1})\, \mathrm{d}w(T) \tag{13}$$

then the partially minimized criterion (5) can be expressed as follows.

$$\mathrm{E}\Big[B_{T-1}(\mathscr{D}_{T-1}) + l_{T-1}(\mathscr{D}_{T-1}) + \sum_{k=1}^{T-2} l_k(\mathscr{D}_k) \mid \mathscr{D}_0\Big] =$$

$$= \int (B_{T-1}(\mathscr{D}_{T-1}) + l_{T-1}(\mathscr{D}_{T-1}))\, \mathrm{p}(\mathscr{X}_{T-1} \mid \mathscr{D}_0)\, \mathrm{d}\mathscr{X}_{T-1} +$$

$$+ \sum_{k=1}^{T-2} \int l_k(\mathscr{D}_k)\, \mathrm{p}(\mathscr{X}_k \mid \mathscr{D}_0)\, \mathrm{d}\mathscr{X}_k$$

In this way the first step of the optimization procedure for $k = T$ is completed and the next step for $k = T - 1$ is prepared. Denoting

$$F_{T-1}(u(T-1), w(T-1), \mathscr{D}_{T-2}) = \int (B_{T-1}(\mathscr{D}_{T-1}) + l_{T-1}(\mathscr{D}_{T-1}))$$

$$\mathrm{p}(y(T-1) \mid T-2; v(T-1), u(T-1))\, \mathrm{d}y(T-1)\, \mathrm{p}(v(T-1) \mid \mathscr{V}_{T-2})\, \mathrm{d}v(T-1)$$

and proceeding further in the same way the following result is obtained.

Result (5A). Optimum control synthesis.

The control strategy minimizing the criterion (5) is deterministic

$$u(k) = u_k^*(w(k), \mathscr{D}_{k-1}) \tag{14}$$

and is generated by the functional recursion

$$F_k(u(k), w(k), \mathscr{D}_{k-1}) = \tag{15}$$

$$= \iint (B_k(\mathscr{D}_k) + l_k(\mathscr{D}_k))\, \mathrm{p}(y(k) \mid k-1; v(k), u(k))\, \mathrm{d}y(k)\, \mathrm{p}(v(k) \mid \mathscr{V}_{k-1})\, \mathrm{d}v(k)$$

$$u_k^*(w(k), \mathscr{D}_{k-1}) = \arg\min_{u(k)} F_k(u(k), w(k), \mathscr{D}_{k-1}) \tag{16}$$

$$F_k^*(w(k), \mathscr{D}_{k-1}) = \min_{u(k)} F_k(u(k), w(k), \mathscr{D}_{k-1}) \tag{17}$$

$$B_{k-1}(\mathscr{D}_{k-1}) = \int F_k^*(w(k), \mathscr{D}_{k-1})\, \mathrm{p}(w(k) \mid \mathscr{W}_{k-1})\, \mathrm{d}w(k) \tag{18}$$

which has to be solved for $k = T, T-1, \ldots, 1$ with the initial condition $B_T(\mathscr{D}_T) = 0$. The minimum of the criterion is $J^* = B_0(\mathscr{D}_0)$.

Bellman function

Because the subset $\mathscr{X}_{k-1}$ of the set of data $\mathscr{D}_{k-1} = \{\mathscr{X}_{k-1}, \mathscr{D}_0\}$ on which the optimal control strategy (14) operates is not known at the moment when the strategy has to be decided, the optimal control law for each k must be determined as a function of these not yet known data $\mathscr{X}_{k-1}$. This domain of the function $u_k^*(\cdot)$, in general of growing dimension, can be reduced to a finite and fixed dimension if the following conditions are fulfilled.

a) There exist sufficient statistics $S_y(k)$, $S_v(k)$, $S_w(k)$, here understood as deterministic functions, in general multivariate but of fixed dimensions, of observed data and possibly also of the time index k, such that

$$\mathrm{p}(y(k) \mid k-1; v(k), u(k)) = \mathrm{p}(y(k) \mid v(k), u(k), S_y(k-1)) \tag{19}$$

$$\mathrm{p}(v(k) \mid V_{k-1}) = \mathrm{p}(v(k) \mid S_v(k-1)) \tag{20}$$

$$\mathrm{p}(w(k) \mid W_{k-1}) = \mathrm{p}(w(k) \mid S_w(k-1)) \tag{21}$$

b) The evolution of these statistics is known

$$S_y(k) = \mathrm{f}_y(y(k), u(k), v(k), S_y(k-1), k) \tag{22}$$

$$S_v(k) = \mathrm{f}_v(v(k), S_v(k-1), k) \tag{23}$$

$$S_w(k) = \mathrm{f}_w(w(k), S_w(k-1), k) \tag{24}$$

c) It is possible to decompose the loss function so that

$$l(\mathscr{D}_T) = \sum_{k=1}^{T} l_k(y(k), v(k), u(k), w(k), S_l(k-1)) \tag{25}$$

where $S_l(k)$ is a fixed-dimensional function of the data such that

$$S_l(k) = \mathrm{f}_l(y(k), w(k), u(k), S_l(k-1)) \tag{26}$$

Let $S(k)$ be the union

$$S(k) = S_l(k) \cup S_y(k) \cup S_v(k) \cup S_w(k) \tag{27}$$

By inspection of the recursion (15) to (18) it can be easily verified that under these conditions the optimal control law producing $u(k)$ operates on $w(k)$ and on the statistic $S(k-1)$. If we omit, for brevity, the explicit notation of the arguments of the functions

$$l_k = l_k(y(k), w(k), u(k), v(k), S_l(k-1))$$

and

$$B_k = B_k(S(k)) \tag{28}$$

then the optimal control law is

$$u_k^*(w(k), S(k-1)) = \arg\min_{u(k)} \mathsf{E}[B_k + l_k \mid u(k), w(k), S_l(k-1)] \tag{29}$$

where the Bellman function (28) is determined by the difference equation

$$B_{k-1} = \mathsf{E}[\min_{u(k)} \mathsf{E}[B_k + l_k | u(k), w(k), S(k-1)] | S(k-1)] \tag{30}$$

which has to be solved for $k = T, T-1, \ldots, 1$ with the initial condition $B_T = 0$.

Algorithm of dynamic programming

Under given conditions one step of the functional recursion (15) to (18) can be decomposed into the following three stages:

1. Given $B_k + l_k$ take the mean value over $y(k)$ and $v(k)$

$$F_k(u(k), w(k), S(k-1)) = \mathsf{E}[B_k + l_k \mid u(k), w(k), S(k-1)] = \tag{31}$$

$$= \iint (B_k + l_k)\, \mathsf{p}(y(k) \mid v(k), u(k), S_y(k-1))\, \mathrm{d}y(k)\, \mathsf{p}(v(k) \mid S_v(k-1))\, \mathrm{d}v(k)$$

2. Determine the minimum of $F_k(u(k), w(k), S(k-1))$ with respect to $u(k)$

$$F_k^*(w(k), S(k-1)) = \min_{u(k)} F_k(u(k), w(k), S(k-1)) \tag{32}$$

$$u_k^*(w(k), S(k-1)) = \arg\min_{u(k)} F_k(u(k), w(k), S(k-1)) \tag{33}$$

3. Take the mean value over $w(k)$ to obtain the Bellman function for the next step of the recursion

$$B_{k-1} = \mathsf{E}[F_k^*(w(k), S(k-1)) \mid S(k-1)] = \tag{34}$$

$$= \int F_k^*(w(k), S(k-1))\, \mathsf{p}(w(k) \mid S_w(k-1))\, \mathrm{d}w(k)$$

From (15) and (18) or from (31) and (34) it is seen that the optimum control synthesis requires to define suitable models for the evolution of external signals v and w. They will be introduced together with a quadratic loss function in the following paragraph.

5.2. Quadratic criterion and models of external signals

In the rest of this section the following quadratic criterion will be used to measure the expected performance of the controller

$$J = \frac{1}{T} \mathsf{E}[\sum_{k=1}^{T} (|Q_r(k); M_r(k)\, r(k + T_u)| + |Q_u(k); M_k(k)\, \Delta u(k)|) + \tag{35}$$

$$+ \left| Q_s(T);\, M_s(T)\,(\hat{s}(T + T_u \mid T) - \bar{s}) \right| \mid D_0]$$

where $r(k)$ is the control error

$$r(k) = y_c(k) - w(k) \tag{36}$$

$Q_r(k)$ and $Q_u(k)$ are diagonal matrices with nonnegative diagonal entries, $Q_s(T)$ is diagonal with positive diagonal entries, $M_r(k)$, $M_u(k)$ and $M_s(T)$ are monic LT-matrices of appropriate dimensions, and $\bar{s}$ is the desired value of the last state for $k = T + T_u$. T_u is the possible process delay. Note that the first control error which can be influenced by the choice of $u(1)$ is $r(1 + T_u)$.

The first nonnegative quadratic form in the criterion (35) $\left| Q_r(k);\, M_r(k)\, r(k + T_u) \right| = r'(k + T_u)\, M_r'(k)\, Q_r(k)\, M_r(k)\, r(k + T_u)$, $Q_r(k) \geqq 0$, reflects the requirement that the control errors (36) be small. The choice of $Q_u(k) \geqq 0$ and $M_u(k)$ in the second quadratic form of the criterion (35) makes it possible to damp the movements of the actuator(s) if it is required by the given technology and/or implementation. There are seldom reasons to choose the matrices $M_r(k)$ and $M_u(k)$ different from unit matrices.

The last term in (34) with $Q_s > 0$, penalizing the deviation of the estimate of the last state from its desired value, is introduced in order to secure the asymptotic stability of the control loop also for $Q_u(k) = 0$. Note that for $T \to \infty$ and for finite Q_s and $\bar{s}$ this last term is negligible but only when all signals driving the state estimator (4.24) are stable. Large entries of Q_s help to stabilize the control loop also in case of relatively short control horizon T.

To be able to determine the control strategy minimizing the criterion (35) it is necessary to adopt suitable models for the evolution of external signals, i.e. for the measurable external disturbance v (if available) and for the command signal w. These models are required to determine the conditional means in (15) or (31) and in (18) or (34).

Measurable external disturbance

As in most practical industrial cases the measurable external disturbances are nonstationary we shall design the controller so that it may be optimal for a "generalized random walk"

$$v(k) = v(k - 1) + e_v(k) \tag{37}$$

where

$$\mathsf{E}[e_v(k) \mid \mathcal{V}_{k-1}] = \mathsf{E}[e_v(k)] = 0 \tag{38}$$

and the variances of the uncorrelated increments $e_v(k)$, $k = 1, \ldots, T$, can be arbitrarily time-varying but finite and independent of the past history $\mathcal{V}_{k-1}$. The random walk (37) generalized in such a way is a very realistic model for load changes at unpredictable time instants, drifts, etc. Note that such a model does not need to be identified. The practical experience indicates that usually not much can be gained if a more detailed model for the evolution of the measurable external disturbance is considered.

Command signal

As it has been discussed in the paragraph 1.2 the following typical operating modes of the controller will be considered.

Regulation. In industrial process control often the task of the controller is to compensate measurable or unmeasurable stochastic disturbances and to keep certain output variables as close as possible to prescribed constant values. If the output y_c is measured as the deviation from the given fixed setpoint then $y_c(k) = r(k)$ and $w(k) = 0$ for all k.

Program control. In industrial practice cases are met when the desired output of the process is time-varying but a priory preprogrammed for the entire control horizon T. Then the task of the controller is to manipulate the actuators in such a way that the true output y_c of the process follows the command signal w which is a priori known and the controller, when generating $u(k)$ can operate also on the future values of the command signal $w(j)$, $j > k$, stored in the memory of the computer. This prior information often can significantly improve the performance of the controller. For instance, the controller informed about the future change of the command signal starts to manipulate the process well in advance in order to minimize the control errors, especially when its actions are damped by a relatively large weight Q_u in the criterion (35).

Positional servo. If the future course of the command signal is uncertain a probabilistic model has to be employed. We shall consider the case when the changes of the command signal cannot be predicted. Then a suitable model is the above introduced generalized random walk

$$w(k) = w(k-1) + e_w(k) \tag{39}$$

It should be emphasized once more that the variances of the mutually uncorrelated increments $e_w(k)$ can be arbitrarily time-varying but they are assumed to be independent of the past history of the command signal. The prediction of such a process is

$$\mathrm{E}[w(j) \mid \mathscr{W}_k] = w(k), \quad j > k \tag{40}$$

In the following paragraphs algorithms of optimum control synthesis will be designed for the case of program control and for the case of positional servo. The case of regulation is covered either by program control with $w(k) = 0$ for all k or by positional servo with zero variance of $e_w(k)$.

5.3. Algorithm of optimum control synthesis for incremental process models

Consider a process with time delay $T_u \geqq 0$ described by the state model in the incremental form which, with the time index shifted by T_u steps ahead, reads

(41)

$$A\begin{bmatrix}\Delta y(k+T_u)\\ s(k+T_u)\end{bmatrix} = Hs(k-1+T_u) + b\,\Delta u(k) + d\,\Delta v(k+T_u-T_v) + ce(k+T_u)$$

where the matrices A, b, d are defined by (3.66), the matrix c is defined by (3.67), and the matrix H by (4.6)

$$H = \mu\begin{bmatrix}0\\ I\end{bmatrix} + \begin{bmatrix}I\\ 0\end{bmatrix}$$

Recall that for $\mu = 1$ the relation (41) is a canonical state representation of an incremental Delta model while for $\mu = 0$ it represents an incremental ARMA model. As discussed in Section 3 the incremental form (41) can represent also positional models if the c-parameters are suitably modified.

Since in steady state the mean value of the state s of the incremental model (41) is equal to zero it is appropriate to choose $\bar{s} = 0$ in the criterion (35).

In the present paragraph the following Result (5 B) will be proved and the corresponding algorithms will be designed.

Result (5 B): Optimal controller for incremental process models.

Suppose that the controlled process is described by the incremental model (41). If the evolution of the external measurable disturbance is assumed to be a generalized random walk (37) then in case of *program control* the structure of the optimal control law minimizing the criterion (35) is

(42) $$\Delta u(k) = -m_{uw}(k) - m_{ur}(k)\,\bar{r}(k) - m_{us}(k)\,\hat{s}(k-1+T_u \mid k-1)$$

where

(43) $$\bar{r}(k) = \hat{y}_c(k-1+T_u \mid k-1) - w(k+T_u)$$

and $\hat{s}(k-1+T_u \mid k-1)$ is the T_u-steps-ahead prediction of the state of the model (41). The parameters $m_{uw}(k)$, $m_{ur}(k)$ and $m_{us}(k)$ of the optimal control law (42) are vector and matrices of appropriate dimensions which can be determined by the algorithm of dynamic programming with the Bellman function of the nonnegative definite quadratic form

(44) $$B_k = |Q_0(k);\, m_{0w}(k) + m_{0r}(k)\,r(k+T_u \mid k)| + \\ + |Q_s(k);\, m_w(k) + m_r(k)\,r(k+T_u \mid k) + M_s(k)\,\hat{s}(k+T_u \mid k)| + \beta(k)$$

where

(45) $$r(k+T_u \mid k) = \hat{y}_c(k+T_u \mid k) - w(k+T_u)$$

and $\beta(k)$ is the component which cannot be influenced by the previous control actions, i.e. by $\Delta u(j)$ for $j \leqq k$.

The parameters of the Bellman function (44) are:

$Q_0(k) \geqq 0$ ... diagonal matrix of dimension ∂y,

$m_{0w}(k)$ ... column-vector of dimension ∂y,

$m_{0r}(k)$ ... matrix of dimensions $\partial y \times \partial y_c$ $(\partial y_c = \partial w = \partial r)$
$Q_s(k) \geqq 0$ ... diagonal matrix of dimension $\partial s = n\,\partial y$,
$m_w(k)$ ... column-vector of dimension ∂s,
$m_r(k)$ ... matrix of dimensions $\partial s \times \partial r$,
$M_s(k)$ ... monic LT-matrix of dimension ∂s.

In case of *positional servo* the optimal control law and the Bellman function are modified so that $m_{uw}(k) = 0$, $m_{0w}(k) = 0$, $m_w(k) = 0$ and

$$r(k + T_u \mid k) = \hat{y}_c(k + T_u \mid k) - w(k), \tag{46}$$
$$\bar{r}(k) = \hat{y}_c(k - 1 + T_u \mid k - 1) - w(k)$$

Remark (*a*). The prediction $\hat{y}_c(k - 1 + T_u \mid k - 1)$ and $\hat{s}(k - 1 + T_u \mid k - 1)$ on which the optimal controller operates can be calculated by T_u steps of the recursion (4.66) and (4.67) starting with the state estimate $\hat{s}(k - 1 \mid k - 1)$ supplied by the state estimator according to Result (4A) in single-output case or according to Result (4 B) in multi-output case. Clearly, for $T_u = 0$ no prediction is required and $\hat{y}(k - 1 \mid k - 1) = y(k - 1)$ which is available when $u(k)$ is generated. An alternative way how to handle the process delay is to extend the model order by T_u and to proceed according to the first method described in the paragraph 4.5, however, at the cost of higher computational burden.

Remark (*b*). The Bellman function being a nonnegative definite form can be factorized and/or decomposed in various ways. This means that the right-hand side of (44) is not unique. The form given in (44) has been found algorithmically and numerically most advantageous among a number of other possibilities which have been investigated. This holds also for the algorithms described in the sequel.

Remark (*c*). Since the criterion is quadratic and the employed models are linear the algorithm of dynamic programming is reduced to operations on quadratic forms. For numerical and algorithmic reasons it is more advantageous to perform the main optimization stage (31) and (32) using the dyadic reduction instead of following the route of the Riccati-like equation to which the recursive relation (30) could be brought in this linear-quadratic case. However, the reader should be acquainted with the decomposition and minimization of nonnegative definite quadratic forms described in the paragraph 2.3.

Single-input single-output process, $T_u = 0$

To make the exposition easier to follow the Result (5 B) will first be proved for this simplest case. The proof will be given by designing an algorithm which propagates the Bellman function in the form (44) producing the optimal control law (42) at the same time.

Consider the following decomposition of the loss function

$$l_T = |Q_r(T); r(T)| + |Q_u(T); \Delta u(T)| + |Q_s(T); M_s(T)\,\hat{s}(T \mid T)| \tag{47}$$

(48) $$l_k = |Q_r(k); r(k)| + |Q_u(k); \Delta u(k)|, \quad k < T$$

In the given simple case

(49) $$r(k) = y(k) - w(k), \quad \bar{r}(k) = y(k-1) - w(k)$$

Let us suppose that the parameters of the Bellman function (44) are given numerically for some $k < T$ and let us perform one step of dynamic programming from k to $k-1$. To make the writing easier to survey we shall occasionally omit the time indexing of numerically given quantities when no confusion can occur. First we shall design the algorithm for the program control. It can be easily modified for the case of positional servo afterwards.

It is advantageous to decompose the algorithm into the following stages.

Stage 1*a*. To prepare $B_k + l_k$ for calculating the conditional mean (31) it is suitable to modify, using the dyadic reduction, the following sum of two quadratic forms so that $r(k)$ appears only in one of them.

(50) $$|Q_0(k); m_{0w}(k) + m_{0r}(k)\, r(k)| + |Q_r(k); r(k)| =$$

$$= \left|\begin{bmatrix} Q_0 \\ Q_r \end{bmatrix}; \begin{bmatrix} m_{0w} & m_{0r} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ r(k) \end{bmatrix}\right| = \left|\begin{bmatrix} \bar{Q}_0 \\ \bar{Q}_r \end{bmatrix}; \begin{bmatrix} m_{0w} & 0 \\ m_{rw} & 1 \end{bmatrix} \begin{bmatrix} 1 \\ r(k) \end{bmatrix}\right| =$$

$$= |\bar{Q}_0; m_{0w}| + |\bar{Q}_r; m_{rw} + r(k)|$$

The second equality in (48) means application of the dyadic reduction to reduce m_{0r} to zero. In the given simple case the following explicit formulae can perform the task.

(51) $$\bar{Q}_r = Q_r + Q_0 m_{0r}^2, \quad m_{rw} = (Q_0/\bar{Q}_r)\, m_{0r} m_{0w}, \quad \bar{Q}_0 = (Q_r/\bar{Q}_r)\, Q_0$$

The reason for the modification (48) is that the first term on the most right-hand side of (48) cannot be influenced by the previous control actions and has to be added to the unreducible part $\beta(k)$ of the Bellman function (44) in order to prevent the growing of certain numbers within the numerical algorithm of optimum control synthesis.

(52) $$\Delta\beta_w(k) = |\bar{Q}_0; m_{0w}| = \bar{Q}_0 m_{0w}^2$$

Stage 1*b*. For calculating the conditional mean (31), first over $y(k)$ and than over $v(k)$, it is suitable to express $r(k)$ and $\hat{s}(k \mid k)$, which enter $B_k + l_k$ as random variables, in the following way.

(53) $$r(k) = y(k) - w(k) = y(k-1) - w(k) + \Delta y(k) =$$

$$= \bar{r}(k) + \Delta\hat{y}(k \mid k-1; u(k)) + \varepsilon(k)$$

where

$$\varepsilon(k) = y(k) - \hat{y}(k \mid k-1; v(k), u(k))$$

From (4.20) we have

$$\hat{s}(k \mid k) = \hat{s}(k \mid k-1; v(k), u(k)) + (\tilde{c}(k) - \bar{a})\, \varepsilon(k)$$

This makes it possible to express $B_k + l_k$ as follows

$$B_k + l_k = |Q_u; \Delta u(k)| + |\bar{Q}_r; m_{rw} + \bar{r}(k) + \Delta\hat{y}(k \mid k-1; v(k), u(k)) + \varepsilon(k)| +$$
$$+ |Q_s; m_w + m_r \bar{r}(k) + [m_r, M_s] \begin{bmatrix} \Delta\hat{y}(k \mid k-1; v(k), u(k)) \\ \hat{s}(k \mid k-1; v(k), u(k)) \end{bmatrix} + (m_r + M_s \cdot$$
$$\cdot (\tilde{c} - \bar{a}))\, \varepsilon(k)| + \beta(k) + \Delta\beta_w(k)$$

Now taking into account that according to (4.22)

$$\mathrm{Var}\,[\varepsilon(k) \mid k-1; v(k), u(k)] = \mathrm{Var}\,[y(k) \mid k-1; v(k), u(k)] = \varrho\, d_y(k)$$

and that according to (41)

$$\Delta\hat{y}(k \mid k-1; v(k), u(k)) = \hat{s}_1(k-1 \mid k-1) + b_0\, \Delta u(k) + d_0\, \Delta v(k)$$
$$\begin{bmatrix} \Delta\hat{y}(k \mid k-1; v(k), u(k)) \\ \hat{s}(k \mid k-1; v(k), u(k)) \end{bmatrix} = A^{-1}(H\hat{s}(k-1 \mid k-1) + b\, \Delta u(k) + \mathrm{d}\, \Delta v(k))$$

we can determine the conditional mean over $y(k)$

$$\mathsf{E}[B_k + l_k \mid k-1; v(k), u(k)] = |Q_u; \Delta u(k)| +$$
$$+ |\bar{Q}_r; m_{rw} + \bar{r}(k) + \hat{s}_1(k-1 \mid k-1) + b_0\, \Delta u(k) + d_0\, \Delta v(k)| +$$
$$+ |Q_s; m_r \bar{r}(k) + G(H\hat{s}(k-1 \mid k-1) + b\, \Delta u(k) + d\, \Delta v(k))| +$$
$$+ \beta(k) + \Delta\beta_w(k) + \Delta\beta_y(k)$$

where

$$G = [m_r, M_s]\, A^{-1} \tag{54}$$

$$\Delta\beta_y(k) = \varrho\, d_y(k)\, (\bar{Q}_r + |Q_s; m_r + M_s(\tilde{c} - \bar{a})|) \tag{55}$$

Now it is already easy to determine the conditional mean over $v(k)$ and thus to complete the evaluation of (31). According to (37) and (38) we have $\mathsf{E}[\Delta v(k) \mid k-1; u(k)] = 0$ and if we denote

$$\varrho_v(k) = \mathrm{Var}\,[e_v(k)]$$

then (31) gives for the case of program control

$$F_k(\Delta u(k), \bar{r}(k), \hat{s}(k-1 \mid k-1)) = \mathsf{E}[B_k + l_k \mid k-1; u(k)] = \tag{56}$$
$$= |Q_u; \Delta u(k)| + |\bar{Q}_r; b_0\, \Delta u(k) + m_{rw} + \bar{r}(k) + \hat{s}_1(k-1 \mid k-1)| +$$
$$+ |Q_s; m_u\, \Delta u(k) + m_w + m_r \bar{r}(k) + GH\hat{s}(k-1 \mid k-1)| +$$
$$+ \beta(k) + \Delta\beta_w(k) + \Delta\beta_y(k) + \Delta\beta_v(k)$$

where

$$m_u = Gb \tag{57}$$

$$\Delta\beta_v(k) = \varrho_v(k)\, (|\bar{Q}_r; d_0| + |\bar{Q}_s; Gd|) \tag{58}$$

To complete the first stage of the algorithm of dynamic programming we shall show how the matrix G (54) and the vector m_u (57) can be efficiently calculated. Making

use of (4.15) we obtain

$$G = [m_r, M_s]\begin{bmatrix} 1 & 0 \\ -\bar{a} & I \end{bmatrix} = [m_a, M_s] \tag{59}$$

where

$$m_a = m_r - M_s\bar{a} \tag{60}$$

Substituting of (59) into (57) gives

$$m_u = [m_a, M_s]\begin{bmatrix} b_0 \\ \bar{b} \end{bmatrix} = m_a b_0 - M_s\bar{b} \tag{61}$$

Also the product GH can be calculated very easily

$$GH = [m_a, M_s]\left(\mu\begin{bmatrix} 0 \\ I \end{bmatrix} + \begin{bmatrix} I \\ 0 \end{bmatrix}\right) = \mu M_s + [m_a, M_s]\begin{bmatrix} I \\ 0 \end{bmatrix} \tag{62}$$

Recall that for Delta models $\mu = 1$ while for ARMA models $\mu = 0$. Note the special form of GH which is close to a monic LT-matrix. This will appear advantageous in the following stage of the algorithm.

Stage 2a. Now the dyadic reduction will be emloyed to minimize $F_k(\Delta u(k), \bar{r}(k), \hat{s}(k-1 \mid k-1))$ with respect to $\Delta u(k)$. For this purpose it is suitable to rearrange (56) into a single quadratic form the scheme of which is shown in Fig. 7. In the top row of this scheme the variables are indicated by which the underlying columns are multiplied when the quadratic form is evaluated as a sum of weighted squares. Each row produces one square the weight of which is given in the most left column. Empty spaces are zeros which do not enter the calculation. Note the special form of *GH*.

	Δu	1	$\bar{r}$	$\hat{s}_1$	$\hat{s}_2$	.	.	$\hat{s}_n$
Q_u	1	0	0	0	0	0	0	0
Q_r	b_0	m_{rv}	1	1				
.	.	.	.	.	1			
.	.	.	.	.	.	1		
Q_s	m_u	m_w	m_r	.	GH	.	1	
.	.	.	.	.	.	.	.	1
.	.	.	.	.	.	.	.	.

Fig. 7. Scheme of the quadratic form F_k to be minimized.

To minimize such a quadratic form with respect to $u(t)$ it is sufficient to apply the dyadic reduction so that the one lying under $\Delta u(t)$ is used to reduce the rest of the underlying column to zeros as shown in Fig. 8. Note that neither the zeros in the empty space of the scheme nor the ones indicated in Fig. 8 are destroyed if the reduction is performed downwards. At the same time the first row of the numerical area in the scheme, originally filled with zeros (Fig. 7), becomes, in general, nonzero as indicated in Fig. 8. After this modification the only square of the quadratic form

F_k which depends on the choice of $u(k)$ is produced by this first row

$$\tilde{Q}_u(\Delta u(k) + m_{uw} + m_{ur}\bar{r}(k) + m_{us}\hat{s}(k-1 \mid k-1))^2 \tag{63}$$

and can be totally zeroed, and thus the nonnegative definite

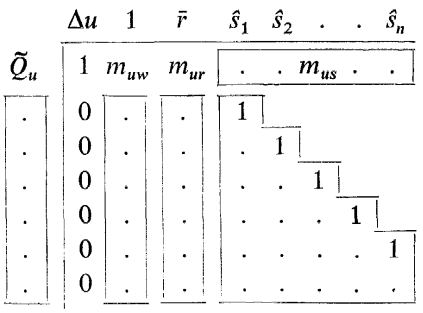

Fig. 8. Minimization of the quadratic form F_k by dyadic reduction.

quadratic form F_k minimized, by the choice

$$\Delta u(k) = -m_{uw} - m_{ur}\bar{r}(k) - m_{us}\hat{s}(k-1 \mid k-1) \tag{64}$$

This proves the optimal control law (42) for the given simple case. Note that the weight $\tilde{Q}_u$ in (63) determines the increase of the criterion if the optimal control law could not be fulfilled. If $\tilde{Q}_u$ is small then the minimum is flat.

Stage 2b. If the square (63) represented by the first numerical row in Fig. 8 is zeroed by the optimal control law (65) then the remaining rows represent the minimalized quadratic form F_k^*. However, it is necessary to transform it into the form of the Bellman function (44). This can be done by reducing the last row as depicted

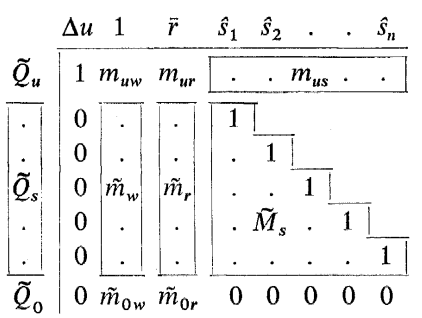

Fig. 9. Reduction of the last row – reconstruction of the propagated form of the Bellman function.

in Fig. 9. Again the dyadic reduction can be employed to perform the task. However, not to destroy the desired structure the last row must be reduced from right to left using, as reducing, the rows with ones lying above the zeroed entries.

After this modification the minimized quadratic form F_k^* can be decomposed

in the following way.

$$F_k^*(\bar{r}(k), \hat{s}(k-1 \mid k-1)) = |\tilde{Q}_0(k); \tilde{m}_{0w}(k) + \tilde{m}_{0r}(k)\,\bar{r}(k)| + \tag{65}$$
$$+ |\tilde{Q}_s(k); \tilde{m}_w(k) + \tilde{m}_r(k)\,\bar{r}(k) + \tilde{M}_s(k)\,\hat{s}(k-1 \mid k-1)| +$$
$$+ \beta(k) + \Delta\beta_w(k) + \Delta\beta_y(k) + \Delta\beta_v(k)$$

Stage 3. The Bellman function B_{k-1} is obtained from F_k^* if in (65) it is expressed

$$\bar{r}(k) = y(k-1) - w(k) = y(k-1) - w(k-1) - \Delta w(k) = \tag{66}$$
$$= r(k-1) - \Delta w(k)$$

Since in program control $\Delta w(k)$ is known in advance the operation (34) is replaced by the following correction of $\tilde{m}_w(k)$ and $\tilde{m}_{0w}(k)$

$$m_w(k-1) = \tilde{m}_w(k) - \tilde{m}_r(k)\,\Delta w(k)\,, \tag{67}$$
$$m_{0w}(k-1) = \tilde{m}_{0w}(k) - \tilde{m}_{r0}(k)\,\Delta w(k)$$

Hence the Bellman function B_{k-1} is obtained in the reproduced form (44) with $Q_0(k-1) = \tilde{Q}_0(k)$, $m_{r0}(k-1) = \tilde{m}_{r0}(k)$, $Q_s(k-1) = \tilde{Q}_s(k)$, $m_r(k-1) = \tilde{m}_r(k)$ and $M_s(k-1) = \tilde{M}_s(k)$.

This concludes the recursion and to complete the proof for the case of program control it remains to show that the given structure of the Bellman function and of the optimal control law suit also for $k = T$. Considering the initial condition $B_T = 0$ it is easily seen that besides $Q_s(T)$, $M_s(T)$, $Q_r(T)$ and $Q_u(T)$ given by the last component of the loss function (47) the algorithm starts with

$$m_w(T) = 0\,, \quad m_{0w}(T) = 0\,, \quad m_r(T) = 0\,, \quad m_{0r}(T) = 0\,, \tag{68}$$
$$Q_0(T) = 0\,, \quad \beta(T) = 0$$

Note that $m_w(k)$ and $m_{0w}(k)$ remain zero also for $k < T$ if $\Delta w(k) = 0$ for all k.

Positional servo. If the future course of the command signal is uncertain then the conditional mean (34) has to be determined. If the generalized random walk (39) with

$$\mathsf{E}[\Delta w(k) \mid \mathscr{W}_{k-1}] = 0\,, \quad \mathrm{Var}\,[\Delta w(k) \mid \mathscr{W}_{k-1}] = \varrho_w(k)$$

is used to model the uncertainty of the future command signal at the moment $k = 1$ then the above Stage 3 has to be modified in the following way. Instead of (66) and (67) we now have

$$\mathsf{E}[\bar{r}(k) \mid k-1] = r(k-1)\,, \quad m_w(k-1) = \tilde{m}_w(k) = 0\,,$$
$$m_{0w}(k-1) = \tilde{m}_{0w}(k) = 0$$

This reduces to zero also m_{rw} in (51) which means that the entire corresponding column in the schemes Fig. 7, Fig. 8 and Fig. 9 can be omitted. Apparently, also the increment (52) of the component $\beta(k)$ of the Bellman function is reduced to zero but it is replaced by an other increment produced by the conditional mean (34)

with F_k^* given by (65).

$$\Delta\beta_w(k) = \varrho_w(k)\,(|\tilde{Q}_0(k);\, \tilde{m}_{0r}(k)| + |\tilde{Q}_s(k);\, \tilde{m}_r(k)|)$$

Process delay $T_u > 0$

It is not difficult to verify that the above algorithm of optimum control synthesis holds also for the process model with time delay if the control error $r(k)$ and the related variable $\bar{r}(k) = y(k-1) - w(k)$ on which the optimal controller operates are replaced by their T_u-steps-ahead predictions (45) and (43) in case of program control, or (46) in case of positional servo. The only difference caused by the process delay is a more complex evaluation of the increments of the component $\beta(k)$ of the Bellman function. Since they influence only the minimum of the criterion which can be achieved by the optimal control, but not the optimal control law itself, they are not followed in detail here. The PASCAL procedure *LQGI* listed below can be used also for processes with delays.

Stochastic deadbeat control

Recall that the mean value of the state s of an incremental model is zero in steady state.

Since the weights $Q_r(k)$ and $Q_u(k)$, by which the control error and the change in the position of the actuator are incorpotated into the quadratic criterion, can be different for different k it is possible to choose $Q_s(T) > 0$, $Q_r(T) > 0$, $Q_u(T) = 0$ and for $k < T$, $Q_r(k) = 0$, $Q_u(k) = 0$. By inspection of the above algorithm it can be verified that with this choice the rank of the matrix on which the algorithm operates is decreased by one in each step if dynamic programming. Since for $k = T$ this rank is, in general, $n + 1$ the stochastic deadbeat control can be obtained by $n + 1$ steps of dynamic programming. In such a case the minimum of the criterion is $B_0 = \beta(0)$ and does not depend on the state estimate $\hat{s}(0 \mid 0)$.

It is a great advantage of the numerical algorithm based on dyadic reduction that it can safely manage such singular cases. This means that the below listed procedure *LQGI* can well be used to calculate deadbeat control and other (often more reasonable) modifications of the quadratic criterion.

Remark (*d*). Note that the parameters of the optimal control law (42) do not depend on the *c*-parameters of the process model. The *c*-parameters are reflected only in the state estimate $\hat{s}(k-1 \mid k-1)$ on which the optimal controller (including the prediction if required) operates. Note also that the measurable external disturbance, v, if the uncertainty of its future development is modelled by a generalized random walk, does not enter the optimal control law explicitly. The feedforward from v only helps to improve the estimate of the state.

PASCAL procedure LQGI

The following procedure performs one step of dynamic programming described above and generates the parameters of the optimal control law. Nonstandard types of its parameters are

```
TYPE poly = ARRAY [0..nmax] OF REAL;
     row = ARRAY [0..jmax] OF REAL;
     Matr = ARRAY [0..imax] OF row;
```

where the integer constant nmax is the maximal model order, jmax = nmax + 3, imax = nmax + 1. Since the procedure *LQGI* makes use of the procedure *DYDR* (see Section 2) the type *row* must be common for both of them.

Parameters:

mti ... type INTEGER; model-type indicator (in text denoted by μ); set *mti* = 1 for Delta model, *mti* = 0 for ARMA.

n ... type INTEGER; model order.

a, b ... type poly; model parameters; $a[i] = a_i$; $b[i] = b_i$.

Qr, Qu ... type REAL; nonnegative weights by which the square of the control error and the square of the change in the position of the actuator are penalized in the control step for which the control law is calculated.

dw ... type REAL; *dw* = 0 in case of positional servo; increment of the command signal preprogrammed for the next control step in case of program control (T_u steps ahead in case of process delay if the model state is not extended).

M ... type *Matr*; the matrix on which the procedure operates; both in call-state and in return-state the submatrices are placed as depicted in Fig. 9; parameters of the optimal control law appear in the first row indexed by $iu = 0$: $m_{uw} = M[iu, jw]$, $m_{ur} = M[iu, jr]$, $m_{us,j} = M[iu, jr + j]$, $j = 1..n$; the nonnegative weights are placed in the first column with the column index $jQ = 0$; the first row indexed by $iu = 0$ can be arbitrary when the procedure is called; rows with indices $iu + i$, $i = 1..n + 1$, must be initialized before the first call of the procedure *including the ones* $M[iu + i, jr + i] = 1$, $i = 1..n$. Large initial entries $M[iu + i, jQ] = Q_{s,i}(T)$ stabilize the control loop.

```
PROCEDURE LQGI (mti, n: INTEGER; a, b: poly; Qr, Qu, dw: REAL;
     VAR M: Matr);
CONST iu = 0; ir = 1;
     jQ = 0; ju = 1; jw = 2, jr = 3; js = 4;
VAR  i, j, ii, jj, i1, im, jn: INTEGER;
     am, bm, Mij: REAL;
BEGIN
jn := n + jr; im:= n + ir;
```

```
Mij := M[im, jr]; Qr := Qr + M[im, jQ] * Mij * Mij;
M[ir, jw] := M[im, jQ]/Qr * Mij * M[im, jw];
i1 := im;
FOR i := n DOWNTO 1 DO
  BEGIN
  ii := i1-1;
  am := M[ii, jr] - a[i]; bm := b[i];
  jj := js;
  FOR j := 1 TO i - 1 DO
    BEGIN
    Mij := M[ii, jj];
    am := am - Mij * a[j]; bm := bm + Mij * b[j];
    jj := jj + 1;
    M[i1, jj] := mti * M[ii, jj] + Mij
    END;
  M[i1, js] := mti * M[ii, js] + am;
  M[i1, ju] := bm + am * b[0];
  M[i1, jQ] := M[ii, jQ]; M[i1, jr] := M[ii, jr];
  i1 := ii
  END;
M[ir, jQ] := Qr; M[ir, ju] := b[0]; M[ir, jr] := 1;
M[iu, jQ] := Qu;
FOR j := jw TO jn DO M[iu, j] := 0;
jj := js;
FOR i := ir TO im DO
  BEGIN
  DYDR(M[i], M[iu], M[i, jQ], M[iu, jQ], ju, jw, jj);
  IF jj < jn THEN jj := jj + 1
  END;
FOR i := im - 1 DOWNTO ir DO
  BEGIN
  j := jj - 1;
  DYDR(M[im], M[i], M[im, jQ], M[i, jQ], jj, jw, j);
  jj := j
  END;
IF dw <> 0 THEN
  FOR i := ir TO im DO M[i, jw] := M[i, jw] - M[i, jr] * dw
END;
```

Remark (*e*). If the procedure *LQGI* has to be used for the most common case of positional servo or of regulation then it suffices to set the procedure parameter *dw* permanently to zero. However, in such a case the following simplification can be recommanded:

– Omit the procedure parameter *dw*, the third line of the body where $M[ir, jw]$ is calculated, and the last IF statement.
– Replace *jw* by *jr*.
– In CONST declaration omit *jw* and set $jr = 2$ and $js = 3$.

Multivariate case

The extension of the above algorithm of optimum control synthesis for a multivariate case ($\partial y \geqq 1$, $\partial u \geqq 1$, $\partial y_c = \partial w = \partial r \leqq \partial y$) is rather straightforward. If we omit the calculation of the increments of $\beta(k)$ which do not influence the optimal control law then it suffices to consider, instead of rows and columns in the schemes Fig. 7, Fig. 8 and Fig. 9, block-rows and block-columns of dimensions given in Result (5B), to replace the ones by monic LT-matrices, and to modify the particular stages of the algorithm in the following way.

Stage 1*a*. Instead of the explicit formulae (51) the modification (50), which now reads

$$|Q_0; m_{0w} + m_{0r}\, r(k)| + |Q_r; M_r\, r(k)| = |\bar{Q}_0; m_{0w}| + |\bar{Q}_r; m_{rw} + \bar{M}_r\, r(k)|$$

where M_r and $\bar{M}_r$ are monic LT-matrices, must be performed using dyadic reduction.

Stage 1*b*. Instead of explicit inverting of the monic LT-matrix A (see (4.48), as applied in (59) and (60), it is now more advantageous to calculate the first ∂y columns m_a of the matrix G recursively using the relation following from (54).

$$GA = [m_a, M_s]$$

Some care must be exercised when calculating the second quadratic form on the right-hand side of (56) corresponding to the second numerical block-row in Fig. 7, especially when the number of controlled outputs is lower then the number of observed outputs, i.e. when $\partial y_c < \partial y$. If the vector of controlled outputs y_c is placed in y as first, $y' = [y_c', \cdot]$, and the model parameters a_0, a monic LT-matrix, and b_0 are partitioned correspondingly

$$a_0 = \begin{bmatrix} a_{0c} & 0 \\ \cdot & \cdot \end{bmatrix}, \quad b_0 = \begin{bmatrix} b_{0c} \\ \cdot \end{bmatrix}$$

then from the first ∂y_c rows of the model equation (41) it is obtained

$$\Delta \hat{y}_c(k + T_u \,|\, k - 1; u(k)) = a_{0c}^{-1}(\hat{s}_{1c}(k - 1 + T_n \,|\, k - 1) + b_{0c}\, \Delta u(k))$$

where s_{1c} is the corresponding subcomponent of state component s_1. However, in order to match the rest of the algorithm (reduction of the last block-row in Stage 2b) the second numerical block row in Fig. 7 must consist of ∂y rows and therefore the discussed quadratic form has to be extended in a way which does not change its value.

$$\left| \begin{bmatrix} \bar{Q}_r \\ 0 \end{bmatrix}; \begin{bmatrix} a_{0c}^{-1} b_{0c} \\ 0 \end{bmatrix} \Delta u(k) + \begin{bmatrix} m_{rw} \\ 0 \end{bmatrix} + \begin{bmatrix} a_{0c}^{-1} \bar{M}_r \\ 0 \end{bmatrix} \bar{r}(k) + \right.$$
$$\left. + \begin{bmatrix} a_{0c}^{-1} & 0 \\ 0 & \mathrm{I} \end{bmatrix} \hat{s}_1(k - 1 + T_u \,|\, k - 1) \right|$$

The inversion of a_{0c} is a very low price paid for a drastic simplification of state estimation, and especially of joint parameters and state estimation (see Section 6) in multivariate case. As a_{0c} is a monic LT-matrix the caclulation can be performed by a simple recursion operating on all matrix entries in the upper part of the block-row which have to be multiplied by this inverse.

Stage 2*a*. The only modification is that instead of a single column a block of ∂u columns has to be reduced to zero sequentially from left to right.

Stage 2*b*. Similarly, the block of ∂y last rows has to be reduced to zero proceeding upwards in order not to destroy the structure.

Stage 3. In case of program control (67) remains valid if m_r and m_{0r} are interpreted as matrices of dimensions $\partial s \times \partial r$ and $\partial y \times \partial r$, respectively. Note that even in multivariate case m_w and m_{w0} are just single columns.

In case of positional servo or of regulation no calculation is performed in Stage 3 and the procedure can be simplified similarly to single output case (see Remark (e)).

5.4. Algorithm of optimum control synthesis for positional process models

In this paragraph the Result (5 B) will be modified for the positional process model (3.63)

(69)
$$A\begin{bmatrix} y(k + T_u) \\ s(k + T_u) \end{bmatrix} = Hs(k - 1 + T_u) + bu(k) + dv(k + T_u - T_v) + ce(k + T_u) + k_x$$

The matrix parameters A, b, c, d and k_x are defined by (3.66) and (3.67) while the nonparametric matrix H will be considered in the form (4.6) to cover both ARMA and Delta cases. Compare (69) with the previously considered incremental model (41) and recall that

$$k'_x = [0, 0, \ldots, 0, k'_c]$$

Result (5 C): Optimal controller for positional process models.

Let the controlled process be described by the positional model (69) and let the measurable external disturbance be considered as the generalized random walk (37). Then the control law minimizing the quadratic criterion (35) for the case of *positional servo* and $T_v \geqq T_u$ can be given the structure

(70) $$\Delta u(k) = -m_{uu}(k)\, u(k - 1) - m_{up}(k) - m_{uw}(k)\, w(k) - m_{uv}(k)\, v(k - 1) - \\ - m_{us}(k)\, \hat{s}(k - 1 + T_u \mid k - 1)$$

where $\hat{s}(k - 1 + T_u \mid k - 1)$ is the T_u-steps-ahead prediction of the state of the model (69). The parameters of the control law (70) can be determined by the algorithm of dynamic programming with the Bellman function of the following nonnegative

definite quadratic form

(71) $$B_k =$$

$$= |Q_s(k);\ m_u(k)\,u(k) + m_w(k)\,w(k) + m_p(k) + m_v(k)\,v(k) + M_s(k)\,\hat{s}(k + T_u \mid k)| +$$
$$+ |Q_p(k);\ m_{pu}(k)\,u(k) + m_{pw}(k)\,w(k) + m_{pp}(k) + m_{pv}(k)\,v(k)| + \beta(k)$$

where $M_s(k)$ and $m_{pu}(k)$ are monic LT-matrices, $m_p(k)$ and $m_{pp}(k)$ are vectors of dimensions ∂s and ∂u, respectively, and the rest of parameters are rectangular matrices of appropriate dimensions. $\beta(k)$ is the component which cannot be influenced by previous control actions.

In case of *program control*, when the command signal is given a priori for the entire control horizon, the terms $m_w(k)\,w(k)$ and $m_{pw}(k)\,w(k)$ can be omitted in the Bellman function and the command signal can be incorporated into the absolute terms $m_p(k)$ and $m_{pp}(k)$. Then the optimal control law does not contain the term $m_{uw}(k)\,w(k)$.

Remark (*d*). The Remarks (a, b, c) from the previous paragraph are relevant also here.

Single-input single-output process, $T_u = 0$

Since the extensions for the multivariate case and for a nonzero process delay are very similar to the case of incremental model the Result (5 C) will be proved and the algorithm for optimum control synthesis will be designed only for this simple case.

Similarly to (47) and (48) it is suitable to decompose the loss function as follows.

$$l_T = |Q_r(T);\ r(T)| + |Q_u(T);\ \Delta u(T)| + |Q_s(T);\ M_s(T)\,(\hat{s}(T \mid T) - \bar{s})|$$
$$l_k = |Q_r(k);\ r(k)| + |Q_u(k);\ \Delta u(k)|, \quad k < T$$

First we shall perform a general step of dynamic programming from k to $k - 1$ and afterwards we shall consider the first step for $k = T$ and discuss the proper choice of the last desired state $\bar{s}$ in the criterion (35).

Again, one step of dynamic programming will be decomposed into three stages.

Stage 1. For calculating the conditional mean (31) it is suitable to introduce

$$\varepsilon(k) = y(k) - \hat{y}(k \mid k - 1;\ v(k),\ u(k))$$

and to express $\hat{s}(k \mid k)$ according to (4.20)

(72) $$\hat{s}(k \mid k) = \hat{s}(k \mid k - 1;\ v(k),\ u(k)) + (\tilde{c}(k) - \vec{a})\,\varepsilon(k)$$

Clearly

$$y(k) = \hat{y}(k \mid k - 1;\ v(k),\ u(k)) + \varepsilon(k)$$

From the model (69) we have

(73) $$A\begin{bmatrix}\hat{y}(k \mid k - 1;\ v(k),\ u(k))\\ \hat{s}(k \mid k - 1;\ v(k),\ u(k))\end{bmatrix} = H\,\hat{s}(k - 1 \mid k - 1) + b\,u(k) + d\,v(k) + k_x$$

the first row of which gives

$$\hat{y}(k \mid k-1; v(k), u(k)) = \hat{s}_1(k-1 \mid k-1) + b_0\, u(k) + d_0\, v(k) \tag{74}$$

Using the random-walk model (37) for $v(k)$ it is possible to express the output $y(k)$ as follows.

$$y(k) = \hat{s}_1(k-1 \mid k-1) + b_0\, u(k) + d_0\, v(k-1) + \varepsilon(k) + d_0\, e_v(k) \tag{75}$$

Omitting the time index of numerically given quantities and making use of (73) we can also write

$$M_s\, \hat{s}(k \mid k) = [0, M_s] \begin{bmatrix} \hat{y}(k \mid k-1; v(k), u(k)) \\ \hat{s}(k) \mid k-1; v(k), u(k)) \end{bmatrix} + M_s(\tilde{c} - \bar{a})\, \varepsilon(k) = \tag{76}$$

$$= GH\, \hat{s}(k-1 \mid k-1) + Gb\, u(k) + Gd\, v(k-1) + Gk_x + M_s(\tilde{c} - \bar{a})\, \varepsilon(k) + Gd\, e_v(k)$$

where

$$G = [0, M_s]\, A^{-1} = [-M_s\bar{a}, M_s]$$

Recalling that $b' = [b_0, \bar{b}']$ and $d' = [d_0, \bar{d}']$ we have

$$Gb = -M_s\bar{a}b_0 + M_s\bar{b}\,, \quad Gd = -M_s\bar{a}d_0 + M_s\bar{d}$$

Since M_s is a monic LT-matrix we also have

$$(Gk_x)' = [0, 0, \ldots, 0, k_c]$$

Now it is already easy to calculate the conditional mean (31) $F_k = \mathsf{E}[B_k + l_k \mid |D_{k-1}, u(k), w(k)]$. Since the random variables $\varepsilon(k)$ and $e_v(k)$ have zero mean, by their definition are uncorrelated and their variances are

$$\mathrm{Var}\,[\varepsilon(k) \mid k-1] = \varrho\, d_y(k)\,, \quad \mathrm{Var}\,[e_v(k) \mid \mathscr{V}_{k-1}] = \varrho_v(k)$$

it is obtained

$$F_k = |Q_u;\, \Delta u(k)| + |Q_p;\, u(k) + m_{pw}\, w(k) + m_{pp} + m_{pv}\, v(k-1)| + |Q_r;\, b_0\, u(k) - w(k) + d_0\, v(k-1) + \hat{s}_1(k-1 \mid k-1)| + |Q_s;\, \bar{m}_u\, u(k) + m_w\, w(k) + \bar{m}_p + \bar{m}_v\, v(k-1) + GH\, \hat{s}(k-1 \mid k-1)| + \beta(k) + \Delta\beta_y(k) + \Delta\beta_v(k) \tag{77}$$

where

$$\bar{m}_u = m_u + Gb = m_u - m_a b_0 + m_b\,, \quad \bar{m}_v = m_v + Gd = m_u - m_a d_0 + m_d \tag{78}$$

$$m_a = M_s\bar{a}\,, \quad m_b = M_s\bar{b}\,, \quad m_d = M_s\bar{d} \tag{79}$$

$$\bar{m}_p = m_p + Gk_x\,, \quad \bar{m}'_p = m'_p + [0, 0, \ldots, 0, k_c] \tag{80}$$

$$GH = [-m_a, M_s]\, H = \mu M_s + [-m_a, M_s] \begin{bmatrix} I \\ 0 \end{bmatrix} \tag{81}$$

Hence, the main calculation, which is to be performed in this stage of the algorithm, is the determination of the vectors m_a, m_b and m_d, all of dimension n, according to (79). The fact that M_s is a monic LT-matrix simplifies the calculation.

Stage 2. To perform the minimization of F_k with respect to $u(k)$ according to (32) and (33) it is suitable to express the sum of nonnegative quadratic forms in (77) as a single quadratic form as shown in the scheme Fig. 10 where $\Delta u = \Delta u(k)$ and $u = u(k-1)$.

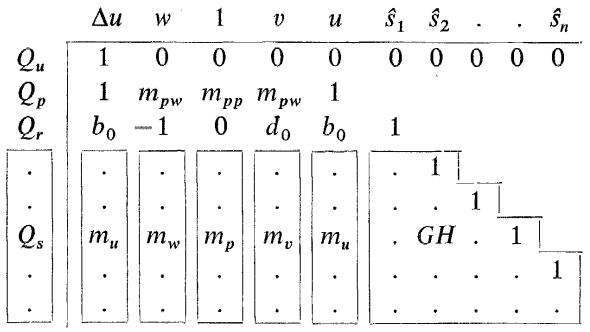

Fig. 10. Scheme of the quadratic form F_k for positional models.

Note that $u(k)$ is expressed as $\Delta u(k) + u(k-1)$. As before, the empty spaces mean zeros which do not enter the calculation and the dots mean the numbers in general nonzero.

To minimize the quadratic form with respect to $\Delta u = \Delta u(k)$ it is now sufficient to use the first row as reducing and to zero the rest of the column lying under Δu. If the dyadic reduction is applied sequentially from up to down the required structure of the entire matrix is maintained as shown in Fig. 11. Because the minimization procedure is, in principle, the same as in the above case of incremental models we can shorten the commentary.

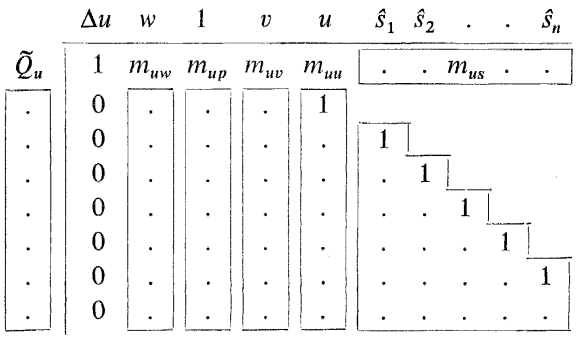

Fig. 11. Minimization of the quadratic form F_k for positional models by dyadic reduction.

After this transformation of the nonnegative quadratic form the only square which depends on the choise of $\Delta u(k)$ is produced by the first row and is totaly zeroed

by the control law

(82) $$\Delta u(k) =$$

$$= - m_{uu}\, u(k-1) - m_{uw}\, w(k) - m_{up} - m_{uv}\, v(k-1) - m_{us}\, \hat{s}(k-1 \mid k-1)$$

which coincides with the general form of the optimal control law (70) for the given simple case.

The propagated form of the Bellman function can be reconstructed by reducing the right hand part of the last row as shown in Fig. 12. This reduction must be performed from right to left in order not to destroy the desired structure of the form. Note the right positioning of the column multiplied by $u = u(k-1)$ with the one in the proper place. This makes the algorithm compact. The square produced by the last row in Fig. 12 does not depend on the previous control actions and thus contributes to the β-part of the Bellman function.

(83) $$\Delta\beta_0(k) = \mathsf{E}[Q_0;\, m_{0w}\, w(k) + m_{0p}(k) + m_{0v}(k)\, v(k-1) \mid D_0]$$

	Δu	w	1	v	u	$\hat{s}_1$	$\hat{s}_2$	.	.	$\hat{s}_n$
$\tilde{Q}_u$	1	m_{uw}	m_{up}	m_{uv}	m_{uu}	.	.	m_{us}	.	.
$\tilde{Q}_p$	0	$\tilde{m}_{pw}$	$\tilde{m}_{pp}$	$\tilde{m}_{pv}$	1					
.	0	.	.	.	.	1				
.	0	.	.	.	.	.	1			
$\tilde{Q}_s$	0	$\tilde{m}_w$	$\tilde{m}_p$	$\tilde{m}_v$	$\tilde{m}_u$	.	$\tilde{M}_s$	1		
.	0	.	.	.	.	.	.	.	1	
.	0	.	.	.	.	.	.	.	.	1
Q_0	0	m_{0w}	m_{0p}	m_{0v}	0	0	0	0	0	0

Fig. 12. Reconstruction of the propagated form of the Bellman function by the reduction of the last row.

Stage 3. In this last stage the conditional mean over $w(k)$ has to be taken according to (34). If the a priori uncertain command signal is modelled as the generalized random walk (39) then $\mathsf{E}[w(k) \mid \mathcal{W}_{k-1}] = w(k-1)$. This means that it is sufficient to replace $w(k)$ in the Bellman function by $w(k-1)$ and to increase its β-part, (which actually does not need to be calculated) by

(83) $$\Delta\beta_w(k) = \varrho_w(k)\,(|\tilde{Q}_p;\, \tilde{m}_{pw}| + |\tilde{Q}_s;\, \tilde{m}_w|)$$

where $\varrho_w(k) = \mathrm{Var}\,[e_w(k)]$. Recall that the variances $\varrho_w(k)$ as well as $\varrho_v(k)$ can be arbitrarily time varying.

Program control. In case of program control the a priori numerically given $w(k)$ can be incorporated into the algorithm in Stage 2 so that its negative value is inserted instead of the zero in the row weighted by Q_r and in the column multiplied by 1 in the scheme Fig. 10. This means that the column lying in this scheme under w can be fully omitted.

Choice of the last desired state

It remains to consider the first step of dynamic programming for $k = T$. Since $B_T = 0$ we can write

$$B_T + l_T = |Q_r(T); r(T)| + |Q_s(T); \Delta u(T)| + |Q_s(T); M_s(T)(\hat{s}(T \mid T) - \bar{s})|$$

It is clear that it much depends on the choice of $\bar{s}$. If we chose the desired last state $\bar{s}$ as a numerically given vector it would be possible to consider formally $B_T = |Q_s(T); m_p(T) + M_s(T)\hat{s}(T \mid T)|$ with $m_p(T) = -M_s(T)\bar{s}$ and to proceed as in general step. However, this would not be a very reasonable choice. It is more appropriate to require that the process be as close as possible to a steady state after the last control action $\Delta u(T)$ is applied. This corresponds to the choice

$$\bar{s} = \hat{s}(T-1 \mid T-1) \tag{84}$$

With this choice we can proceed as follows. According to the above relation (76) it holds

$$|Q_s; M_s(\hat{s}(T \mid T) - \hat{s}(T-1 \mid T-1)| =$$

$$= |Q_s; (GH - M_s)\hat{s}(T-1 \mid T-1) + Gb\,u(T) + Gd\,v(T-1) +$$

$$+ Gk_x + M_s(\tilde{c} - \bar{a})\varepsilon(k) + Gd\,e_v(T)|$$

This gives, see (78) to (81),

$$\bar{m}_u(T) = -m_a b_0 + m_b\,, \quad \bar{m}_v(T) = -m_a d_0 + m_d\,, \quad \bar{m}_p'(T) = [0, 0, \ldots, 0, k_c]$$

$$GH - M_s = (\mu - 1)M_s + [-m_a, M_s]\begin{bmatrix} 0 \\ I \end{bmatrix} \tag{85}$$

where

$$m_a = M_s(T)\bar{a}\,, \quad m_b = M_s(T)\bar{b}\,, \quad m_d = M_s(T)\bar{d}$$

This means that it is possible to start the dynamic programming for $k = T$ with $m_u(T) = 0$, $m_w(T) = 0$, $m_p(T) = 0$ and $m_v(T) = 0$ and all what has to be done to achieve the choice (84) is to decrease in the first step of dynamic programming the model type indicator μ by one as shown by (85).

PASCAL procedure LQGP

The following procedure performs one step of dynamic programming and generates the optimal control law for a single-input single-output positional model with zero steady state offset $k_c = 0$ and for the case of positional servo with no feedforward from a measurable external disturbance. However, it can be easily modified for the other cases discussed above. The nonstandard types of its parameters are the same as for the above procedure LQGI. To economize the memory of the computer it is suitable to choose imax = nmax + 2 and jmax = nmax + 3.

Parameters:

mti ... type INTEGER; model type indicator (in text denoted by μ); $mti = 1$ for Delta model, $mti = 0$ for ARMA; in the first call of the procedure *mti* must be decreased by 1 to start dynamic programming with (84), e.g. for ARMA model in the first call set $mti = -1$.

n ... type INTEGER; model order.

a, b ... type *poly*; model parameters; $a[i] = a_i$, $b[i] = b_i$

Qr,Qu ... type REAL; nonegative weights in the criterion (35).

M ... type *Matr*; the matrix on which the procedure operates; both in call-state and in return-state the submatrices are placed as depicted in Fig. 12; parameters of the control law (82) appear in the first row indexed by $iu = 0$: $m_{uw} = M[iu, jw]$, $m_{uu} = M[iu, ju]$, $m_{us,j} = M[iu, ju + j]$, $j = 1, 2, .. n$, (*jw* and *ju* are declared as CONST); the rows with indeces $i = iu + k$, $k = 1, 2, .. n + 1$, must be incialized before the first call of the procedure *including the ones* $M[iu + k, jw + k]$; large nonnegative weights $Q_{s,k} = M[jQ, ip + k]$, $k = 1 .. n$, stabilize the control loop; standard initialization of the remaining entries is zero.

```
PROCEDURE LQGP (mti, n: INTEGER; a, b: poly; Qr, Qu: REAL;
                VAR M: Matr);
CONST iu = 0; ip = 1; ir = 2;
      jQ = 0; jD = 1; jw = 2; ju = 3; js = 4;
VAR i, j, ii, jj, i1, im, jn: INTEGER;
    am, bm, Mij: REAL;
BEGIN
jn := n + jd; im := n + ir;
i1 := im;
FOR i := n DOWNTO 1 DO
  BEGIN
  ii := i1 - 1;
  am := a[i]; bm := b[i];
  jj := js;
  FOR j := 1 TO i - 1 DO
    BEGIN
    Mij := M[ii, jj];
    am := am + Mij * a[j]; bm := bm + Mij * b[j];
    jj := jj + 1;
    M[i1, jj] := mti * M[ii, jj] + Mij
    END;
  M[i1, js] := mti * M[ii, js] - am;
  bm := M[ii, ju] + bm - am * b[0];
  M[i1, jD] := bm; M[i1; ju] := bm;
```

```
    M[i1, jQ] := M[ii, jQ]; M[i1, jw] := M[ii, jw];
    i1 := ii
    END;
  M[iu, jQ] := Qu;
FOR j := jw TO jn DO M[iu, j] := 0 ;
M[ip, jD] := 1;
M[ir, jQ] := Qr; M[ir, jD] := b[0]; M[ir, jw] := -1; M[ir, ju] := b[0];
jj := ju;
FOR i := ip TO im DO
    BEGIN
    DYDR(M[i], M[iu], M[i, jQ], M[iu, jQ], jD, jw, jj);
    IF jj < jn THEN jj := jj + 1
    END;
FOR i := im - 1 DIWNTO ip DO
    BEGIN
    j := jj - 1;
    DYDR (M[im], M[i], M[im, jQ], M[i, jQ], jj, jw, j);
    jj := j
    END
END;
```


REFERENCES

[13] J. S. Meditch: Stochastic Optimal Linear Estimation and Control. McGraw Hill, New York 1969.

[14] V. Peterka: On LQ optimum control of ARMAX processes. Proc. 9th IFAC Congress, Budapest, 1986.

[15] A. P. Sage: Optimum Systems Control. Prentice Hall, Englewood Cliffs 1981.

[16] V. Strejc: State Space Theory of Linear Discrete Control. Academia, Prague 1978.

[17] M. Ullrich: Optimum control of some stochastic systems. Proceedings of the VIIIth conference ETAN, Beograd, November 1964, 291—298.

6. SIMULTANEOUS PARAMETER AND STATE ESTIMATION

In the foregoing sections it was assumed that the parameters of the process model were a priori known. However, this is rarely the case in industrial practice. To make the theory practicable it is highly desirable to develop procedures and reliable numerical algorithms which make it possible to estimate the model parameters in real time under the operating conditions of the control system. This is the topic of the present section.

Usually, an attempt to estimate the parameters of a state-space model jointly with its state leads to a problem of nonlinear filtering which can be practically

solved only with some approximation. The purpose of the first paragraph of this section it so give a general view on the problem and to investigate under what conditions the problem can be simplified if the parameters are estimated separately from the state. It turns out that such a simplification exists for the parameters a, b, d and k_c of the linear models introduced in Section 3.

In the second paragraph a filter is derived which updates the statistics (certain matrices) which make it possible to express the estimate of the model state as a linear function of the unknown parameters a, b, d and k_c. This opens the way for the exact solution of the problem of simultaneous estimation of these parameters and of the model state.

In the third paragraph it is shown that the problem of parameter estimation can be solved exactly within normal probability distributions. A filter is derived which updates the mean values and the covariances of the unknown parameters and determines the joint probability distribution for both the parameters and the state. It turns out that also in this case the variance of the white-noise component of the models does not need to be known if only the conditional means (point estimates) are of interest. It enters the procedure only as a factor by which all covariances are multiplied.

Unfortunately, the c-parameters of the linear models we deal with cannot be exactly estimated in real time. Throughout this section it is assumed that they are suitably chosen as the base of exponentials by which the "tails" of a regression model are approximated as discussed in paragraph 3.2. The procedure developed in the third paragraph of this section provides all probabilistic characteristics which are required to calculate the aposterior probability distribution on a finite number of hypotheses about posible values of these parameters. This can help to make the choice. However, a detailed discussion of this problem is outside the scope of this paper. An interested reader is referred to Section 6 in [10] for the way how to proceed.

With no loss of generality the external measurable disturbance v will not be considered explicitly in this section. It can be considered as an additional process input and the fact that this input cannot be manipulated is not essential for the given problem. If required, the d-parameters of the model can be estimated in the very same way as the b-parameters.

6.1. Conceptual solution

Suppose that a given process can be described by a state space model which, if its patameters are known, defines the c.p.d.f. (3.18)

$$\mathrm{p}(y(t), s(t) \mid s(t-1), u(t)) \tag{1}$$

If a set of parameters θ of this model is unknown then the model does not define (1) but only

$$\mathrm{p}(y(t), s(t) \mid s(t-1), u(t), \theta) \tag{2}$$

To estimate the unknown parameters θ jointly with the state $s(t)$ means, in Bayesian view, to evolve the c.p.d.f.

$$\mathrm{p}(s(t), \theta \mid t) \tag{3}$$

If the state $s(t)$ is extended by the model parameters θ then, following the procedure applied in the paragraph 4.1, it is obtained instead of (4.1)

$$\mathrm{p}(y(t), s(t), \theta \mid t-1; u(t)) = \tag{4}$$
$$= \mathrm{p}(y(t), s(t) \mid s(t-1), \theta, u(t))\, \mathrm{p}(s(t-1), \theta \mid t-1)\, \mathrm{d}s\,(t-1)$$

instead of (4.2)

$$\mathrm{p}(y(t) \mid t-1; u(t)) = \iint \mathrm{p}(y(t), s(t), \theta \mid t-1; u(t))\, \mathrm{d}s(t)\, \mathrm{d}\theta \tag{5}$$

and instead of (4.3)

$$\mathrm{p}(s(t), \theta \mid t) = \frac{\mathrm{p}(y(t), s(t), \theta \mid t-1; u(t))}{\mathrm{p}(y(t) \mid t-1; u(t))} \tag{6}$$

The functional recursion (4), (5) and (6) solves the problem of joint parameter and state estimation conceptually, but, as mentioned above, only rarely can be reduced to a feasible algebraic recursion without any approximation. Our case is an exception.

The following result will appear very useful.

Result (6A): Estimation of the parameters separately from the model state.

If there exists a statistic $X(t)$, a finite dimensional function of the observed data but not of the unknown parameters,

$$X(t) = \mathrm{f}(X(t-1), y(t), u(t)) \tag{7}$$

such that

$$\mathrm{p}(s(t) \mid t; \theta) = \mathrm{p}(s(t) \mid X(t), \theta) \tag{8}$$

then, under natural conditions of control, the unknown parameters θ can be estimated separately from the state $s(t)$ using the relations

$$\mathrm{p}(\theta \mid t) = \frac{\mathrm{p}(y(t) \mid X(t-1), \theta, u(t))\, \mathrm{p}(\theta \mid t-1)}{\mathrm{p}(y(t) \mid t-1; u(t))} \tag{9}$$

where

$$\mathrm{p}(y(t) \mid X(t-1), \theta, u(t)) = \tag{10}$$
$$= \int \mathrm{p}(y(t) \mid s(t-1), u(t), \theta)\, \mathrm{p}(s(t-1) \mid X(t-1), \theta)\, \mathrm{d}s(t-1)$$

$$\mathrm{p}(y(t) \mid t-1; u(t)) = \int \mathrm{p}(y(t) \mid X(t-1), \theta, u(t))\, \mathrm{p}(\theta \mid t-1)\, \mathrm{d}\theta \tag{11}$$

Then, for any t, the joint c.p.d.f. for the estimated parameters and the state can be

determined as follows.

$$\mathrm{p}(s(t), \theta \mid t) = \mathrm{p}(s(t) \mid X(t), \theta)\, \mathrm{p}(\theta \mid t) \tag{12}$$

Proof. There is not much to be proved. The given relations are just applications of the elementary operations (2.1) and (2.2), and of the natural conditions of control (2.5)

$$\mathrm{p}(\theta \mid t-1; u(t)) = \mathrm{p}(\theta \mid t-1)$$

It is clear that the usefulness of the Result (6A) much depends on the existence of the statistic $X(t)$ with the properties (7) and (8), and on the form of the c.p.d.f. (9). In the following paragraph it will be shown that such a statistic exists for the parameters a, b and k_c (and d if required) of the linear process models considered in this paper. As the c.p.d.f. (9) is normal if $e(t)$ is assumed to be normally distributed, the problem can be solved within normal probability distributions and the functional recursion can be reduced to algebraic operations on conditional means and covariances.

6.2. C-filters

Let us consider the process model in the form (4.5)

$$A \begin{bmatrix} y(t) \\ s(t) \end{bmatrix} = H\, s(t-1) + b\, u(t-T_u) + k_x + c\, e(t) \tag{13}$$

where the parameters A, b, c and k_x are defined by (3.66) and (3.67). If the nonparametric matrix H of dimensions $(\partial y + \partial s) \times \partial s$ is considered in the form (4.6)

$$H = \mu \begin{bmatrix} 0 \\ I \end{bmatrix} + \begin{bmatrix} I \\ 0 \end{bmatrix}$$

then (13) can represent both ARMA and Delta models. If $y(t)$ and $u(t)$ are replaced by their increments and if it is set $k_x = 0$ then also incremental models are covered.

For the sake of simplicity only the single-input single-output case will be considered. However, the presented solution can be well extended for the multivariate case, namely thanks to the Result (4 B). (Actually, the multivariate case can be solved by running the same ∂y filters in parallel.)

Let the unknown parameters be

$$\theta = \{\bar{a}, b, k_c\} = \{a_1, a_2, \ldots a_n, b_0, b_1, \ldots b_n, k_c\} \tag{14}$$

In order to see whether the Result (6 A) can be applied we have to investigate the c.p.d.f. (8). From the Result (4 A) we know that, under given assumptions, this c.p.d.f. is normal with the covariance matrix (4.21), which is independent of θ, and with the conditional mean $\hat{s}(t \mid t)$, which envolves according to the difference equation (4.24) and is a function of the unknown parameters (14).

$$\begin{aligned} \hat{s}(t \mid t; \theta) = {} & (\mu I + C(t))\, \hat{s}(t-1 \mid t-1; \theta) - \\ & - (\bar{a} - \bar{c}(t))\, y(t) + (\bar{b} - \bar{c}(t)\, b_0)\, u(t-T_u) + k_s \end{aligned} \tag{15}$$

To determine $\hat{s}(t \mid t; \theta)$ as an explicit function of θ let us find the explicit solution of the difference equation (15).

If the transition matrix $G(t, k)$ is introduced so that

$$G(t, k) = \prod_{j=k+1}^{t} (\mu I + C(j)), \quad k < t \tag{16}$$

$$G(t, t) = I \tag{17}$$

$$G(t, k) = (\mu I + C(t))\, G(t-1, k) \tag{18}$$

then the explicit solution of (15) can be written as follows

$$\hat{s}(t \mid t; \theta) = \hat{s}_0(t \mid t) - \hat{s}_y(t \mid t; \bar{a}) + \hat{s}_u(t \mid t; b) + \hat{s}_k(t \mid t; k_c) \tag{19}$$

$$\hat{s}_0(t \mid t) = G(t, 0)\, \hat{s}(0) \tag{20}$$

$$\hat{s}_y(t \mid t; \bar{a}) = \sum_{k=1}^{t} G(t, k)\, (\bar{a} - \tilde{c}(k))\, y(k) \tag{21}$$

$$\hat{s}_u(t \mid t; b) = \sum_{k=1}^{t} G(t, k)\, (\bar{b} - \tilde{c}(k)\, b_0)\, u(k) \tag{22}$$

$$\hat{s}_k(t \mid t; k_c) = \sum_{k=1}^{t} G(t, k)\, k_s \tag{23}$$

where $\hat{s}(0)$ is the expected value of the initial state $s(0)$ before the observation of the process starts and, according to (4.28),

$$k_s' = [0, 0, \ldots 0, k_c] \tag{24}$$

Note that, in case of an incremental model, k_c, k_s and thus also $\hat{s}_k(t \mid t; k_c)$ are zero.

It will appear advantageous to express

$$\bar{a} - \tilde{c}(k) = C^*(k)\, a\,, \quad \bar{b} - \tilde{c}(k)\, b_0 = C^*(k)\, b \tag{25}$$

where

$$C^*(k) = [-\tilde{c}(k), I] \tag{26}$$

Note that $C^*(t)$ of dimensions $n \times (n+1)$ is the matrix $C(t)$ (4.25) extended by an additional column the only nonzero entry of which is a one in the last row. Using (25) it is possible to rewrite (21) and (22) as follows.

$$\hat{s}_y(t \mid t; \bar{a}) = Y(t)\, a\,, \quad \hat{s}_u(t \mid t; b) = U(t)\, b \tag{27}$$

where $Y(t)$ and $U(t)$ are rectangular matrices of dimensions $n \times (n+1)$

$$Y(t) = \sum_{k=1}^{t} G(t, k)\, C^*(k)\, y(k), \quad U(t) = \sum_{k=1}^{t} G(t, k)\, C^*(k)\, u(k) \tag{28}$$

and $a' = [1, \bar{a}']$. Hence, the explicit solution (19) of the difference equation (15) can be expressed as the following linear function of the unknown parameters.

$$\hat{s}(t \mid t; \theta) = K_0(t) - Y(t)\, a + U(t)\, b + K(t)\, k_c \tag{29}$$

where $K_0(t)$ and $K(t)$ are vectors.

Applying (17) and (18) it is easy to find that the following recursion holds.

$$Y(t) = (\mu I + C(t))\, Y(t-1) + C^*(t)\, y(t), \qquad Y(0) = 0 \tag{30}$$

$$U(t) = (\mu I + C(t))\, U(t-1) + C^*(t)\, u(t), \qquad U(0) = 0 \tag{31}$$

$$K(t) = (\mu I + C(t))\, K(t-1) + [0, 0, \ldots 0, 1]', \quad K(0) = 0 \tag{32}$$

$$K_0(t) = (\mu I + C(t))\, K_0(t-1) \qquad K_0(0) = \hat{s}(0) \tag{33}$$

Note that each nonzero entry of the matrix added in (30) and (31) is multiplied by the scalar $y(t)$ or $u(t)$, respectively.

If the prior probability distribution of the initial state $s(0)$ is chosen sufficiently flat (large numbers on the diagonal of $D_s(0)$ in (4.30)) it is suitable to choose $\hat{s}(0) = 0$ and to omit the term (33).

In this way the c.p.d.f. (8) has been brought into the desired form with the statistic

$$X(t) = \{Y(t), U(t), K(t)\} \tag{34}$$

Summing up we have the following

Result (6 B): C-filters.

Estimation of the model parameters $\bar{a}$, b and k_c simultaneously with the state $s(t)$ requires to filter the input and the output of the process using filters operating on matrices $Y(t)$ and $U(t)$ according to (30) and (31). If the possible offset k_c is estimated then also the vector $K(t)$ has to be evolved in real time according to (32). The time-varying coefficients $\tilde{c}(t)$ which enter the filtering are generated by the algorithm described in the paragraph 4.3.

PASCAL procedure GFIL

The procedure *CFIL* listed below incorporates one sample of the filtered signal into a statistic of a matrix form according to (30) or (31). The nonstandard types of its parameters are

TYPE *row* = ARRAY [0 .. nmax] OF REAL;
Mat = ARRAY [1 .. nmax] OF *row*;

where nmax is an INTEGER constant chosen with respect to the maximal model order to be considered.

Parameters:

Xf ... type *Mat*; matrix state of the filter (eg. *Y* or *U*) to be updated (according to (30) or (31)).
x ... type REAL; signal sample (eg. *y* or *u*).
c ... type row; vector of coefficients generated by the procedure *CGEN*.
n ... model order.
mti ... model-type indicator, *mti* = 1 for Delta, *mti* = 0 for ARMA.

```
PROCEDURE CFIL(VAR Xf: Mat; x: REAL; c: row; n, mti: INTEGER);
VAR i, j: INTEGER;
BEGIN
FOR j := 0 TO n DO
  BEGIN
  FOR i := 1 TO n - 1 DO
    Xf[i, j] := mti * Xf[i, j] + Xf[i + 1, j] - c[i] * Xf[1, j];
  Xf[n, j] := mti * Xf[n, j] - c[n] * Xf[1, j]
  END;
FOR i := 1 TO n DO
  BEGIN
  Xf[i, i] := Xf[i, i] + x ;
  Xf[i, 0] := Xf[i, 0] - c[i] * x
  END
END;
```

6.3. Real-time estimation

Now we are prepared to solve the problem of estimating the parameters $\bar{a}$, b and k_c simultaneously with the state $s(t)$ in real time according to the Result (6 A). However, since the problem is linear and Gaussian it will be sufficient to operate only on conditional mean values and covariances.

Let us order the set of $2n + 2$ unknown parameters (14) into the vector

$$\theta' = [a_1, a_2, \dots a_n, b_0, b_1, \dots b_n, k_c] \tag{35}$$

and let us order the statistic (34) into the matrix

$$X(t) = [-Y(t), U(t), K(t)] = [-Y^*(t), Z(t)] \tag{36}$$

where $Y^*(t)$ is the first column of the matrix $Y(t)$.

Then from (29) for $K_0(t) = 0$ we have

$$\hat{s}(t \mid t; \theta) = X(t)\begin{bmatrix}1\\ \theta\end{bmatrix} = -Y^*(t) + Z(t)\,\theta \tag{37}$$

Since we are interested in recursive estimation let us assume that the mean value of the uncertain parameter vector (35), conditioned on the input-output data observed up to the sampling period $t - 1$, is given

$$\mathsf{E}[\theta \mid t - 1] = \hat{\theta}(t - 1) \tag{38}$$

and that also its covariance matrix is given in the factorized form

$$\mathrm{Var}\,[\theta \mid t - 1] = \varrho\, L_\theta(t - 1)\, D_\theta(t - 1)\, L'_\theta(t - 1) \tag{39}$$

where the monic LT-matrix $L_\theta(t - 1)$ and the diagonal matrix $D_\theta(t - 1)$ are given numerically while the scalar factor $\varrho = \mathrm{Var}\, e(t)$ does not need to be known. The

mean value (38) and the covariance matrix (39) fully determine the c.p.d.f. $\mathsf{p}(\theta \mid t-1)$ if it is normal.

To be able to update $\mathsf{p}(\theta \mid t-1) = \mathsf{p}(\theta \mid t-1; u(t))$ with respect to a newly observed output $y(t)$ we need to determine the numerator of (9), i.e. the joint c.p.d.f.

$$\mathsf{p}(y(t), \theta \mid t-1; u(t)) = \mathsf{p}(y(t) \mid X(t-1), u(t), \theta)\, \mathsf{p}(\theta \mid t-1) \tag{40}$$

From the Result (4 A) it is known that $\mathsf{p}(y(t) \mid X(t-1), u(t), \theta)$ is normal with

$$\mathsf{E}[y(t) \mid t-1; u(t), \theta] = \hat{y}(t \mid t-1; u(t), \theta) = \hat{s}_1(t-1 \mid t-1; \theta) + b_0\, u(t) \tag{41}$$

$$\mathrm{Var}\,[y(t) \mid t-1; u(t), \theta] = \varrho\, d_y(t) \tag{42}$$

where $d_y(t)$ does not depend on the unknown parameters and is supplied by the procedure *CGEN* simultaneously with the coefficients $\tilde{c}(t)$ required for updating the statistic $X(t-1)$.

The conditional mean value $\hat{s}_1(t-1 \mid t-1; u(t), \theta)$ is given by the first row of (37) for the time index $t-1$

$$\hat{s}_1(t-1 \mid t-1; \theta) = X_1(t-1)\begin{bmatrix}1\\ \theta\end{bmatrix} = -Y_1^*(t-1) + Z_1(t-1)\,\theta + b_0\, u(t)$$

where $X_1(t-1)$ and $Z_1(t-1)$ are the first rows of the matrices $X(t-1)$ and $Z(t-1)$, and $Y_1^*(t-1)$ is the first entry of the column vector $Y^*(t-1)$. Thus, (41) can be written

$$\hat{y}(t \mid t-1; u(t), \theta) = -Y_1^*(t-1) + Z_1(t-1)\,\theta + b_0\, u(t)$$

or more concisely

$$\hat{y}(t \mid t-1; u(t), \theta) = -Y_1^*(t-1) + z(t)\,\theta \tag{43}$$

where $z(t)$ is a row-vector with the entries

$$z_j(t) = Z_{1,j}(t-1) \quad \text{for} \quad j \neq n+1 \tag{44}$$

$$z_{n+1}(t) = Z_{1,n+1}(t-1) + u(t) \tag{44'}$$

Now the required joint c.p.d.f. (40) can be determined using the Result (2 B) slightly modified in the following way. Replace in (2.29) to (2.45) y by θ and x by $y(t)$, exchange their positions in (2.41) and (2.42) and, of course, correspondingly also their covariances. In this way it is obtained

$$\mathsf{E}[y(t) \mid t-1; u(t)] = -Y_1^*(t-1) + z(t)\,\hat{\theta}(t-1) \tag{45}$$

and, when we introduce the row-vector

$$f(t) = z(t)\, L_\theta(t-1) \tag{46}$$

the joint covariance matrix is

$$\mathrm{Var}\left[\begin{matrix}y(t)\\ \theta\end{matrix}\,\middle|\, t-1; u(t)\right] = \tag{47}$$

$$= \varrho \begin{bmatrix} 1 & f(t) \\ 0 & L_\theta(t-1) \end{bmatrix} \begin{bmatrix} d_y(t) & 0 \\ 0 & D_\theta(t-1) \end{bmatrix} \begin{bmatrix} 1 & 0 \\ f'(t) & L'_\theta(t-1) \end{bmatrix}$$

To perform the conditioning (9) it is sufficient, according to the Result (2 A), to modify the factorization of the joint covariance matrix (47), in the following way

$$\begin{bmatrix} 1 & f(t) \\ 0 & L_\theta(t-1) \end{bmatrix} \begin{bmatrix} d_y(t) & 0 \\ 0 & D_\theta(t-1) \end{bmatrix} \begin{bmatrix} 1 & 0 \\ f'(t) & L'_\theta(t-1) \end{bmatrix} = \tag{48}$$

$$= \begin{bmatrix} 1 & 0 \\ g_\theta(t) & L_\theta(t) \end{bmatrix} \begin{bmatrix} \tilde{d}_y(t) & 0 \\ 0 & D_\theta(t) \end{bmatrix} \begin{bmatrix} 1 & g'_\theta(t) \\ 0 & L'_\theta(t) \end{bmatrix}$$

The algorithm which performs this modification will be described below. Then the Result (2 A) gives

$$\hat{\theta}(t) = \hat{\theta}(t-1) + g_\theta(t)\,\hat{\varepsilon}(t) \tag{49}$$

where $\hat{\varepsilon}(t)$ is the prediction error

$$\hat{\varepsilon}(t) = y(t) - \hat{y}(t \mid t-1; u(t)) \tag{50}$$

$$\hat{\varepsilon}(t) = y(t) + Y_1^*(t) - z(t)\,\hat{\theta}(t-1)$$

At the same time it is obtained

$$\mathrm{Var}\,[\theta \mid t] = \varrho\, L_\theta(t)\, D_\theta(t)\, L'_\theta(t) \tag{51}$$

$$\mathrm{Var}\,[y(t) \mid t-1; u(t)] = \varrho\, \tilde{d}_y(t) \tag{52}$$

This, together with the updating the statistic (36) according to (30), (31) and (32), solves our problem of recursive parameter estimation.

It remains to determine the joint probability distribution for the parameters θ and the state $s(t)$. A straightforward application of the Result (2 B) gives

$$\mathsf{E}[s(t) \mid t] = X(t) \begin{bmatrix} 1 \\ \hat{\theta}(t) \end{bmatrix} = -Y^*(t) + Z(t)\,\hat{\theta}(t) \tag{53}$$

$$\mathrm{Var} \begin{bmatrix} \theta \\ s(t) \end{bmatrix} \Big| t \Big] = \varrho \begin{bmatrix} L_\theta(t) & 0 \\ Z(t)\, L_s(t) & L_s(t) \end{bmatrix} \begin{bmatrix} D_\theta(t) & 0 \\ 0 & D_s(t) \end{bmatrix} \begin{bmatrix} L'_\theta(t) & L'_\theta(t)\, Z'(t) \\ 0 & L'_s(t) \end{bmatrix} \tag{54}$$

The procedure can be summarized as follows.

Result (6 C): Simultaneous parameter and state estimation.

Suppose that the observed process can be described by the model (13) with given parameters c but unknown parameters a, b and k_c. The unknown parameters and the state $s(t)$ can be estimated in real time, under the earlier stated assumptions with no approximation involved, using the following procedure.

Given the statistics L_s, D_s, X, L_θ, D_θ and $\hat{\theta}$ from the previous step, say for the time index $t-1$, proceed as follows.

1. Using the given parameters c apply the procedure CGEN to update L_s and D_s, and generate $\tilde{c}$ and d_y.

2. After a new input $u(t)$ is applied and the succeeding output $y(t)$ is observed compose the row vector z acording to (44), place $y(t) - Y_1^*(t-1)$ into z_0 and using d_y apply the below described procedure *LDFIL* to update L_θ, D_θ and the parameter estimates $\hat{\theta}$.
3. Using $\tilde{c}$ update the statistic X applying the procedure *CFIL* to its components Y and U. If also k_c is estimated update the last colmn of X according to (32).
4. Multiply the statistic X by the parameter estimates $\hat{\theta}$ according to (53) to obtain the estimate of the state $s(t)$.
5. Repeat 1.

The uncertainty of the estimates is characterized by the covariance matrix (54). The variance of the one-step-ahead prediction of the output $y(t)$ is given by (52). The variance ϱ of the white-noise component of the model (13) does not need to be known if only the estimates (the conditional means) are of interest. However, it is assumed to be constant.

Remark (*a*). Into $\hat{\theta}(0)$, $D_\theta(0)$ and $L_\theta(0)$ the prior information about the possible values of the estimated parameters can be incorporated. This can be very useful when starting a selftuning control.

Remark (*b*). The recursive estimation of the parameters θ derived in this paragraph is, actually, nothing else than a least-square regression where $z(t)$ is the regressor, $y(t) + Y_1^*(t-1)$ is the regressant and $d_y(t)$ is the weight assigned to the minimized square of the residual.

LD filter

The modification of the matrix factorization (48) can be performed using dyadic reduction. The left-hand side of (48) can be considered as a weighted sum of symmetric dyads. In the scheme Fig. 13 each row represents one dyad and its weight placed in the most left column. N is the number of estimated parameters, in our case (if also k_c is estimated) $N = 2n + 2$. The empty spaces are zeros which do not enter the computation.

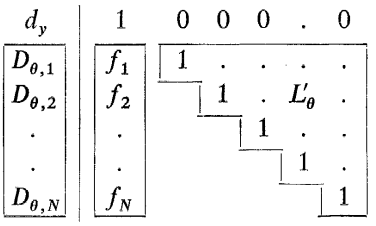

Fig. 13. Scheme illustrating the application of dyadic reduction to the modification of the matrix factorization (48).

The goal of the algorithm, to achieve the desired form of the right-hand side of (48), is to zero all entries f_i lying under the one which is the only nonzero entry

in the first row when the algorithm starts. Hence, the first row is used as reducing in the dyadic reduction. However, not to destroy the upper triangular form of the matrix L_θ it is necessary to start with f_N and proceed backwards for $i = N, N-1, \ldots 1$ until the situation depicted in Fig. 14 is reached. The vector g_θ, required for the updating of the parameter estimates according to (49), appears in the first row as indicated.

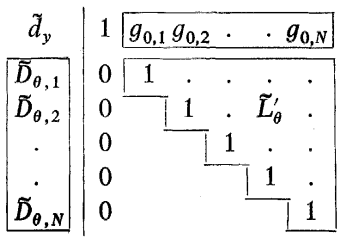

Fig. 14. Reduction of f generates g_θ in the first row.

PASCAL procedure LDFIL

The below listed procedure performs one step of least-square regression with weighting of the minimized squares of residuals. It calculates the vector f according to (46), performs the modification of the matrix factorization (48) and updates the parameter estimates. At the same time it updates L_θ and D_θ in (51) and generates $\tilde{d}_y$ for (52). The nonstandard types of its parameters are

```
TYPE row    = ARRAY [0 .. Nmax] OF REAL;
     Matrix = ARRAY [0 .. Nmax] OF row;
```

where Nmax is an INTEGER constant chosen with respect to the maximal number of the estimated parameters. Since the procedure makes use of the procedure *DYDR* the type *row* must be common for both of them.

Parameters:

Par ... type VAR *row*; parameter estimates *Par* [i] = θ_i; must be initialized before the first call of the procedure; *Par* [0] not used.

D ... type VAR row; diagonal of D_θ extended by d_y in $D[0]$; procedure modifies $D[0]$ to $\tilde{d}_y$ and therefore $D[0]$ must be restored whenever the procedure is called; *D* must be initialized before the first call of the procedure when the large numbers $D[i]$ for $i = 1 \ldots N$ mean very uncertain prior estimate *Par*.

L ... type VAR *Matrix*; monic upper triangular matrix L'_θ extended by the column with index $j = 0$ where the vector f' can be found, and by the row with index $i = 0$ where the gain vector g'_θ appears; must be initialized before the first call of the procedure including the ones on the main diagonal, $L[i, i] = 1$ for $i = 1 \ldots N$.

z ... type *row*; row of processed data, $z[0]$ regressant, $z[j], j = 1 \ldots N$, regressor.

N ... type INTEGER; number of estimated parameters.

```
PROCEDURE LDFIL(VAR Par, D: row; VAR L: Matrix; z: row; N: INTEGER);
VAR i, j: INTEGER;
    f, e: REAL;
BEGIN
e := z[0];
FOR i := 1 TO N DO
  BEGIN
  e := e - z[i] * Par [i];
  f := z[i];
  FOR j := i + 1 TO N DO f := f + z[j] * L[i, j];
  L[i, 0] := f;
  L[0, i] := 0
  END;
FOR i := N DOWNTO 1 DO DYDR (L[i], L[0], D[i], D[0], 0, i, N);
FOR i := 1 TO N DO Par [i] := Par [i] + L[0, i] * e
END;
```

7. ADAPTIVE AND SELF-TUNING CONTROL

The previous section gave an unswer to the problem of how to predict the output of an observed process one step ahead and at the same time to estimate the state of its linear model if the main parameters of the model are not known. Under the stated assumption the answer is exact. Unfortunately, the problem of optimum control synthesis, minimizing the criterion (5.35) for the control horizon $T > 1$, is much more difficult and the present theory does not yield its exact solution which could be practically realized using the present-day computer technique. This is discussed in the first paragraph of this last section.

Fortunately, there exists an approximative solution of the given problem which is well feasible and appears to be successful in practical applications even when its full theoretical analysis is still lacking. It is the self-tuning control, based on the so-called certainty equivalence hypothesis, discussed in the second paragraph. In this concluding paragraph it will be outlined how the above derived algorithms can be employed to implement such a control practically.

7.1. Problem of dual control

In the previous Section 6 it was derived that, under the lack of knowledge of the parameters θ (6.14), the predictive c.p.d.f. $\mathsf{p}(y(t) \mid t-1; u(t))$ is normal with the mean value (6.45) and the variance (6.52). The inspection of the above results also shows that the statistic

$$(1) \qquad S_y(t-1) = \{X(t-1), \hat{\theta}(t-1), L_\theta(t-1), D_\theta(t-1)\}$$

is sufficient for this c.p.d.f. so that

$$\mathrm{p}(y(t) \mid t-1; u(t)) = \mathrm{p}(y(t) \mid S_y(t-1), u(t)) \tag{2}$$

Hence, all is prepared for the general method of optimum control synthesis according to the Result (5 A). Moreover, as a finite dimensional sufficient statistic (1) exists (it exists also for unknown ϱ) the domain of the Bellman function is of fixed and nongrowing dimension. However, when trying to apply the algorithm of dynamic programming to this case serious difficulties are met already in the second step. The function F_k defined by (5.31) is no more quadratic, neither elementar. Even when it is calculated as a multidimensional table its minimization with respect to $u(k)$ is no simple problem. These are the reasons why such a control – called dual – is unrealistic for practical use.

The problem of dual control was formulated firstly by Feldbaum [18] but could be solved only for the most simple cases. The adjective dual means that such a control modifies the control actions, which would be optimal for the known parameters, so that it introduces perturbations improving the parameter estimation and thus also the future controls.

7.2. Self-tuning control

A natural approximation of the optimal dual control is to replace the unknown parameters by their point estimates and to perform the control synthesis as if the model parameters were known. This is sometimes called certainty equivalence hypothesis. Practically it means to combine the algorithm for parameter and state estimation described in the Result (6 C) with one of the algorithms for optimum control synthesis given in Section 5. However, it also means that the control law has to be redesigned when the parameter estimates are updated. There are two possible ways how to keep the computational load within reasonable proportions.

Receding control horizon. The control horizon for which the control law is designed is chosen only slightly longer than is the settling time of the process, which is usually a priori known with sufficient accuracy. However, then the settling of the final state must be heavily weighted in the criterion in order to ensure the stability of control.

Iterative optimization. Usually better and simpler is to proceed as follows. For $T \to \infty$ the parameters of the Bellman function, which determines the optimal control for the current sampling period, converge to constants (except m_w and m_{0w} in (5.44) or m_p and m_{pp} in (5.71) in case of program control.) To find this asymptotic Bellman function it appears to be sufficient to perform in each control step only a very limited numer of optimization steps (just one is usually sufficient) using the algorithm of dynamic programming. Such a control strategy is called IST (Iterations Spread in Time) in [1] where it is discussed in more detail.

When starting the self-tunig control it is advisable to incorporate into the chosen

prior probability distributions as much prior information about the controlled process as possible. The way how to proceed can be found in [19].

Time-varying processes

The control problem considered in this paper has been formulated and solved as control of a process which can be described by a linear stochastic model with unknown but constant parameters. The resulting controller has the ability to accumulate the information about the controlled process carried by the observed data and to use it for automatic tuning of its control law. This removes the by-hand tuning of standardly used controllers.

In many industrial applications the properties of the controlled process vary in time and it is required that the adaptive controller be able to track these variations. Then it is necessary to suppress the old information in order to make space for the new one. The technique of exponential forgetting has been widely used for this purpose. A probabilistic interpretation of this technique can be found in [10]. It explaines the difficulties which may occur when the standard exponential forgetting is applied in a closed control loop. Therefore a new technique of forgetting has been developed [20, 21] which removes these difficulties. The identification procedure derived in Section 6 provides all probabilistic characteristics which are required for the application of these heuristic but rationally based technique also for the models considered in this paper.

When concluding it is appropriate to emphasize that the presented theoretical results cannot be applied mechanically. It is the engineer, the control system designer, who has to think and to apply the theoretical results adequately to his practical problem, considering the assumptions under which the theory was developed. The theory only can help him in keeping his thinking consistent in complex situations and to provide algorithms which make it possible to realize his ideas in a reliable way.

REFERENCES

[18] A. A. Feldbaum: Dual control theory I—IV. Automat. Remote Control *21* (1960), 874—880, 1033—10039, and *22* (1961), 1—12, 109—121.

[19] M. Kárný: Quantification of prior knowledge about global characteristics of linear normal models. Kybernetika *20* (1984), 5, 376—385.

[20] R. Kulhavý and M. Kárný: Tracking of slowly varying parameters by directional forgetting. Proc. 9th IFAC Congress, Budapest, 1984, Vol. X, 81—85.

[21] R. Kulhavý: Restricted exponential forgetting in real-time identification. Preprints 7th IFAC/IFORS Symposium on Identification and System Parameter Estimation. York, 1985.