Least squares and maximum likelihood estimation of Markov-parameters

Vaessen, Jos

Award date:
1983

DEPARTMENT OF ELECTRICAL ENGINEERING

EINDHOVEN UNIVERSITY OF TECHNOLOGY

Group Measurement and Control

LEAST SQUARES AND MAXIMUM LIKELIHOOD ESTIMATION OF MARKOV-PARAMETERS

by Jos Vaessen

This report is submitted in partial fulfillment of the requirements for the degree of electrical engineer (M.Sc.) at the Eindhoven University of Technology.
The work was carried out from Nov. 1982 until Aug. 1983
in charge of Prof. dr. P. Eykhoff
under supervision of ir. A.C.P.M. Backx (Philips)
dr. ir. A.A.H. Damen
and dr. ir. A.J.W. van den Boom

SUMMARY

A possibility for modelling a MIMO-system is a representation based on the multivariable impulse responses: the Markov-parameters.
In this report several estimation schemes are derived for these parameteres. The difference in these schemes is either the numerical implementation or the more detailed use of the noise model.
All estimation procedures are based on the Least Squares principle and the Maximum Likelihood principle.
For the estimation of the parameters of a noise model a non-linear Least-Squares function has to be minimized.To accomplish this, three non-iterative procedures have been derived, approximately arriving at the minimum, and three iterative procedures have been developed, which are theoretically reaching the exact minimum.
All these procedures are compared theoretically and many simulations have been performed to test their properties such as accuracy and efficiency.

CONTENTS

CHAPTER 1. INTRODUCTION

In the world around us, we notice an increasing interest in describing complicated systems. Examples are biomedical, economic, technical systems and many others that may possess a high degree of complexity.
All those systems have the following in common: They are multivariable and dynamical. In this case, 'multivariable' reflects the property that they have more than one input and more than one output, and that the inputs may influence more than one output at a time;
such a system is called a MIMO-system (Multi Input,Multi Output).
It can be represented by a block diagram as is shown in fig. 1.-1.

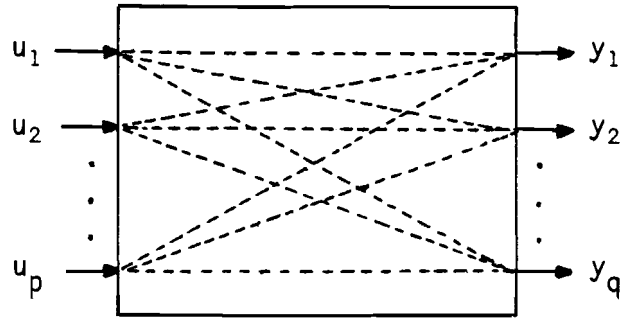


fig. 1.-1 block diagram of a MIMO-system

Because of the internal complexity of many systems it is often necessary to look at them in a 'black box' approach: This means that we are not interested in the internal behaviour but only in the relations between the input variables $u_1,u_2,\ldots,u_p$, that we assume to represent the excitation of the system, and the output variables $y_1,y_2,\ldots y_q$ representing the response of the system to the excitation. These relations can be formulated in several ways, that result in several kinds of models.
The problem of system identification is to find a mathematical description or model that describes the behaviour of a system. Information that can be used to solve this problem is a priori information of the system and measurements of the input- and output variables.
The choice of a certain model is dependent on the intended use but, once we have chosen for some model that is able to represent the system, the unknown parameters of that model have to be estimated, based on noisy ob-

servations of the behaviour of the system.

In engineering situations, the results of system identification are mainly used for control purposes, which justifies the choice of a 'black box' model or input-output model, but it also restricts the choice of models in the sense that a model suited for system identification will not always be applicable to controller design.
In practice we have to cope with the phenomenon of noise that consists of measurement-noise and of errors due to a discrepancy between the model and the actual process.
For control purposes often not only a description of the process is needed, but also some information about the character of the noise. This is one of the reasons why we not only pay attention to the identification of the process but also to the noise.

In chapter 2 some possibilities are described for modelling a MIMO-system and extra attention is given to the model that will be used in this report. In chapter 3 some estimation methods are described and these methods are extended in chapter 4 and 5 to be able to cope with more complicated structures of the noise model.
Chapter 6 gives a short description of a method to change the chosen model into another description that is better suited for control purposes.
In chapter 7 some results are given of simulations.

CHAPTER 2. MODEL DESCRIPTION

In the first section of this chapter attention will be paid to the modelling of MIMO-systems. From the set of models described, a model is chosen which gives an approximate description of a multivariable system. This approximate description however, will appear to be well suited for estimation purposes.

2.1 Possibilities for modelling a MIMO-system

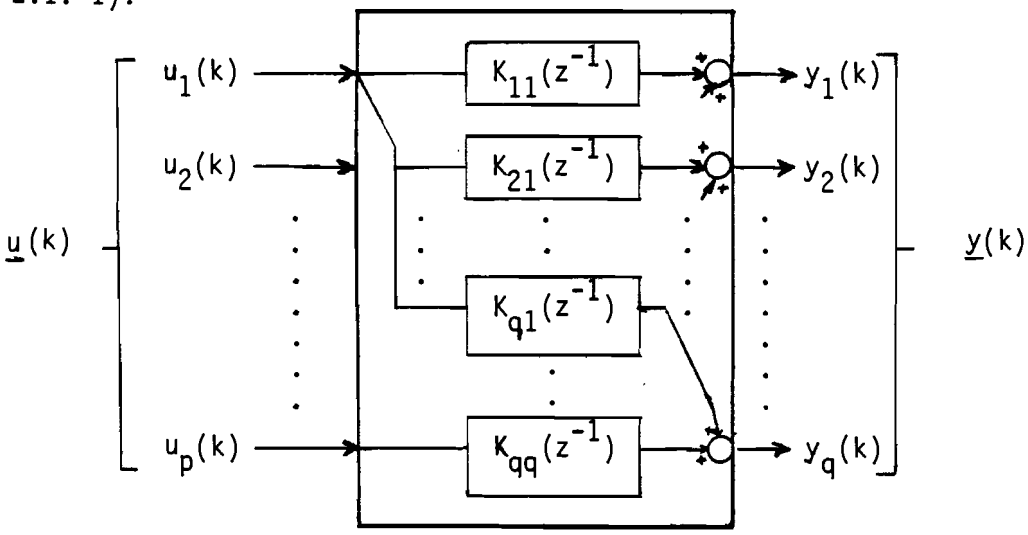
Analogous to SISO-systems, (Single Input Single Output), it is possible to present several kinds of models of multivariable systems. But, before we start our review, we will make some assumptions to limit the number of systems that we shall take into account. From now on, we only consider: linear, time-invariant, discrete-time and stable systems of finite dimension.
These limitations are fairly heavy compared to the processes, we deal with in practice. Nevertheless, it is often possible to linearize a real system in the neighbourhood of a working point and approximate it by a linear system. Time-invariance is not allways required, since the model can be adapted when the process is observed in an on-line manner.
The restriction to stable and finite dimensional systems is quite trivial; and the same for discrete-time systems, because we merely deal with sampled in- and output signals.

The way of modelling, which is best known, is the one using a transfer function:

$$\underline{y}(k) = K(z^{-1}) \cdot \underline{u}(k) \qquad \text{eq. 2.1.(1)}$$

where $\underline{y}^T(k)$ is the output vector $(y_1(k), \ldots, y_q(k))$ and $\underline{u}^T(k)$ the input vector $(u_1(k), \ldots, u_p(k))$, at time instant k, of a system with p inputs and q outputs.

$K(z^{-1})$ is a rational matrix of variable z of dimensions q×p.This means that the relation between every input and every output can be described by a single transfer function as we know it from the SISO-case (see fig. 2.1.-1).



Thus:

$$K(z^{-1}) = \begin{bmatrix} K_{11}(z^{-1}) & \cdots & K_{1p}(z^{-1}) \\ \vdots & & \vdots \\ K_{q1}(z^{-1}) & \cdots & K_{qp}(z^{-1}) \end{bmatrix}$$

in wich $K_{ij}(z^{-1})$ is the transfer function from input j to output i.

Another possibility is a state space description. In a state space model, state variables $\underline{x}$ are introduced that contain information about the past of the system. Using this information it is possible to construct the output signal $\underline{y}(k)$, based only on the input $\underline{u}(k)$ and the state of the system at time instant k:$\underline{x}(k)$.
By means of these variables we can represent a system in the following way:

$$\underline{x}\,(k+1) = A \,.\, \underline{x}\,(k) + B \,.\, \underline{u}\,(k) \qquad \text{eq. 2.1.(2a)}$$

$$\underline{y}\,(k) = C \,.\, \underline{x}\,(k) + D \,.\, \underline{u}\,(k) \qquad \text{eq. 2.1.(2b)}$$

where $\underline{x}(k)$: the (n×1)- state vector $\underline{x}$ at time instant k
$\underline{u}(k)$: the (p×1)- input vector $\underline{u}$ at time instant k
$\underline{y}(k)$: the (q×1)- output vector $\underline{y}$ at time instant k
A : (n×n)- system matrix, where n is called the dimension of the state space or dimension of the system
B : (n×p)- distribution matrix
C : (q×n)- output matrix
D : (q×p)- input- output matrix
p,q : resp. the number of inputs and outputs of the system

The dynamical behaviour of this system is defined completely by the set of matrices {A,B,C,D} which is called a realization of the system. In this report we will restrict ourselves to 'controllable' and 'observable' systems. A process is called 'controllable' when it is possible to find an (unconstrained) control vector which brings the system from any initial state to any specified final state in finite time.
A sufficient condition for controllability is that the controllability matrix Δ :

$$\Delta \triangleq \left[B \mid AB \mid A^2B \mid \ldots \mid A^{r-1}B \right] \qquad \text{eq. 2.1.(3)}$$

has rank n.
A process is called 'observable' when it is possible to determine the state of that process from the measurements of the output. A sufficient condition is that the observability matrix Γ :

$$\Gamma \triangleq \begin{bmatrix} C \\ CA \\ CA^2 \\ \cdot \\ \cdot \\ CA^{r-1} \end{bmatrix} \qquad \text{eq. 2.1.(4)}$$

has rank n.

A third model that may be used in describing the behaviour of a MIMO- sytem is the so- called Hankel- model, which is based on the impulse responses of the system.
This model can be derived from the state space model as follows:
Substitute eq. 2.1.(2a) in eq. 2.1.(2b), which results in:

$$\underline{y}(k) = CA^{k}\underline{x}(0) + \sum_{i=1}^{k} C.A^{i-1}.B.\underline{u}(k-i) + D.\underline{u}(k) \qquad \text{eq. 2.1.(5)}$$

where $\underline{x}(0)$ is the initial state of the system.
Or with $\underline{x}(0) = \Delta\,[r].\beta_0$; where $\Delta\,[r]$ is the controllability matrix with rank n: $\Delta\,[r] = [B|AB|\ldots|A^{r-1}B]$,.
r is an integer such that $\Delta\,[r]$ has rank n

$$\underline{y}(k) = CA^{k}\Delta[r].\underline{\beta}_{r} + \sum_{i=1}^{k} C.A^{i-1}.B.\underline{u}(k-i) + D.\underline{u}(k) \qquad \text{eq. 2.1.(6)}$$

In eq. 2.1.(6) we can distinguish the matrix products $C.A^{i}.B$, which are in fact the multivariable impulse responses of our system. The part of this equation, containing the summation sign, may be regarded as a convolution sum of impulse responses and all input signals since time instant zero. These multivariable impulse responses are called Markov-parameters, which are defined as:

Markov-parameter at time instant i:

$$M(i) = \begin{cases} 0 & , i<0 \\ D & , i=0 \\ C.A^{i-1}.B & , i>0 \end{cases} \qquad \text{eq. 2.1.(7)}$$

An example of a series of Markov-parameters of a first-order SISO-system can be found in fig. 2.1.-2.

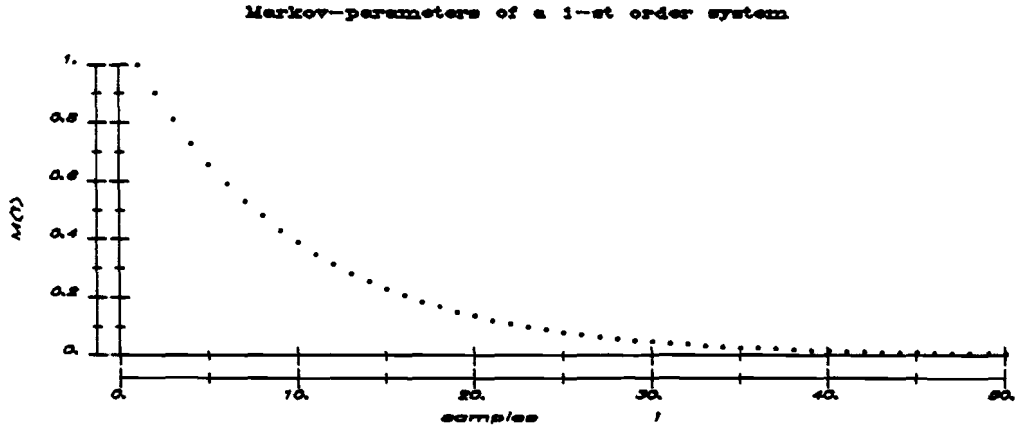


fig. 2.1.-2 Markov-parameters of a 1^{st}-order system

Using the definition of Markov- parameters, eq. 2.1.(6) can be rewritten as:

$$\underline{y}(k) = [CA^{k}B|CA^{k+1}B|\ldots|CA^{k+2-1}B]\underline{\beta}_{r} + \sum_{i=0}^{k} M(i)\,\underline{u}(k-i) \qquad \text{eq. 2.1.(8)}$$

Or in matrix notation:

$$\underline{Y}_{m} = \begin{bmatrix} \underline{y}(0) \\ \underline{y}(1) \\ \cdot \\ \cdot \\ \cdot \\ \underline{y}(m) \end{bmatrix} = \begin{bmatrix} M(0) & 0 & \cdot & \cdot & \cdot & 0 \\ M(1) & M(0) & \cdot & & \phi & \cdot \\ \cdot & \cdot & & \cdot & & \cdot \\ \cdot & \cdot & & & \cdot & \cdot \\ \cdot & \cdot & & & & \cdot \\ M(m) & M(m-1) & \cdot & \cdot & \cdot & M(0) \end{bmatrix} \cdot \begin{bmatrix} \underline{u}(0) \\ \underline{u}(1) \\ \cdot \\ \cdot \\ \cdot \\ \underline{u}(m) \end{bmatrix} + \begin{bmatrix} M(1) & \cdot & \cdot & \cdot & M(r) \\ \cdot & & & & \cdot \\ \cdot & & & & \cdot \\ \cdot & & & & \cdot \\ M(m+1) & \cdot & \cdot & \cdot & M(m+r) \end{bmatrix} \cdot \underline{\beta}_{r}$$

eq. 2.1.(9)

where the index m points to the sample moment under conderation.

Eq. 2.1.(9) can be abbreviated to:

$$\underline{Y}_m = T(o,m) \cdot \underline{U}_m + H(r,\infty) \cdot \underline{\beta}_r \qquad \text{eq. 2.1.(10)}$$

where $\begin{bmatrix} M(0) & . & . & 0 \\ . & . & \phi & . \\ . & & . & . \\ M(m) & . & . & M(0) \end{bmatrix} \triangleq T(0,m)$ and $\begin{bmatrix} M(1) & . & . & M(r) \\ . & & & . \\ . & & & . \\ M(m+1) & . & . & M(m+r) \end{bmatrix} \triangleq H(m,r)$

Matrix T(0,m) is called the Generalized Toeplitz matrix and H(m,r) the Generalized Hankel matrix.
When we substitute the expression $M(i) = C.A^{i-1}.B$ in the Hankel matrix, we find a relation between the Hankel model and the state space model which makes it possible to find a state space description from a Hankel model using impulse responses:

$$\begin{bmatrix} M(1) & . & . & M(r) \\ . & & & . \\ . & & & . \\ M(m+1) & . & . & M(m+r) \end{bmatrix} = \begin{bmatrix} CB & CAB & . & CA^{r-1}.B \\ . & & & . \\ . & & & . \\ CA^mB & . & . & CA^{m+r-1}.B \end{bmatrix} \qquad \text{eq. 2.1.(11)}$$

From eq. 2.1.(6)-eq. 2.1.(9) we see that this matrix is built up as follows:

$$\begin{bmatrix} CB & CAB & . & . & CA^{r-1}.B \\ CAB & & & & . \\ . & & & & . \\ . & & & & . \\ CA^mB & . & . & . & CA^{m+r-1}.B \end{bmatrix} = \begin{bmatrix} C \\ CA \\ . \\ . \\ CA^m \end{bmatrix} \cdot \begin{bmatrix} B & AB & A^2B & . & . & A^{r-1}B \end{bmatrix} = \Gamma[m].\Delta[r]$$

eq. 2.1.(12)

which can be seen as the product of respectively the extended observa-

bility matrix and the extended controllability matrix of the system {A,B,C} (cf. eq. 2.1.(3) and eq. 2.1.(4)).
Although the matrices at the right hand side of eq. 2.1.(12) have larger dimensions, they still have a rank equal to the rank of the system because we have restricted ourselves to observable and controllable systems.
Therefore, when we are able to find the Markov- parameters of a system, it is possible to create a Hankel matrix and then by applying the right decomposition of this Hankel matrix it might be possible to find a realization {A,B,C} of our system. In chapter 6 it will be pointed out that this realization, indeed, can be found.

2.2 A model for the estimation of Markov- parameters

In this section we will derive a model from the Hankel model which is better suited for the estimation of Markov-parameters.
Because of the uniqueness of the impulse responses of a system, Markov-parameters give a unique representation of the system. And thus is this way of modelling very attractive for estimation purposes. Another advantage of this method is that we need no knowledge about the order of the system beforehand; this is in contradistinction to models based on transfer functions or state space models.

When we suppose that the initial conditions of the system are zero i.e. $\underline{x}(0)=\underline{0}$ (so $\underline{\beta}_r=\underline{0}$), eq.2.1.(8) can be rewritten in matrix notation as follows:

$$\begin{bmatrix} \underline{y}^T(0) \\ \cdot \\ \cdot \\ \cdot \\ \underline{y}^T(m) \end{bmatrix} = \begin{bmatrix} \underline{u}^T(0) & \cdot & \cdot & \cdot & 0 \\ \cdot & \cdot & & \phi & \cdot \\ \cdot & & \cdot & & \cdot \\ \cdot & & & \cdot & \cdot \\ \underline{u}^T(m) & \cdot & \cdot & \cdot & \underline{u}^T(0) \end{bmatrix} \cdot \begin{bmatrix} M^T(0) \\ \cdot \\ \cdot \\ \cdot \\ M^T(m) \end{bmatrix} \qquad \text{eq. 2.2.(1)}$$

A great disadvantage of both descriptions, eq. 2.1.(9) and 2.2.(1) is the length of the impulse responses which have to be taken into account. In eq. 2.2.(1) m+2 Markov-parameters are unknown and this number increases when more samples become available.
From the equation above, it is easily seen that we have as many equations as unknown parameters. This implies that we don't have any redundancy in our information, so we are not able to eliminate noise influences, which are present in the measured data.
The nature of impulse responses makes it reasonable to suppose that after e.g. k samples of such an impulse response the influence is so small that we may neglect its contribution to the convolution sum (see fig. 2.2.-1).

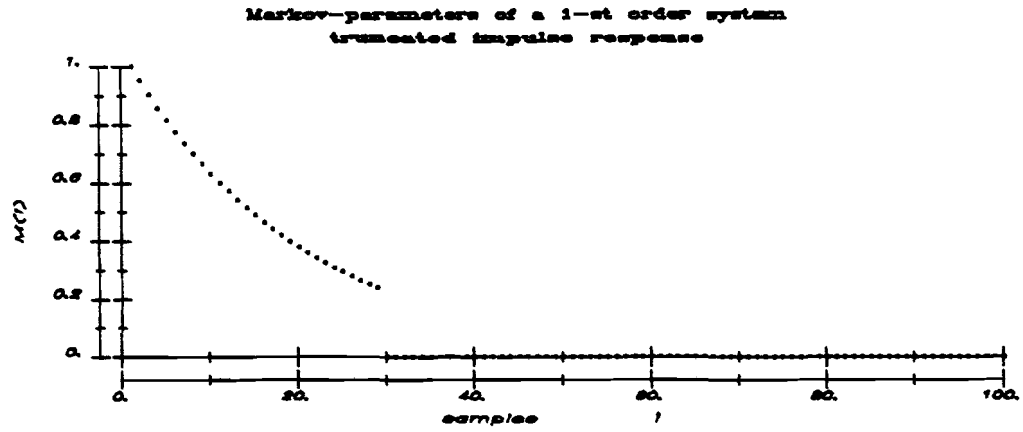


fig. 2.2.-1 truncated impulse response

Using this property of the impulse response eq. 2.2.(1) can be written as:

$$\begin{bmatrix} \underline{y}^T(0) \\ \vdots \\ \underline{y}^T(m) \end{bmatrix} = \begin{bmatrix} \underline{u}^T(0) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ \underline{u}^T(m) & \cdots & \underline{u}^T(m-k) \end{bmatrix} \cdot \begin{bmatrix} M^T(0) \\ \vdots \\ M^T(k) \end{bmatrix} + \underbrace{\begin{bmatrix} 0 & \cdots & 0 \\ \underline{u}^T(0) & \ddots & \vdots \\ \vdots & \ddots & \vdots \\ \underline{u}^T(m-k-1) & \cdots & \underline{u}^T(0) \end{bmatrix} \cdot \begin{bmatrix} M^T(k+1) \\ \vdots \\ M^T(m) \end{bmatrix}}_{(1)} \qquad \text{eq. 2.2.(2)}$$

In the equation above, (1) represents the influence of what is called the 'tail' of the Markov-parameters, so the part of the impulse response that is negligibly small.

From now on, we assume that the contribution of the 'tail' and the part caused bij the initial state is so small that it can be interpreted as a negligible part of the noise at the output.
This means that we must take care that we take into account enough Markov-parameters and k is chosen big enough to assure that the contribution of the tail can be omitted.

Using the assumptions, stated above, eq. 2.2.(2) becomes:

$$\begin{bmatrix} \underline{y}^T(0) \\ \vdots \\ \underline{y}^T(m) \end{bmatrix} = \begin{bmatrix} \underline{u}^T(0) & \cdots & \phi \\ \vdots & \ddots & \\ \underline{u}^T(m) & \cdots & \underline{u}^T(m-k) \end{bmatrix} \cdot \begin{bmatrix} M^T(0) \\ \vdots \\ M^T(k) \end{bmatrix} + \begin{bmatrix} \underline{\xi}^T(0) \\ \vdots \\ \underline{\xi}^T(m) \end{bmatrix} \qquad \text{eq. 2.2.(3)}$$

in which $\underline{\xi}^T(\ell) = (\xi_1(\ell),\ldots,\xi_q(\ell))$ is additive output noise, that consists of tail influences and influences of the initial state of the system, i.e. highly dependent 'noise'. If we want to take into account measurement noise, we are only able to cope with output noise and input noise that is independent of the input signal and it is reasonable to suppose that these are also part of the additive output noise.

When the tail influence is small enough, the measurement noise, which is supposed to be white noise, has such a large influence that from now on $\underline{\xi}$ is assumed to be white channel independent noise, due to measurement errors.

From eq. 2.2.(3) we learn that we have more input- and output data than unknown parameters, so we are dealing with an estimation problem, which will be treated in chapter 3 and following chapters.

Rewriting eq. 2.2.(3) in matrix notation results in:

$$Y = S_m^T \cdot M_k + \Xi \qquad \text{eq. 2.2.(4)}$$

where: Y : ((m+1) × q)- matrix containing the output data
S_m^T: ((m+1) × (k+1).p)- matrix containing the input- and shifted input data
M_k : ((k+1).q × p)- matrix containing (k+1) Markov- parameters
Ξ : ((m+1) × q)- matrix that contains the additive output noise

This equation gives us the model which we will use for the estimation of the Markov- parameters (fig. 2.2.-2) .

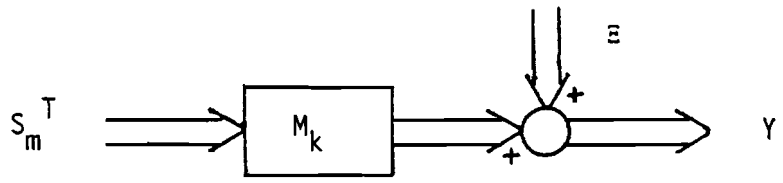


fig. 2.2.-2 model for the estimation of Markov- parameters

2.3 An extension of the model for the estimation of noise parameters

Until now, we have only considered additive output noise which was assumed to be white (see eq. 2.2.(4)). Because this property is almost never met in practice, we shall loosen our restraints to the assumption of additive output noise which may be coloured noise (see fig. 2.3 -1).

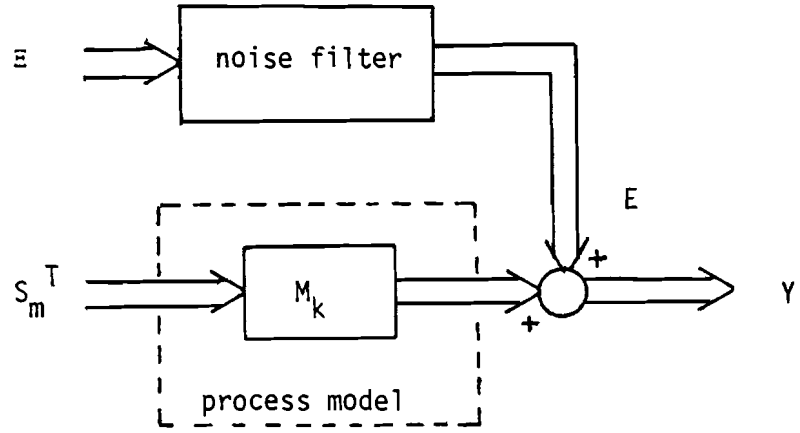


fig. 2.3 -1 extended model

In fig. 2.3.-1 E represents the equation error : $E = Y - S_m^T . M_k$; i.e. that part of the output signal that couldn't be explained by the process model. This equation error may be coloured noise, which can be presumed to be the output of a noise colouring filter with white noise at the input. The white noise is a noise sequence of which the samples are independent in time and place and also independent of the input signal. This means that the following expressions are valid:

$$\text{independent in time: } E\{ \underline{\xi}(i) . \underline{\xi}^T(j) \} = \begin{cases} 0, & \text{if } i \neq j \\ \sigma^2.I, & \text{if } i=j \end{cases} \qquad \text{eq. 2.3.(1)}$$

$$\text{independent in place: } E\{ \xi_k(i) . \xi_n(i) \} = \begin{cases} 0, & \text{if } k \neq n \\ 1, & \text{if } k=n \end{cases} \qquad \text{eq. 2.3.(2)}$$

As a criterion for the estimation of the parameters of our model, we adopt the following:

- try to find as much correlation between the input and out-

put of our system as possible
- in order to accomplish this, correlation between the samples of the equation error or between the equation error and Ξ is allowed.

Having performed this procedure we hope that we have minimized the dependencies that remain in Ξ. If this is not the case, the procedure should be repeated until these dependencies are minimized, and Ξ becomes white gaussian noise in the end.
Using this procedure the estimated noise model gives us a better estimate of the Markov-parameters of the process and the noise model can also be used for control purposes.

As we have seen in section 2.1, there are several possibilities to describe the dependencies in the equation error or the dependencies between the equation error and Ξ.
The most important property of our model must be uniqueness because, we want to find only one specific parameterization of our model, given a set of input and output signals. This implies that a full description in terms of transfer functions is not possible, because this wouldn't give us a unique solution (e.g. adding a pole to the denominator and a zero to the numerator would give us the same in- and output relation, but a different parameterization).
What remains is a noise model using either the Markov-parameters (a Moving Average (MA) model), which, in fact, only describes the numerator of the transfer function or an Auto-Regressive (AR) model that gives a description according to the denominator of the transfer function (see also appendix A).
The first model describes a relation between the input signal of the noise model and the output signal and this causes a problem, for we don't know the input signal beforehand. This implies that we must 'estimate' the input signal Ξ and correct it later.
The second model doesn't give rise to such a problem because we try to find correlation between the samples of the equation error only.
So we have to know, only, the equation error E. For this reason, we have chosen for an Auto-Regressive model for the noise filter.

In the following part of this section this model will be developed.
As has been mentioned before, for an Auto-Regressive description of the noise model we need to find correlations between the samples of the output signal E of the noise filter.
This can be regarded as finding the auto-regressive parameter matrices A(1), A(2),...,A(r) that fulfil the following equation:

$$\underline{e}(\ell) = A(1).\underline{e}(\ell-1) + A(2).\underline{e}(\ell-2) + ..+ A(r).\underline{e}(\ell-r) + \underline{\xi}(\ell) \qquad \text{eq. 2.3.(3)}$$

where $\underline{e}(\ell)$: is a (q×1)-vector that stands for the equation error at sample moment ℓ

A(i): is a (q×q)-matrix, representing the dependency between $\underline{e}(\ell)$ and $\underline{e}(\ell-1)$

r : is the number of auto-regressive parameters that we take into account, which means that we neglect the contribution of $\underline{e}(\ell-i)$ to $\underline{e}(\ell)$ for i>r. This neglection has an effect which is similar to the neglection of the 'tail' of the Markov-parameters as described in 2.2.

Transposing eq. 2.3.(3) and rewriting it in matrix notation results in:

$$E = \begin{bmatrix} \underline{e}^T(\ell) \\ \vdots \\ \underline{e}^T(\ell+m) \end{bmatrix} = \begin{bmatrix} \underline{e}^T(\ell-1) & \cdots & \underline{e}^T(\ell-r) \\ \vdots & & \vdots \\ \underline{e}^T(\ell+m-1) & \cdots & \underline{e}^T(\ell+m-r) \end{bmatrix} \cdot \begin{bmatrix} A(1) \\ \vdots \\ A(r) \end{bmatrix} = H_m^T . A_r$$

eq. 2.3.(4)

with:

$$H_m^T = \begin{bmatrix} \underline{e}^T(\ell-1) & \cdots & \underline{e}^T(\ell-r) \\ \vdots & & \vdots \\ \underline{e}^T(\ell+m-1) & \cdots & \underline{e}^T(\ell+m-r) \end{bmatrix} \quad \text{and} \quad A_r = \begin{bmatrix} A(1) \\ \vdots \\ A(r) \end{bmatrix}$$

Comparing this equation with eq. 2.2.(3) of section 2.2, we see immediately a great similarity between them. Suppose we start observing our system at time instant ℓ-k. Then it is possible to describe the model (fig. 2.3.-2) in the following way.

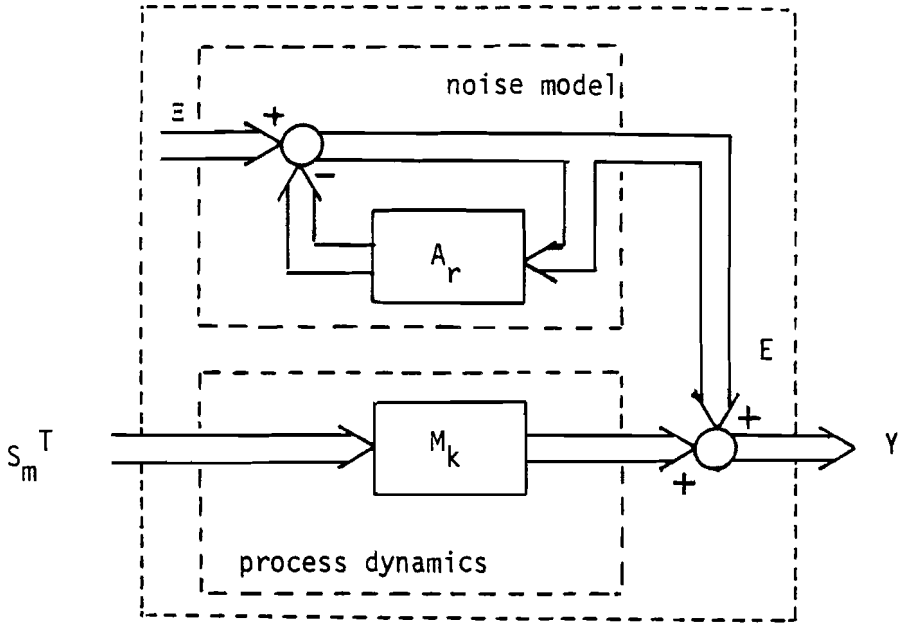


fig. 2.3.-2 an extended model for the estimation of Markov- parameters and AR- noise parameters

$$Y = S_m^T \cdot M_k + E \qquad \text{eq. 2.3.(5)}$$

$$E = H_m^T \cdot A_r + \Xi \qquad \text{eq. 2.3.(6)}$$

Or combining these two equations:

$$Y = S_m^T \cdot M_k + H_m^T \cdot A_r + \Xi \qquad \text{eq. 2.3.(7)}$$

$$Y = (S_m^T \mid H_m^T) \cdot \left[\frac{M_k}{A_r}\right] + \Xi \qquad \text{eq. 2.3.(8)}$$

$$\begin{bmatrix} \underline{y}^T(\ell) \\ \vdots \\ \underline{y}^T(\ell+m) \end{bmatrix} = \left[\begin{array}{ccc|ccc} \underline{u}^T(\ell) & \cdots & \underline{u}^T(\ell-k) & \underline{e}^T(\ell-1) & \cdots & \underline{e}^T(\ell-r) \\ \vdots & & \vdots & \vdots & & \vdots \\ \underline{u}^T(\ell+m) & \cdots & \underline{u}^T(\ell+m-k) & \underline{e}^T(\ell+m-1) & \cdots & \underline{e}^T(\ell+m-r) \end{array}\right] \cdot \begin{bmatrix} M^T(0) \\ \vdots \\ \hline M^T(k) \\ A^T(1) \\ \vdots \\ A^T(r) \end{bmatrix} + \begin{bmatrix} \underline{\xi}^T(\ell) \\ \vdots \\ \underline{\xi}^T(\ell+m) \end{bmatrix}$$

eq. 2.3.(9)

Taking a good look at eq. 2.3.(8) and eq. 2.3.(9), we see that we, in fact, have extended the signal matrix S_m^T with the 'equation error matrix' H_m^T and the same for the parameter matrices M_k and A_r.
This is why this method is called the Extended Matrix Method (EMM).
This model will be of good use to us for estimation purposes, as will be shown in chapter 4.

2.4 The modelling of an offset

In this last section of this chapter, we shall pay some attention to the offset phenomenon which can be found at the input and the output. Suppose that the observed input vector $\underline{u}(\ell)$ differs from the actual input by an offset $\underline{u}_0$, caused by measurement errors, which leads to the next expression:

$$\underline{y}(\ell) = \sum_{i=0}^{k} M(i) \cdot (\underline{u}(\ell-i) + \underline{u}_0) + \underline{e}(\ell) \qquad \text{eq. 2.4.(1)}$$

In this equation, $\underline{e}(\ell)$ may be white noise or coloured noise, which is not important at this moment.
Rewriting eq. 2.4.(1) in matrix notation we find:

$$\begin{bmatrix} \underline{y}^T(\ell) \\ \vdots \\ \underline{y}^T(\ell+m) \end{bmatrix} = \begin{bmatrix} \underline{u}^T(\ell) & \cdots & \underline{u}^T(\ell-k) \\ \vdots & & \vdots \\ \underline{u}^T(\ell+m) & \cdots & \underline{u}^T(\ell+m-k) \end{bmatrix} \cdot \begin{bmatrix} M^T(0) \\ \vdots \\ M^T(k) \end{bmatrix} + \begin{bmatrix} \underline{u}_0^T & \cdots & \underline{u}_0^T \\ \vdots & & \vdots \\ \underline{u}_0^T & \cdots & \underline{u}_0^T \end{bmatrix} \cdot \begin{bmatrix} M^T(0) \\ \vdots \\ M^T(k) \end{bmatrix} + \begin{bmatrix} \underline{e}^T(\ell) \\ \vdots \\ \underline{e}^T(\ell+m) \end{bmatrix}$$

eq. 2.4.(2)

Because the second term of the right hand side of eq. 2.4.(2) is a constant term, it may also be regarded as an offset at the output signal:

$$\begin{bmatrix} \underline{y}_0^T \\ \vdots \\ \underline{y}_0^T \end{bmatrix} = \begin{bmatrix} \underline{u}_0^T & \cdots & \underline{u}_0^T \\ \vdots & & \vdots \\ \underline{u}_0^T & \cdots & \underline{u}_0^T \end{bmatrix} \cdot \begin{bmatrix} M^T(0) \\ \vdots \\ M^T(k) \end{bmatrix} \qquad \text{eq. 2.4.(3)}$$

Substituting eq. 2.4.(3) in eq. 2.4.(2) and rearranging this equation results in:

$$\begin{bmatrix} \underline{y}^T(\ell) \\ \vdots \\ \underline{y}^T(\ell+m) \end{bmatrix} = \left[\begin{array}{c|ccc} 1 & \underline{u}^T(\ell) & \cdots & \underline{u}^T(\ell-k) \\ \vdots & \vdots & & \vdots \\ 1 & \underline{u}^T(\ell+m) & \cdots & \underline{u}^T(\ell+m-k) \end{array}\right] \cdot \left[\begin{array}{c} \underline{y}_0^T \\ \hline M^T(0) \\ \vdots \\ M^T(k) \end{array}\right] + \begin{bmatrix} \underline{e}^T(\ell) \\ \vdots \\ \underline{e}^T(\ell+m) \end{bmatrix} \qquad \text{eq. 2.4.(4)}$$

From this equation it is easily seen that we have used the same idea as in section 2.3, where we used the extended matrix method. So when we extend the signal matrix S_m^T by a column filled with ones and the parameter matrix M_k with the offset vector $\underline{y}_0^T$, it is possible to estimate an offset on both the input- and the output signal. After estimation it is not possible to discover whether the estimated offset has been an offset on the input or on the output signal or whether it has been the sum of offsets on both.

In the following chapters, concerning the estimation schemes, no attention will be paid to the problem dealing with offset, because it is clear from the derivation above that the solution of this problem needs very few changes, which are not essential for the understanding of the estimation techniques. In the final results however this offset approach will be used.

CHAPTER 3. LEAST SQUARES ESTIMATION OF MARKOV-PARAMETERS

3.1 Introduction

This chapter will deal with the problem of the estimation of Markov-parameters. We suppose that we are given a set of input and output data, that may be corrupted by measurement noise, that is not correlated with the input signal and which is assumed to be white (see also eq. 2.3.(1) and eq. 2.3.(2)).The problem is to find a set of Markov-parameters which describes the input-output relation of the system.
The function by which such an estimate can be obtained is called an estimator. To be able to say something about the quality of such an estimate, it is necessary to define when an estimate can be considered to be a good one (Eykhoff (1974)).

Suppose $\hat{\underline{\theta}}$ is an estimator of the unknown parameters $\underline{\theta}$. That is, when evaluated with data values $\underline{z}$, $\hat{\underline{\theta}}(\underline{z})$ provides a specific estimate of $\underline{\theta}$.
It is clear that considerations about the quality of $\hat{\underline{\theta}}$ are based on the errors, that occur in the estimate:

$$\tilde{\underline{\theta}} = \underline{\theta} - \hat{\underline{\theta}}(\underline{z}) \qquad \text{eq. 3.1.(1)}$$

A reasonable property for a good estimator is, of course, that the average value of the error should be zero:

$$E\{\tilde{\underline{\theta}}\} = E\{\underline{\theta} - \hat{\underline{\theta}}(\underline{z})\} = \underline{0} \qquad \text{eq. 3.1.(2)}$$

Or, equivalently, that the expected value of the estimate is equal to the value of the parameter:

$$E\{\hat{\underline{\theta}}(\underline{z})\} = E\{\underline{\theta}\} = \underline{\theta} \qquad \text{eq. 3.1.(3)}$$

Estimators with this property are called unbiased.

An estimator is said to be consistent if for an increasing number of samples k, the probability that the estimate $\hat{\underline{\theta}}(z)$ equals the true value of the parameter $\underline{\theta}$ tends to one:

$$\lim_{k\to\infty} P(|\hat{\underline{\theta}}(\underline{z})-\underline{\theta}| < \varepsilon) = 1 \qquad \text{eq. 3.1.(4)}$$

with ε arbitrary small.
An estimator is called a minimum variance estimator if for all unbiased estimators $\underline{\gamma}$:

$$\text{cov}\{\hat{\underline{\theta}}\} = E\{(\hat{\underline{\theta}}(\underline{z})-\underline{\theta}).(\hat{\underline{\theta}}(\underline{z})-\underline{\theta})^T\} \leq E\{(\underline{\gamma}(\underline{z})-\underline{\theta}).(\underline{\gamma}(\underline{z})-\underline{\theta})^T\} = \text{cov}\{\underline{\gamma}\}$$

eq. 3.1.(5)

This means that such an estimator has the 'smallest' error covariance among all unbiased estimators of $\underline{\theta}$.

More about the properties of estimators in chapter 5, where we shall deal with the Maximim Likelihood method.

3.2 Least squares estimation using an explicit method

The starting point of this section will be the model, eq. 2.2.(4) which we derived in 2.2 :

$$Y = S_m^T . M_k + \Xi \qquad \text{eq. 2.2.(4)}$$

The basic idea of parameter estimation is:
Given a set of input samples (denoted by S_m^T) and a set of output samples (Y), that may be disturbed by white noise (Ξ), that is channel independent, how can we find an estimate $\hat{M}_K$ which is as close as possible to the true parameters M_k.
This $\hat{M}_k$ will satisfy the following expression:

$$\hat{Y} = S_m^T . \hat{M}_k \qquad \text{eq. 3.2.(1)}$$

where $\hat{Y}$ contains the reconstructed output signal.

Our main interest is to find $\hat{M}_k$ in such a way that the error $Y-\hat{Y}$ is, in some chosen sense, as small as possible.

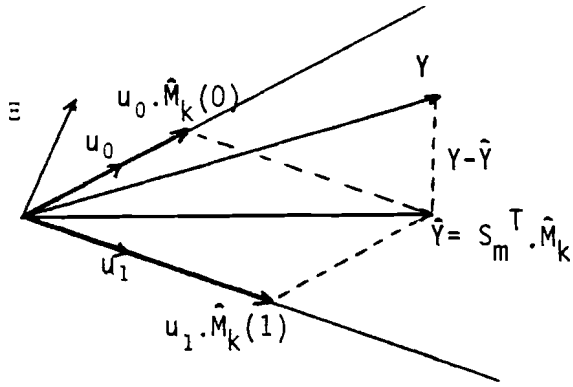


fig. 3.2.-1 geometrical interpretation of LS-estimation

A criterion to minimize $Y-\hat{Y}$ is the criterion of Least Squares (LS). In fig. 3.2.-1 the idea behind the least squares estimation is illustrated for the vector case. Suppose that we have an Euclidean space, which is formed by the input vectors $\underline{u}$, and the parameter vector $\underline{M}_k$ is mapped into this space by S_m^T. Because the output vector $\underline{Y}$ is disturbed by noise, it will not fit in this space. The best approximation $\hat{\underline{Y}}$ to a vector $\underline{Y}$ is the projection of $\underline{Y}$ on the space formed by the input signal. This means that the Euclidean distance between $\underline{Y}$ and $\hat{\underline{Y}}$ is minimized. Therefore, when we were dealing with vectors $\underline{Y}$ and $\hat{\underline{Y}}$, we would have to minimize the loss function V:

$$V = \sum_{i=0}^{m} (y(i)-\hat{y}(i))^2 = (\underline{Y}-\hat{\underline{Y}})^T.(\underline{Y}-\hat{\underline{Y}}) \qquad \text{eq. 3.2.(2)}$$

This is the same as minimization of the sum of all squared error vectors $e(i) = y(i) - \hat{y}(i)$.
But because Y and $\hat{Y}$ are matrices, instead of vectors, we must minimize:

$$V = \text{trace} \{ (Y-\hat{Y})^T.(Y-\hat{Y}) \} \qquad \text{eq. 3.2.(3)}$$

This can easily be verified by writing out eq. 3.2.(3), with $E = Y-\hat{Y}$:

$$V = tr\ (E^T.E) = tr \{ (\underline{e}(\ell),\ldots,\underline{e}(\ell+m)).\begin{bmatrix} \underline{e}^T(\ell) \\ \vdots \\ \underline{e}^T(\ell+m) \end{bmatrix} \}$$

eq. 3.2.(4)

which is identical to:

$$V = tr \{ \sum_{i=0}^{m} \underline{e}(\ell+i).\underline{e}^T(\ell+i) \} = \sum_{j=1}^{q} \sum_{i=0}^{m} e_j^2(\ell+i) \qquad \text{eq. 3.2.(5)}$$

Thus to find a LS-estimate $\hat{M}_{k,LS}$ of the Markov-parameters, it is necessary to minimize the loss function V with respect to $\hat{M}_k$. This minimum is found as follows: (for a table containing formulae of matrix calculus: see Appendix D)

$$\frac{\partial V}{\partial \hat{M}_k} = \frac{\partial}{\partial \hat{M}_k} (tr\{ (Y - S_m^T.\hat{M}_k)^T.(Y - S_m^T.\hat{M}_k) \} \qquad \text{eq. 3.2.(6)}$$

$$= \frac{\partial}{\partial \hat{M}_k} (tr\{ \hat{M}_k^T.S_m.S_m^T.\hat{M}_k \}) - 2.\frac{\partial}{\partial \hat{M}_k} (tr\{ Y^T.S_m^T.\hat{M}_k \})$$

eq. 3.2.(7)

$$= 2.S_m.S_m^T.\hat{M}_k - 2.S_m.Y \qquad \text{eq. 3.2.(8)}$$

The least squares estimator $\hat{M}_{k,LS}$ satisfies $\frac{\partial V}{\partial \hat{M}_k} = 0$:

$$S_m.(Y - S_m^T.\hat{M}_{k,LS}) = 0 \qquad \text{eq. 3.2.(9)}$$

From eq. 3.2.(9) it can be seen that the error matrix $E = Y - S_m^T.\hat{M}_{k,LS}$ is orthogonal to the columns of S_m^T ; something we already expected

from fig. 3.2.-1 .

To be sure that the solution of eq. 3.2.(9) is unique and minimizes eq. 3.2.(3) , the following expression has to hold:

$$V(M_k^*) = V(M_k+\Delta M_k) > V(M_k) \qquad \text{eq. 3.2.(10)}$$

for all ΔM_k .

Writing out $V(M_k^*)$:

$$V(M_k^*) = \text{tr} \{ (Y - S_m^T M_k - S_m^T \Delta M_k)^T . (Y - S_m^T M_k - S_m^T \Delta M_k) \}$$

eq. 3.2.(11)

leads to :

$$V(M_k^*) = V(M_k) - 2.\text{tr} \{\Delta M_k^T . S_m . (Y - S_m^T M_k)\} + \text{tr} \{\Delta M_k^T S_m S_m^T \Delta M_k\}$$

eq. 3.2.(12)

Since M_k has to satisfy eq. 3.2.(9) :

$$V(M_k^*) = V(M_k) + \text{tr} \{\Delta M_k^T S_m S_m^T \Delta M_k\} \qquad \text{eq. 3.2.(13)}$$

This implies that tr $\{\Delta M_k^T S_m S_m^T \Delta M_k\}$ has to be greater than zero, for all ΔM_k, to satisfy eq. 3.2.(10). A necessary and sufficient condition is that $S_m S_m^T$ is a positive definite matrix:

$$\underline{x}^T . S_m S_m^T . \underline{x} > 0 \qquad \text{eq. 3.2.(14)}$$

for all $\underline{x} \neq \underline{0}$.

And this will always be the case as long as S_m has maximal rank.

Concluding we may say that the least squares estimator that satisfies eq. 3.2.(9) results in a unique minimum if S_m is a matrix of full rank.

From eq. 3.2.(9) we know that we have to find a solution in the least squares sense of the following set of equations, to find a least squares

estimate $\hat{M}_{k,LS}$:

$$Y = S_m^T \cdot \hat{M}_{k,LS} \qquad \text{eq. 3.2.(15)}$$

This least squares solution can be found, using two methods:

1) a direct method, using Householder transformations
2) a solution by computing the pseudo-inverse $(S_m^T)^+$ of S_m^T, using Singular Value Decomposition.

Both methods lead to the same $\hat{M}_{k,LS}$. The first method reduces the S_m^T-matrix, which is a $(m+1)\times(k+1).p$-matrix $(m+1>>(k+1).p)$, to a matrix of which the upper square part is a non-singular, upper triangular matrix and the lower part only contains zeros. The transformation is also used for Y and then, $\hat{M}_{k,LS}$ is found by back-substitution and computation of corrections via an iterative procedure.
In the second method, Singular Value Decomposition (SVD) is used to decompose S_m^T:

$$S_m^T = W \cdot \Sigma \cdot V^T \qquad \text{eq. 3.2.(16)}$$

where W: column orthogonal matrix of dimensions $(m+1)\times(k+1).p$

Σ: $(k+1).p\times(k+1).p$-diagonal matrix containing the singular values of S_m^T

V: column orthogonal matrix of dimensions $(k+1).p\times(k+1).p$

For more information about SVD cf. Appendix B.

Using the decomposition, it is possible to compute, what is called, the pseudo-inverse of S_m^T : $(S_m^T)^+$

$$(S_m^T)^+ = (W.\Sigma.V^T)^+ = (V^T)^+.\Sigma^+ .W^+ = V.\Sigma^{-1}.W^T \qquad \text{eq. 3.2.(17)}$$

since $V^T.V=I$ and $W^T.W=I$.

With this pseudo-inverse we are able to find a LS-estimate $\hat{M}_{k,LS}$:

$$\hat{M}_{k,LS} = (S_m^T)^+ \cdot Y = V.\Sigma^{-1}.W^T.Y \qquad \text{eq. 3.2.(18)}$$

This method is very useful to gain some insight in the mapping of Y into the space formed by S_m .
Another advantage of the use of SVD is the ability to compute the condition number of S_m^T, which is defined as (see also Appendix B):

$$\text{cond}(S_m^T) = \|A^{-1}\|_s . \|A\|_s \qquad \text{eq. 3.2.(19)}$$

which equals the biggest singular value divided by the smallest one:

$$\text{cond}(S_m^T) = \frac{\Sigma(1,1)}{\Sigma(p.(k+1),p.(k+1))} \qquad \text{eq. 3.2.(20)}$$

Because S_m^T is of full rank, Σ will also have full rank and thus : $\Sigma(p.(k+1),p.(k+1)) \neq 0$. But when S_m^T is nearly singular the smallest singular value will be very small, which may result in a large condition number and we are dealing with a badly conditioned problem. This leads to an inaccurate solution, since a small change in the solution M_k results in a much bigger change in the output Y. When the input vectors are nearly orthogonal, and thus we have a well conditioned input signal, the condition number will be small. This leads to small computational errors in the computation of $\hat{M}_{k,LS}$.

A great disadvantage of solving $Y = S_m^T.\hat{M}_{k,LS}$ is the size of the Y- and S_m^T-matrix, for they are dependent on the number of samples taken into consideration. Because the data are processed simultaneously, this procedure normally has to be done by batch-processing. On the other hand, very old samples of the input- and output signals are not very interesting so it might be possible to shift a 'window' of, for example, 1000 samples along the data being gathered, with an overlap of 500 samples (see fig. 3.2.-2).

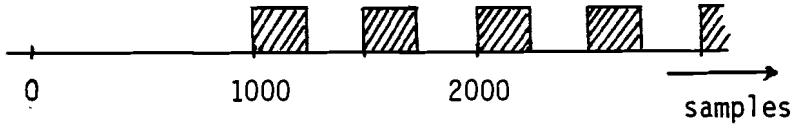


fig. 3.2.-2 'on-line' use of an explicit method

3.3 Recursive Least Squares

The approach described in the previous section has the disadvantage, that when an estimate has been determined based on m samples and new data become available, all computations have to be repeated; thus reprocessing old data. To avoid this highly inefficient procedure it would be attractive to determine an estimate after m+1 samples, based on the estimate after m samples and the new data of sample moment m+1. As will be shown in this section, this can be accomplished by the method that is called Recursive Least Squares.

Before discussing this recursive method, we first mention a matrix identity which we will need and that is called: the matrix inversion lemma:

$$(A + B.C.D)^{-1} = A^{-1} - A^{-1}.B.(C^{-1} + D.A^{-1}.B)^{-1}.D.A^{-1}$$

eq. 3.3.(1)

Verification of this lemma can be obtained by multiplying left and right hand side of eq. 3.3.(1) by (A + B.C.D) which results in the identity matrix I.

According to eq. 3.2.(9) the LS-estimate $\hat{M}_{k,LS}$ has to satisfy:

$$S_m.Y = S_m.S_m^T.\hat{M}_{k,LS} \qquad \text{eq. 3.3.(2)}$$

Let us now introduce the following notations :

$$P_m^{-1} = (S_m.S_m^T) \quad \text{and} \quad G_m = S_m.Y \qquad \text{eq. 3.3.(3)}$$

This means that P_m is the inverse of the covariance matrix of the input samples (uncorrected for the number of samples that are used).
Using this notation, eq. 3.3.(2) can be written as:

$$\hat{M}_{k,LS} = P_m.G_m \qquad \text{eq. 3.3.(4)}$$

Now that we know P_m and G_m , and thus also $\hat{M}_{k,LS}$ at sample moment m, we would like to find, in an easy way, P_{m+1} and G_{m+1} which will give us $\hat{M}_{k,LS}$ at sample moment m+1.

P_{m+1} satisfies the following equation:

$$P_{m+1}^{-1} = (S_{m+1} \cdot S_{m+1}^T) = (S_m | \underline{s}_{m+1}) \cdot \begin{bmatrix} S_m^T \\ \hline \underline{s}_{m+1}^T \end{bmatrix} \qquad \text{eq. 3.3.(5)}$$

where $\underline{s}_{m+1}$ is a (k+1).p×1-vector containing the most recent input data:

$$\underline{s}_{m+1}^T = (\underline{u}^T(m+1), \ldots, \underline{u}^T(m-k+1)) \qquad \text{eq. 3.3.(6)}$$

Working out eq. 3.3.(5) results in :

$$P_{m+1}^{-1} = S_m \cdot S_m^T + \underline{s}_{m+1} \cdot \underline{s}_{m+1}^T = P_m^{-1} + \underline{s}_{m+1} \cdot \underline{s}_{m+1}^T$$

eq. 3.3.(7)

Looking at eq. 3.3.(4) we see that we need P_{m+1} instead of P_{m+1}^{-1} :

$$P_{m+1} = (P_m^{-1} + \underline{s}_{m+1} \cdot \underline{s}_{m+1}^T)^{-1} \qquad \text{eq. 3.3.(8)}$$

When we compare eq. 3.3.(8) with the matrix inversion lemma (eq. 3.3.(1)) we see that the right hand side of eq. 3.3.(8) has a lot in common with the left hand side of eq. 3.3.(1). Let us now substitute the following expressions in the matrix inversion lemma:

$$A = P_m^{-1} \quad , B = \underline{s}_{m+1} \quad , C = 1 \quad , D = \underline{s}_{m+1}^T \qquad \text{eq. 3.3.(9)}$$

These substitutions result in:

$$P_{m+1} = P_m - P_m \cdot \underline{s}_{m+1} \cdot (1 + \underline{s}_{m+1}^T \cdot P_m \cdot \underline{s}_{m+1})^{-1} \cdot \underline{s}_{m+1}^T \cdot P_m$$

eq. 3.3.(10)

This expression is, in fact, a recursive procedure to compute P_{m+1} , when P_m is known.
Now, we only have to derive an expression for G_{m+1}:

$$G_{m+1} = (S_m | \underline{s}_{m+1}) \cdot \left[\frac{Y}{\underline{y}^T(m+1)}\right] = G_m + \underline{s}_{m+1} \cdot \underline{y}^T(m+1) \qquad \text{eq. 3.3.(11)}$$

So it is possible, indeed, to compute P_{m+1} and G_{m+1} when P_m and G_m are known and new data represented by $\underline{s}_{m+1}$ and $\underline{y}^T(m+1)$ become available.
Multiplying P_{m+1} and G_{m+1} results in a new estimate of the Markov-parameters:

$$\hat{M}_{k,m+1} = \hat{M}_{k,m} + P_m \cdot \underline{s}_{m+1} \cdot (1 + \underline{s}_{m+1}^T \cdot P_m \cdot \underline{s}_{m+1})^{-1} \cdot (\underline{y}^T(m+1) - \underline{s}_{m+1}^T \cdot M_{k,m})$$

eq. 3.3.(12)

Define:

$$K_{m+1} \triangleq P_m \cdot \underline{s}_{m+1} \cdot (1 + \underline{s}_{m+1}^T \cdot P_m \cdot \underline{s}_{m+1})^{-1} \qquad \text{eq. 3.3.(13)}$$

and eq. 3.3.(12) reduces to:

$$\hat{M}_{k,m+1} = \hat{M}_{k,m} + K_{m+1} \cdot (\underline{y}^T(m+1) - \underline{s}_{m+1}^T \cdot \hat{M}_{k,m}) \qquad \text{eq. 3.3.(14)}$$

Here we see that the new estimate of M_k is formed as the linear combination of the old estimate and a correction term:

$$\delta\hat{M}_{k,m+1} \triangleq K_{m+1} \cdot (\underline{y}^T(m+1) - \underline{s}_{m+1}^T \cdot \hat{M}_{k,m}) \qquad \text{eq. 3.3.(15)}$$

The residual $(\underline{y}^T(m+1) - \underline{s}_{m+1}^T \cdot \hat{M}_{k,m})$ can be regarded as the difference between the new output sample $\underline{y}^T(m+1)$ and the output sample predicted by the old estimate $\hat{M}_{k,m}$. This is why such estimators are often called Prediction Error Estimators. Thus the residual gives us a correction which is proportional to the magnitude of the error. The expression for K_{m+1} can be seen to be a direct function of P_m.
This means that the 'larger' the error covariance , the more weight is given to the residual.

A problem of this method is that at the beginning of this recursive procedure we don't know G_0 and P_0. A reasonable initial value of G_0 is zero, since $M_{k,0}=0$, and the choice of a starting value of P_0 can be made, according to the following consideration.
In the beginning of the estimation procedure, we expect large errors between the measured output and the predicted one, based on an old estimate. It is, therefore, sensible to give a large weight to the residual and this can be accomplished by choosing a large initial value for P_0.

Resuming, we can formulate the following algorithm (see fig. 3.3.-1) :

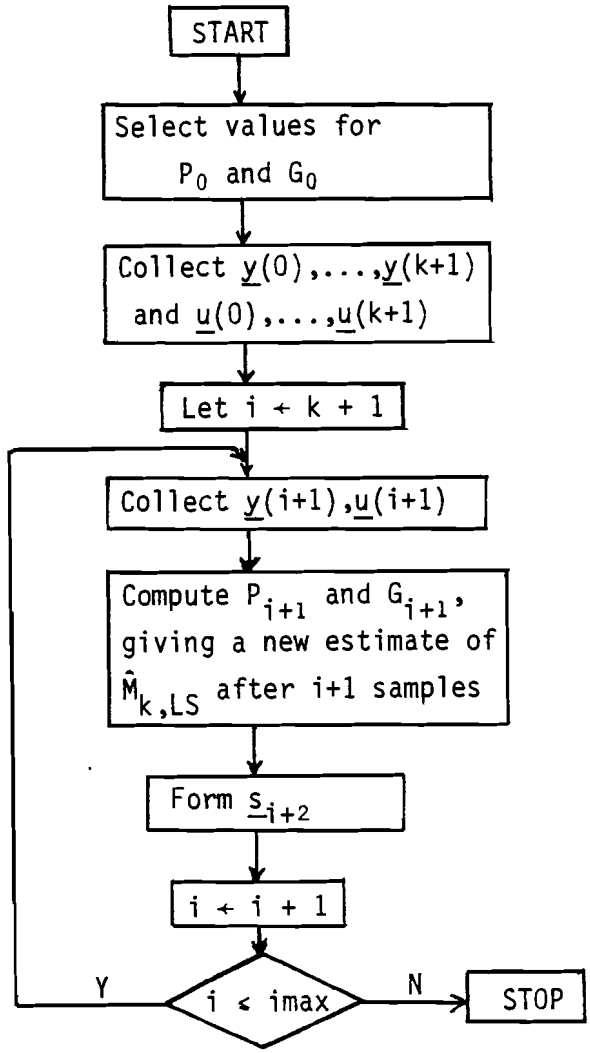


fig. 3.3.-1 block diagram of recursive estimation procedure

A very pleasing property of this recursive method is that we don't need to compute the inverse of a matrix. From eq. 3.3.(12) we see that the inversion reduces to a simple scalar division, which makes it possible to implement this method on a computer very easily.

3.4 An updating algorithm for LS-estimation

Let us recapitulate the most important properties of the methods described in the preceding sections.
The explicit method of 3.2 has the great advantage to be numerically stable and accurate, but needs to process all data simultaneously.
The recursive method, however, computes an estimate based on older estimates, thereby saving a lot of computational time. The great disadvantage of this method is that it may lead to inaccurate results, because of a bad starting point; specially when few samples are available.

The updating algorithm for LS-estimation, which will be described here, is more or less a compromise between the methods of 3.2 and 3.3 .
Our starting point is the same as for the recursive method:

$$S_m.Y = S_m.S_m^T.\hat{M}_{k,LS} \qquad \text{eq. 3.3.(2)}$$

This equation is referred to as the 'normal-equation'.
Because the matrix Y and $S_m^T.\hat{M}_{k,LS}$ are premultiplied by S_m , the dimensions of this equation are not anymore a function of the number of data samples that are taken into account, but only of the number of inputs and outputs of the system and the number of Markov-parameters that we want to estimate.
It is possible to solve this normal-equation as described in 3.2 by applying Singular Value Decomposition (SVD) or Householder-transformations.
Because the dimensions of the matrices are limited, this method needs less computational effort than the explicit method of 3.2 . Another advantage, compared to the recursive method, is that we don't need to know a starting value which could lead to inaccurate results.

The third advantage is the possibility for updating, when new data become available; since it is possible to adapt $S_m.Y$ and $S_m.S_m^T$ when sample m+1 is measured.

Analogous to the procedure, we used in 3.3, we can write:

$$S_{m+1}.S_{m+1}^T = (S_m|\underline{s}_{m+1}).\begin{bmatrix} S_m^T \\ \hline \underline{s}_{m+1}^T \end{bmatrix} = S_m.S_m^T + \underline{s}_{m+1}.\underline{s}_{m+1}^T \qquad \text{eq. 3.4.(1)}$$

where $\underline{s}_{m+1}^T = (\underline{u}^T(m+1),\ldots,\underline{u}^T(m-k+1))$.

The same for $S_m.Y$:

$$S_{m+1}.Y = (S_m|\underline{s}_{m+1}).\begin{bmatrix} Y \\ \hline \underline{y}^T(m+1) \end{bmatrix} = S_m.Y + \underline{s}_{m+1}.\underline{y}^T(m+1) \qquad \text{eq. 3.4.(2)}$$

So it is possible to recompute the matrices needed for the normal equation based on old data. A disadvantage of this method, in contrast to the recursive method, is that for every estimate a set of equations has to be solved. Another problem causes the premultiplication by S_m which leads to eq. 3.3.(1). This premultiplication leads to a condition number of the set of equations, which is the square of the condition number of eq. 3.2.(9), that is used by the explicit method. When the input signal is badly conditioned (e.g. a slowly varying pseudo binairy random signal), this leads to a matrix which is nearly singular and thus to a large condition number of $S_m.S_m^T$. This may lead to inaccurate solutions for $\hat{M}_{k,LS}$. This remark is, of course, also valid for the recursive method which uses also the normal-equation.

So far we have only taken into consideration the estimation of Markov-parameters, assuming that the output noise Ξ is white and channel independent. In the next chapter the three methods, discussed here, will extended to cope with coloured noise.

CHAPTER 4. NON-LINEAR LEAST-SQUARES ESTIMATION

4.1 Introduction

In 2.3 we have derived an extended model for the estimation of Markov-parameters of the process and auto-regressive parameters of the noise model.
This extended model can be described by eq. 2.3.(7):

$$Y = \underbrace{S_m^T \cdot M_k}_{1} + \underbrace{H_m^T \cdot A_r}_{2} + \Xi \qquad \text{eq. 2.3.(7)}$$

The first term of the right hand side represents the part of the output-signal that can be explained by the process. Part 2 stands for the contribution of the noise model to the output and what remains, Ξ, is supposed to contain no information anymore about the dynamical properties of the system and thus to be white noise.
Part 2 of eq. 2.3.(7) is not 'known' until part 1 has been determined. We will show that this will lead to a loss function which is non-quadratic and which can not be minimized in a simple way.

Let us take a look at eq. 2.3.(9):

$$\begin{bmatrix} \underline{y}^T(\ell) \\ \vdots \\ \underline{y}^T(\ell+m) \end{bmatrix} = \left[\begin{array}{ccc|ccc} \underline{u}^T(\ell) & \cdots & \underline{u}^T(\ell-k) & \underline{e}^T(\ell-1) & \cdots & \underline{e}^T(\ell-r) \\ \vdots & & \vdots & \vdots & & \vdots \\ \underline{u}^T(\ell+m) & \cdots & \underline{u}^T(\ell+m-k) & \underline{e}^T(\ell+m-1) & \cdots & \underline{e}^T(\ell+m-r) \end{array}\right] \cdot \begin{bmatrix} M^T(0) \\ \vdots \\ \hline M^T(k) \\ A^T(1) \\ \vdots \\ A^T(r) \end{bmatrix} + \begin{bmatrix} \underline{\xi}^T(\ell) \\ \vdots \\ \underline{\xi}^T(\ell+m) \end{bmatrix}$$

eq. 2.3.(9)

The equation errors $\underline{e}$ can be computed according to eq. 4.1.(1):

$$\underline{e}^T(i) = \underline{y}^T(i) - [\underline{u}^T(i),\ldots,\underline{u}^T(i-k)].\begin{bmatrix} M^T(0) \\ \vdots \\ M^T(k) \end{bmatrix} \qquad \text{eq. 4.1.(1)}$$

From this equation, it is clear that $\underline{e}$ is a function of the Markov-parameters:

$$\underline{e} = \underline{e}(M_k) \qquad \text{eq. 4.1.(2)}$$

When the equation error is a function of M_k then, of course, also H_m^T is, which is formed by 'older' equation errors.
This leads to the following equation of our model, that is non-linear in the parameters, since both M_k and A_r are unknown:

$$Y = [S_m^T \mid H_m^T(M_k)] \cdot \begin{bmatrix} M_k \\ \hline A_r \end{bmatrix} + \Xi \qquad \text{eq. 4.1.(3)}$$

Our main goal is to find estimates $\hat{M}_k$ and $\hat{A}_r$, of respectively M_k and A_r, that satisfy:

$$\hat{Y} = [S_m^T \mid H_m^T(\hat{M}_k)] \cdot \begin{bmatrix} \hat{M}_k \\ \hline A_r \end{bmatrix} \qquad \text{eq. 4.1.(4)}$$

and that minimize the loss function V, which is defined as:

$$V = \mathrm{tr}\{(Y - \hat{Y})^T.(Y - \hat{Y})\} \qquad \text{eq. 3.2.(3)}$$

Because the expression for $\hat{Y}$ is not linear in the parameters, the loss function will not be a simple quadratic function, but of the fourth order. Hence, it will, in general, not be possible to find a minimum of V in a closed form and we will need some iterative procedure to determine the minimum of V by varying M_k and A_r. Such a procedure is illustrated in fig. 4.1.-1. When it is possible to approximate $\hat{Y}$, in some point, by

a linear function, the approximate loss function will be quadratic in $\hat{M}_k$. The minimum of this quadratic function may give us a better estimate, and using this estimate a new approximate loss function can be obtained. Continuing this procedure, we hope that it will converge to the exact minimum of the non-quadratic loss function. Having found this minimum, we have the best possible estimate of M_k and A_r in the least squares sense.

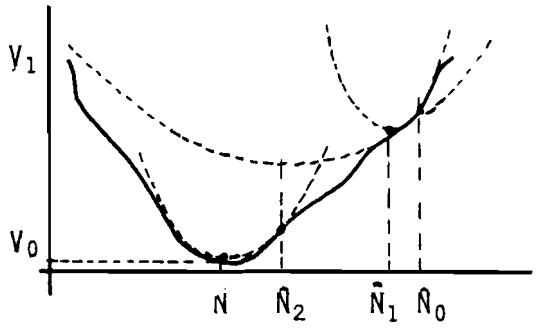


fig. 4.1.-1 approximation of the loss function by a quadratic curve

4.2 Non-linear LS using an iterative explicit method

As we have seen in the introduction of this chapter, H_m^T is a function of the Markov-parameters M_k . To gain some insight in this dependency of M_k, we shall have to write H_m^T explicitly as follows:

$$H_m^T = \begin{bmatrix} \underline{e}^T(\ell-1) & \cdots & \underline{e}^T(\ell-r) \\ \vdots & & \vdots \\ \underline{e}^T(\ell+m-1) & \cdots & \underline{e}^T(\ell+m-r) \end{bmatrix} \qquad \text{eq. 4.2.(1)}$$

Since the equation errors $\underline{e}^T(i)$ are the difference between the measured output and the predicted output based on the Markov-parameters:

$$\underline{e}^T(i) = \underline{y}^T(i) - [\underline{u}^T(i),\ldots,\underline{u}^T(i-k)] \cdot \begin{bmatrix} M^T(0) \\ \vdots \\ M^T(k) \end{bmatrix} \qquad \text{eq. 4.1.(1)}$$

H_m^T can be written as:

$$H_m^T = \begin{bmatrix} \underline{y}^T(\ell-1) & \cdots & \underline{y}^T(\ell-r-1) \\ \vdots & & \vdots \\ \underline{y}^T(\ell+m-1) & \cdots & \underline{y}^T(\ell+m-r-1) \end{bmatrix} - \begin{bmatrix} \underline{u}^T(\ell-1) & \cdots & \underline{u}^T(\ell-k-1) \\ \vdots & & \vdots \\ \underline{u}^T(\ell+m-1) & \cdots & \underline{u}^T(\ell+m-k-1) \end{bmatrix} \cdots \begin{bmatrix} \underline{u}^T(\ell-r) & \cdots & \underline{u}^T(\ell-k-r) \\ \vdots & & \vdots \\ \underline{u}^T(\ell+m-r) & \cdots & \underline{u}^T(\ell+m-k-r) \end{bmatrix} . M^*$$

$$\triangleq Y_m^* \qquad \triangleq S_{m,\ell-1}^T \qquad \triangleq S_{m,\ell-r}^T$$

$$\text{where } M^* = \begin{bmatrix} M_k & 0 & \cdots & 0 \\ 0 & M_k & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & M_k \end{bmatrix} \text{ and } M_k = \begin{bmatrix} M^T(0) \\ \vdots \\ M^T(k) \end{bmatrix} \qquad \text{eq. 4.2.(2)}$$

Using the shorter notations, suggested above, this equation reduces to:

$$H_m^T = Y_m^* - [S_{m,\ell-1}^T \,|\, S_{m,\ell-2}^T \,| \ldots | S_{m,\ell-r}^T] \cdot M^* \qquad \text{eq. 4.2.(3)}$$

Now, suppose that we have a starting point $M_{k,0}$ and $A_{r,0}$ and a resulting M_0^*, then it is possible to linearize eq. 4.1.(4), using eq. 4.2.(3):

$$\hat{Y} = (S_m^T \,|\, Y_m^* - [S_{m,\ell-1}^T \,|\, \ldots. \,|\, S_{m,\ell-r}^T] \cdot M_0^*) \cdot \begin{bmatrix} M_{k,0} \\ A_{r,0} \end{bmatrix} + \qquad \text{(a)}$$

$$(S_m^T \,|\, Y_m^* - [S_{m,\ell-1}^T \,|\, \ldots \,|\, S_{m,\ell-r}^T] \cdot M_0^*) \cdot \begin{bmatrix} \Delta M_k \\ \Delta A_r \end{bmatrix} + \qquad \text{(b)}$$

$$(Y_m^* - [S_{m,\ell-1}^T \,|\, \ldots. \,|\, S_{m,\ell-r}^T] \cdot \Delta M^*) \cdot A_{r,0} + \qquad \text{(c)}$$

$$(Y_m^* - [S_{m,\ell-1}^T \,|\, \ldots. \,|\, S_{m,\ell-r}^T] \cdot \Delta M^*) \cdot \Delta A_r \qquad \text{(d)}$$

eq. 4.2.(4)

where: $\Delta M_k = \hat{M}_k - M_{k,0}$ $\qquad$ $M_0^* = \begin{bmatrix} M_{k,0} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & M_{k,0} \end{bmatrix}$

$\Delta A_r = \hat{A}_r - A_{r,0}$

$$\Delta M^* = \begin{bmatrix} \Delta M_k & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \Delta M_k \end{bmatrix}$$

We shall approximate this equation by neglecting term (c) and (d):

(d): because it contains higher order parts and

(c): since neglection of this implicit term leads to comparable conditions for the $\hat{N}$ that minimizes the loss function (eq. 4.2.(12)) as we found in section 3.2 for $\hat{M}_k$ (eq. 3.2.(9)).

These simplifications result in:

$$\hat{Y} = (S_m^T \,|\, Y_m^* - [S_{m,\ell-1}^T \,|\, \ldots \,|\, S_{m,\ell-r}^T].M_0^*).\begin{bmatrix} M_{k,0} + \Delta M_k \\ \hline A_{r,0} + \Delta A_r \end{bmatrix} \qquad \text{eq. 4.2.(5)}$$

yielding:

$$\tilde{Y} = (S_m^T \,|\, H_m^T(M_{k,0})).\begin{bmatrix} \hat{M}_k \\ \hline \hat{A}_r \end{bmatrix} \qquad \text{eq. 4.2.(6)}$$

Substitution of this equation in the expression for the loss function V, eq. 3.2.(3) gives:

$$V = \mathrm{tr}\{(Y - S_m^T.\hat{M}_k - H_m^T(M_{k,0}).\hat{A}_r)^T.(Y - S_m^T.\hat{M}_k - H_m^T(M_{k,0}).\hat{A}_r)\} \qquad \text{eq. 4.2.(7)}$$

A necessary condition for $\hat{M}_k$ and $\hat{A}_r$ to minimize V is:

$$\frac{\partial V}{\partial \hat{M}_k} = 0 \quad \text{and} \quad \frac{\partial V}{\partial \hat{A}_r} = 0 \qquad \text{eq. 4.2.(8)}$$

Computation of the derivatives leads to:

$$\frac{\partial V}{\partial \hat{M}_k} = S_m \,.\, (Y - S_m^T.\hat{M}_k - H_m^T(M_{k,0}).\hat{A}_r\ ) = 0 \qquad \text{eq. 4.2.(9)}$$

and

$$\frac{\partial V}{\partial \hat{A}_r} = H_m(M_{k,0}).(Y - S_m^T.\hat{M}_k - H_m^T(M_{k,0}).\hat{A}_r) = 0 \qquad \text{eq. 4.2.(10)}$$

Combining these two equations, using the following notation:

$$\beta_m^T(M_{k,0}) = (S_m^T |\ H_m^T(M_{k,0})) \quad \text{and} \quad \hat{N} = \begin{bmatrix} \hat{M}_k \\ \hline \hat{A}_r \end{bmatrix} \qquad \text{eq. 4.2.(11)}$$

we get:

$$\frac{\partial V}{\partial \hat{N}} = \beta_m(M_{k,0}).(Y - \beta_m^T(M_{k,0}).\hat{N}) = 0 \qquad \text{eq. 4.2.(12)}$$

This equation is very similar to eq. 3.2.(7) that is generally valid, while eq. 4.2.(12) refers to the neighbourhood of $M_{k,0}$.

Hence, the $\hat{M}_k$ and $\hat{A}_r$ that minimize the approximated loss function can be found by solving the following set of linear equations in least squares sense:

$$Y = S_m^T.\hat{M}_k + H_m^T(M_{k,0}).\hat{A}_r \qquad \text{eq. 4.2.(13)}$$

which is the same as:

$$Y = \beta_m^T(M_{k,0}) \,.\, \hat{N} \qquad \text{eq. 4.2.(14)}$$

The LS-solution of this equation gives us the possibility to update H_m^T, and thus β_m^T , which results in a new set of equations to be solved.

This leads to an iterative procedure to refine the estimate of $\hat{M}_k$ and $\hat{A}_r$.
The solution of eq. 4.2.(14) can be obtained by using one of the methods described in 3.2; the use of the Householder transformation is here the more attractive one since it needs less computational effort than the singular value decomposition.
When many iterations are needed to reach the minimum of V, the method as described above will cost very much computation time and is, therefore, not recommendable, although it has the advantage of being numerically stable.

As in 3.2 it is possible to give a geometrical interpretation of this iterative non-linear least squares procedure for the vector case.

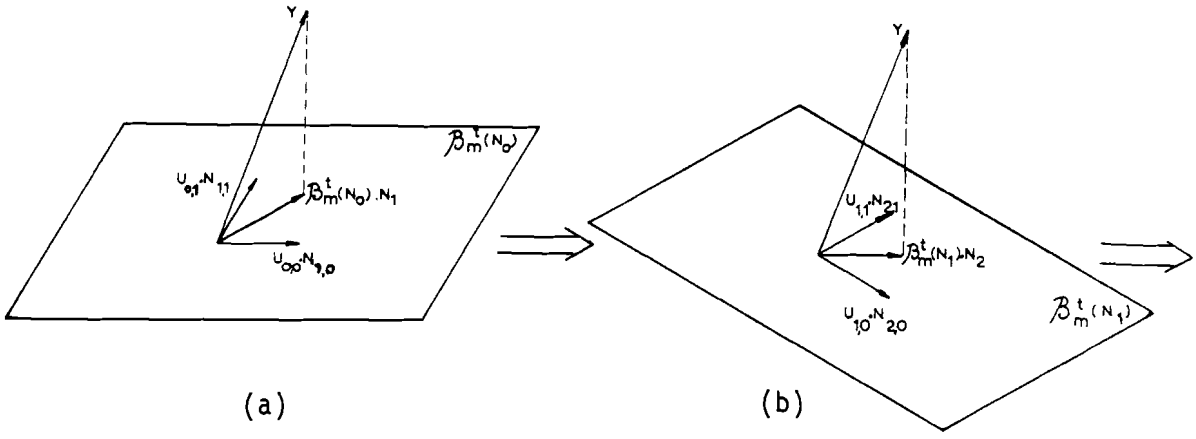


fig 4.2 -1 Geometrical interpretation of NLLS-estimation

For the vector case eq. 4.2.(14) reduces to:

$$\underline{Y} = \beta_m^T(\underline{M}_{k,0}) \cdot \hat{N} \qquad \text{eq. 4.2.(15)}$$

The first time we compute the LS-solution of this equation, an Euclidean space (a) is formed by the input vectors $\underline{u}(i)$ and estimated

equation errors $\hat{\underline{e}}(i)$. The parameter vector $\hat{\underline{N}}$ is mapped into this space by $\beta_m^T(\underline{M}_{k,0})$. Since the output vector $\underline{Y}$ contains noise influences, it doesn't fit into this space. An approximation $\hat{\underline{Y}}$ of $\underline{Y}$ is the projection of $\underline{Y}$ on space (a); thus minimizing the Euclidean distance between $\underline{Y}$ and $\hat{\underline{Y}}$.
The parameter vector $\hat{\underline{N}}_1 = \begin{bmatrix} \hat{M}_{k,1} \\ \hline \hat{A}_{r,1} \end{bmatrix}$, obtained by the projection

$\hat{\underline{Y}} = \beta_m^T(M_{k,0}).\hat{\underline{N}}_1$, can be used to recompute the estimated equation errors according to $\hat{\underline{e}} = \underline{Y} - S_m^T.\underline{M}_{k,1}$. Together with the input vectors $\underline{u}(i)$ these updated equation errors form a new space (b). The output vector $\underline{Y}$ can now be projected on this space (b); resulting in an estimate $\hat{\underline{N}}_2$ and so on. This iterative procedure can be stopped when the estimated equation errors have converged to the exact values $\underline{e}$, resulting in the estimated parameter vector $\hat{\underline{N}}$ that minimizes the loss function V. During this iterative procedure, the estimated noise vector $\hat{\underline{\Xi}}$ is always orthogonal to the columns of $\beta_{m_T}^T(\hat{\underline{N}})$ and this implies that at the end $\hat{\underline{\Xi}}$ is orthogonal to the columns of S_m^T and H_m^T, built from the exact equation errors. Hence, having reached the minimum of the loss function there will be no correlation anymore between $\underline{\Xi}$ and the input signals $\underline{u}(i)$ and between $\underline{\Xi}$ and the equation errors $\underline{e}(i)$.

4.3 Recursive non-linear least squares

In 3.3 we introduced a recursive least squares method, because of the advantage that not all data need to be reprocessed when new data become available.
For the same reason we will develop a non-linear version of this recursive method.

In 4.2 we found that the set of Markov-parameters, $\hat{M}_k$, and auto-regressive parameters, $\hat{A}_r$, which minimize the approximated loss function in the neighbourhood of $M_{k,0}$ and $A_{r,0}$, is given by the solution of eq. 4.2.(12), that is identical to:

$$\beta_m(M_{k,0}) \cdot Y = \beta_m . \beta_m^T(M_{k,0}) . \hat{N}_m \qquad \text{eq. 4.3.(1)}$$

where the subscript m of $\hat{N}_m$ points to the number of samples taken into consideration.

In exactly the same way as in section 3.3 it is possible to derive a recursive solution method for eq. 4.3.(1).

Define: $$P_m^{-1} = \beta_m . \beta_m^T(M_{k,0}) \text{ and } G_m = \beta_m . Y \qquad \text{eq. 4.3.(2)}$$

so: $$\hat{N}_m = P_m . G_m \qquad \text{eq. 4.3.(3)}$$

$\underline{\omega}_{m+1}^T$ will be defined as the extension of β_m^T when a new sample m+1 becomes available:

$$\underline{\omega}_{m+1}^T = (\underline{u}^T(\ell+m+1) \ldots \underline{u}^T(\ell+m+1-k) | \underline{e}^T(\ell+m) \ldots \underline{e}^T(\ell+m-r+1)) \qquad \text{eq. 4.3.(4)}$$

By substituting the extended matrix β_m^T for S_m^T and the extended vector $\underline{\omega}_{m+1}^T$ for $\underline{s}_{m+1}^T$ in resp. eq. 3.3.(10) - eq. 3.3.(12), we get:

$$P_{m+1} = P_m - P_m \cdot \underline{\omega}_{m+1} . (1+\underline{\omega}_{m+1}^T . P_m . \underline{\omega}_{m+1})^{-1} . \underline{\omega}_{m+1}^T . P_m \qquad \text{eq. 4.3.(5)}$$

$$G_{m+1} = G_m + \underline{\omega}_{m+1} . \underline{y}^T(\ell+m+1) \qquad \text{eq. 4.3.(6)}$$

and finally:

$$\hat{N}_{m+1} = \hat{N}_m + P_m . \underline{\omega}_{m+1} . (1+\underline{\omega}_{m+1}^T . P_m . \underline{\omega}_{m+1})^{-1} . (\underline{y}^T(\ell+m+1) - \underline{\omega}_{m+1}^T . \hat{N}_m) \qquad \text{eq. 4.3.(7)}$$

For the choice of a starting value P_0 and N_0 we refer to 3.3, where some remarks on this topic were made.
Thus, having a good starting value for P and N, it is possible to compute a new estimate $\hat{N}_{m+1}$ based on old data, a new output sample and a new value for $\underline{\omega}_{m+1}$.

At this point arises a problem: The new estimate is determined using $\underline{\omega}_{m+1}$ and this vector contains:

$$\underline{e}^T(\ell+m) = \underline{y}^T(\ell+m) - \underline{s}_m^T \cdot \hat{M}_{k,m+1} \qquad \text{eq. 4.3.(8)}$$

Formally, the equation error at sample moment m+1 must be computed based on the Markov-parameters as known after m+1 samples, which is of course due to the non-linearity of eq. 4.1.(4).
But this computation can't be performed since we don't know these Markov-parameters yet. The best prediction, that we are able to, is $\underline{\omega}_{m+1}^T$ based on the Markov-parameters as known after m samples:

$$\underline{e}^T(\ell+m) \simeq \underline{y}^T(\ell+m) - s_m^T \cdot \hat{M}_{k,m} \qquad \text{eq. 4.3.(9)}$$

(This is why this method is also called a prediction error method).
To be sure that this approximation is reasonable we must have a good estimate of the Markov-parameters before we can start estimating the auto-regressive noise-parameters using these approximated equation errors.
This can be obtained by not estimating the auto-regressive parameters at the beginning, but only Markov-parameters, using the procedure of 3.3.
After a number of samples the estimate of the Markov-parameters is fairly reliable and a prediction of the equation error can be computed, using eq. 4.3.(9).
After an estimate $\hat{M}_{k,m+1}$ has been obtained, eq. 4.3.(8) can be computed and be used in $\underline{\omega}_{m+1}$; so that in a next recursion step the errors in the equation error are as small as possible.
This recursive method has the following properties which decrease the reliability of the final estimate of the parameters, we are looking for:

1) The recursive method executes only one iteration; so the minimum of the loss function that is found, is only the minimum of a quadratic curve, that is an approximation of the exact loss function after m samples. When sample m+1 is known, a new approximate loss function is computed, of which the minimum can be determined, etc.. There is, however, no guarantee that the final estimate of the parameters has converged to the value belonging to the exact minimum of the non-quadratic loss function.

2) The equation errors, that are computed during the estimation procedure are corrected nowhere, although at the end a much better estimate of the Markov-parameters is determined. This is due to the fact that this recursive method doesn't reprocess old data, but only the most recent data.

In spite of these disadvantages of this method, it may be very useful; specially because it doesn't need much computational effort.

4.4 An updating algorithm for NLLS-estimation

In 4.1 we have seen that we needed an iterative procedure to minimize the non-**quad**ratic loss function. The explicit method of 4.2, however, appeared to be a very inefficient procedure, because for every iteration all data had to be reprocessed and the solution of the extended set of equations was very time consuming.

In 3.4 we introduced an updating algorithm for LS-estimation, which was a compromise between the explicit and the recursive solution method. Here we shall use the same idea for an iterative updating algorithm for non-linear least squares estimation, which is, in fact, numerically the same as the iterative method of 4.2.

Recalling eq. 4.2.(12):

$$\frac{\partial V}{\partial \hat{N}} = \beta_m(M_{k,0}).(Y-\beta_m{}^T(M_{k,0}).\hat{N} = 0 \qquad \text{eq. 4.2.(12)}$$

we see that in the first iteration, the $\hat{N}$ that minimizes the approximated loss function is found by solving:

$$\beta_m(M_{k,0}).Y = \beta_m.\beta_m{}^T(M_{k,0}).\hat{N} \qquad \text{eq. 4.3.(1)}$$

where $M_{k,0}$ is an initial estimate of the upper part of $\hat{N} = \begin{bmatrix} \hat{M}_k \\ \hat{A}_r \end{bmatrix}$.

This equation is similar to the 'normal-equation' (eq. 3.3.(2)), with which we started in 3.4, except for the difference that the equation above

is an approximation that is only valid in the neighbourhood of $M_{k,0}$. Having found an initial estimate $M_{k,0}$, an iterative procedure can be started very easily by computing $\beta_m(M_{k,0}).Y$ and $\beta_m.\beta_m^T(M_{k,0})$, solving eq. 4.3.(1) and by back substitution of the upper part of the solution $\hat{N} = \begin{bmatrix} \hat{M}_{k,1} \\ \hat{A}_{r,1} \end{bmatrix}$.
This results in an iterative solution procedure using successive substitution. A disadvantage of this method is that convergence is slow and can not be guaranteed.

Using the notation of eq. 4.2.(11):

$$\beta_m^T(M_{k,0}) = (S_m^T | H_m^T(M_{k,0})) \text{ and } \hat{N} = \begin{bmatrix} \hat{M}_k \\ \hat{A}_r \end{bmatrix} \qquad \text{eq. 4.2.(11)}$$

eq. 4.3.(1) can be written as:

$$\begin{bmatrix} S_m.Y \\ H_m^T(M_{k,0}).Y \end{bmatrix} = \begin{bmatrix} S_m.S_m^T & S_m.H_m^T(M_{k,0}) \\ H_m(M_{k,0}).S_m^T & H_m(M_{k,0}).H_m^T(M_{k,0}) \end{bmatrix} \cdot \begin{bmatrix} \hat{M}_k \\ \hat{A}_r \end{bmatrix}$$

eq. 4.4.(1)

We choose $M_{k,0}=0$ as a starting point, which is the same as solving:

$$S_m.Y = S_m.S_m^T.\hat{M}_k \qquad \text{eq. 4.4.(2)}$$

that is ordinairy least squares estimate, that we used in section 3.4. This gives us a useful approximation of the Markov-parameters to start the iterative procedure, that also estimates the auto-regressive parameters of the noise-model.

Until now, we have only treated how to solve eq. 4.3.(1). For the updating of this equation we refer to appendix C, because the derivation of the algorithm is so complicated that it doesn't fit in the context of this section.

To conclude this updating algorithm we shall consider some of its pro-

perties.
As we have already found in section 3.4, the dimensions of the set of equations that had to be solved, are not a function of the number of samples, taken into consideration, but only of the number of parameters that we want to estimate.
Because of this property, the solution of $\hat{N}$, found by iteration, is obtained much faster than by the method of 4.2.
Compared with the recursive method, this updating algorithm has the advantage that for every iteration all the equation errors are recomputed, using the most recent estimate $\hat{M}_k$. This means that no errors are introduced by old predictions of the equation errors. The price, we have to pay, is that more computation time and more storage facilities than for the recursive version are needed.
As already pointed out in 3.4, both the recursive and the updating iterative algorithm may cause more numerical problems than the explicit method because of the premultiplication of β_m^T by β_m, when a badly conditioned input signal is used.
Instead of solving eq. 4.3.(1) by successive substitution, it is also possible to minimize the loss function

$$V = \text{trace } \{(Y-\hat{Y})^T.(Y-\hat{Y})\} \qquad \text{eq. 3.2.(3)}$$

by using a Newton method for the minimization of a non-linear function. This method has better properties concerning convergence than a method using successive substitution.

Substitution of $\hat{Y} = \beta_m^T(\hat{M}_k).\hat{N}$ in eq. 3.2.(3)

results in:

$$V = \text{tr.}\{(Y-\beta_m^T(\hat{M}_k).\hat{N})^T.(Y-\beta_m^T(\hat{M}_k).\hat{N})\} \qquad \text{eq. 4.4.(3)}$$

Which is the same as minimizing

$$V = \text{tr.}\{\hat{\Xi}^T.\hat{\Xi}\} \qquad \text{eq. 4.4.(4)}$$

where $\hat{\Xi}$ is an estimate of the noise that can't be explained by the process model nor the noise model.

Working out eq. 4.4.(3) results in:

$$V = \text{tr}\ \{Y^T.Y - Y^T.\beta_m{}^T(\hat{M}_k).\hat{N} - \hat{N}.\beta_m(\hat{M}_k).Y + \hat{N}^T.\beta_m(\hat{M}_k).\beta_m{}^T(\hat{M}_k).\hat{N}\} \qquad \text{eq. 4.4.(5)}$$

that is identical to:

$$V = \text{tr}\ (Y^T.Y) - 2.\text{tr}(\hat{N}^T.\beta_m.Y) + \text{tr}\ \{\hat{N}^T.\beta_m.\beta_m{}^T.\hat{N}\} \qquad \text{eq. 4.4.(6)}$$

Since we already computed $\beta_m(\hat{M}_k).Y$ and $\beta_m(\hat{M}_k)\beta_m{}^T(\hat{M}_k)$ for the updating algorithm (see eq. 4.3.(1)), the computation of the loss function V is not so difficult.
For minimization of V we need an initial estimate, which can be found by performing 1 iteration of the updating algorithm.
A disadvantage of this method is that to minimize V, first order derivatives with respect to all parameters have to be computed. So although the advantage of a better convergence, this method is not very efficient because of practical computational aspects. The reason for describing this method becomes clear in the next chapter, where we shall see that this method can be used to compute a maximum likelihood estimate.

Before finishing this chapter, we give a short enumeration of the algorithms that we derived in this chapter, and their properties:

a) an explicit method, which is a numerically stable and iterative procedure resulting in a least squares estimate of the Markov-parameters and the auto-regressive noise parameters. However, this method needs much computer storage and a lot of computation time.

b) a recursive method, that results in an approximated least squares estimate of the parameters, since only one iteration is carried out. A great advantage of this method is that not much computational effort or storage is needed.

c) an updating iterative procedure leading to a least squares estimate of the parameters being, in fact, a compromise between the explicit and the recursive method, as far as accuracy, needed computation time and storage is concerned.

d) by small changes to the updating iterative procedure we can find a least squares estimate by minimization of the loss function, using a 'hill-climbing' method. The execution of this method is very time comsuming, but it has better properties concerning convergence than the methods using successive substitution (a and c).

CHAPTER 5 MAXIMUM LIKELIHOOD ESTIMATION

In this chapter we shall pay attention to the Maximum Likelihood-method (ML), which is one of the generally applicable methods for parameter estimation. In the first section we will work out the ML-method for the two different models, discussed in chapter 2. Also, the relation between the maximum likelihood method and the methods based on least squares estimation techniques, described in chapter 3 and 4, is pointed out.
In the second section the maximum likelihood method will be derived for the estimation of the parameters of a third model. This third model is introduced to be able to deal with additive output noise, which is coloured by a filter that contains a direct term.
This means that the output noise of channel i may be correlated with the input noise of channel j, at the same moment, which is a phenomenon that is often met in practice.

5.1 Maximum likelihood estimation and its relation with least squares

The method of least squares estimation, which was treated in chapter 3 and 4, was introduced independently of probability theory. In fact, we only minimized a sum of squares of the error between the observed output and the output predicted by the model. Therefore, this method is often called a least squares prediction error method (PEM).
The criterion of this method can also be derived from a probabilistic point of view, using the maximum likelihood method.
First, we shall derive the ML-method for the model of fig. 2.2.-2 that has been used in chapter 3. This means that the additive output noise Ξ is assumed to be stationary, white, Gaussian noise, that is 'white' in time ând in place (cf. eq. 2.3.(1) and 2.3.(2)),(the case where this additive output noise is coloured, Gaussian noise will be dealt with later on).

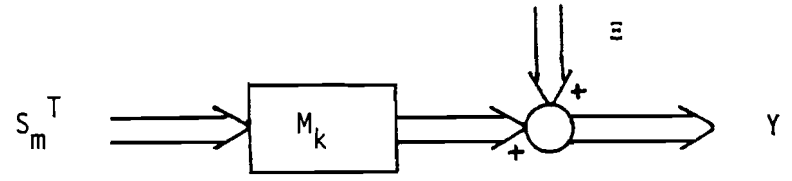


fig. 2.2.-2 model for the estimation of Markov-parameters

Our model can be described by

$$Y = S_m^T . M_k + \Xi \qquad \text{eq. 2.2.(4)}$$

where Ξ is assumed to contain noise vectors $\underline{\xi}(i)$ that have a Gaussian distribution $N(0,\sigma^2.I)$ for all sample moments i that are taken into account, which means that:

$$\forall_i : E\{\underline{\xi}(i)\} = \underline{0} \text{ and } R \triangleq E\{\underline{\xi}.\underline{\xi}^T\} = \sigma^2 I \qquad \text{eq. 5.1.(1)}$$

Because the vector $\underline{\xi}$ has a normal distribution and because it is stationary noise, it has a multivariate probability density function $p(\underline{\xi})$:

$$p(\underline{\xi}) = \frac{1}{2\pi^{q/2}(\det R)^{\frac{1}{2}}} . \exp\left[-\tfrac{1}{2}(\underline{\xi}^T.R^{-1}.\underline{\xi})\right] \qquad \text{eq. 5.1.(2)}$$

where q = dimension of the $(q\times 1)$-vector $\underline{\xi}$.
According to the assumption about the whiteness of the noise, all m+1 samples of $\underline{\xi}$ are independent.
The joint probability density function can, therefore, be written as the product of the density functions of each sample $\underline{\xi}(i)$:

$$p(\Xi) = p(\underline{\xi}(\ell),\ldots,\underline{\xi}(\ell+m)) = \prod_{i=0}^{m} p(\underline{\xi}(\ell+i)) \qquad \text{eq. 5.1.(3)}$$

which results in:

$$p(\Xi) = \frac{1}{(2\pi)^{\frac{q(m+1)}{2}}.(\det R)^{\frac{m+1}{2}}} . \exp\left[-\tfrac{1}{2}\sum_{i=0}^{m} \underline{\xi}^T(\ell+i).R^{-1}.\underline{\xi}(\ell+i)\right] \qquad \text{eq. 5.1.(4)}$$

Using: $\underline{\xi}^T(\ell+i) = \underline{y}^T(\ell+i) - \underline{u}^T(\ell+i).\hat{M}(0) -\ldots- \underline{u}^T(\ell+i-k).\hat{M}(k)$

$$= \underline{y}^T(\ell+i) - \underline{\hat{y}}^T(\ell+i) \qquad \text{eq. 5.1.(5)}$$

we see that $p(\Xi)$ can be presented as a function of the Markov-parameters $\hat{M}(0),\ldots,\hat{M}(k)$ that have to be estimated.
Eq. 5.1.(4) can be regarded as a measure of the 'likelihood' for a particular value of Ξ, when given a value of the Markov-parameters.

When we substitute the observed output samples $\underline{y}^T(\ell+i)$ in eq. 5.1.(4), this results in eq. 5.1.(6) that is called the 'likelihood function' L:

$$L(Y;M_k) = \frac{1}{(2\pi)^{\frac{q(m+1)}{2}}.(\det R)^{\frac{m+1}{2}}} \exp\left[-\tfrac{1}{2}\sum_{j=\ell}^{\ell+m}(\underline{y}^T(j)-\underline{\hat{y}}^T(j)).R^{-1}.(\underline{y}(j)-\underline{\hat{y}}(j)\right] \qquad \text{eq. 5.1.(6)}$$

where $\underline{\hat{y}}(j)$ is a function of the Markov-parameters.
A good estimate of the Markov-parameters is such an estimate that it is the 'most likely' value for $\hat{M}_k$, given the data substituted in the likelihood-function. This estimate of $\hat{M}_k$ can be found by maximizing $L(Y;M_k)$.

Instead of maximizing $L(Y;M_k)$, we will minimize $-\ln L(Y;M_k)$:

$$-\ln L = \frac{q(m+1)}{2}.\ln 2\pi + \frac{m+1}{2}\ln(\det R) + \tfrac{1}{2}\sum_{j=\ell}^{\ell+m}[(\underline{y}^T(j)-\underline{\hat{y}}^T(j)).R^{-1}.(\underline{y}(j)-\underline{\hat{y}}(j))] \qquad \text{eq. 5.1.(7)}$$

Substituting the a priori information $R=\sigma^2.I$ and using $\mathrm{tr}(A^T.B)=\mathrm{tr}(A.B^T)$, we find:

$$-\ln L = (m+1).(\frac{q}{2}\ln 2\pi + \tfrac{1}{2}.\ln q + \ln\sigma) + \frac{1}{2\sigma^2}\mathrm{tr}\{(Y-\hat{Y})^T(Y-\hat{Y})\} \qquad \text{eq. 5.1.(8)}$$

where $\hat{Y} = S_m^T.\hat{M}_k$.
A necessary condition for minimization of $(-\ln L)$ is:

$$\frac{\partial(-\ln L)}{\partial \hat{M}_k} = 0 \quad \text{and} \quad \frac{\partial(-\ln L)}{\partial\sigma} = 0 \qquad \text{eq. 5.1.(9)}$$

Computation of the last part gives us:

$$\frac{\partial(-\ln L)}{\partial\sigma} = \frac{m+1}{\sigma} - \frac{1}{\sigma^3}.\mathrm{tr}\{(Y-\hat{Y})^T.(Y-\hat{Y})\} = 0 \qquad \text{eq. 5.1.(10)}$$

This leads to:

$$\sigma^2 = \frac{1}{m+1}.\mathrm{tr}\{(Y-\hat{Y})^T.(Y-\hat{Y})\} \qquad \text{eq. 5.1.(11)}$$

Substitution of eq. 5.1.(11) in eq. 5.1.(8) and computation of $\frac{\partial(-\ln L)}{\partial \hat{M}_k}$ results in the following condition for $\hat{M}_k$:

$$\frac{\partial \mathrm{tr}\{(Y-\hat{Y})^T.(Y-\hat{Y})\}}{\partial \hat{M}_k} = 0 \qquad \text{eq. 5.1.(12)}$$

When we compare this result with eq. 3.2.(4) we find that the maximum likelihood estimate $\hat{M}_{k,ML}$ that fulfils eq. 5.1.(12) is the same as the least squares estimate $\hat{M}_{k,LS}$.
This implies that in this special case, where Ξ is white, channel independent, stationary, Gaussian noise, the least squares estimate that we discussed in chapter 3 is the same as the maximum likelihood estimate, although the least squares principle makes sense even if no probabilistic assumptions are made.

The maximum likelihood estimator has some nice properties, which are in this case also valid for the LS-estimator: (Eykhoff,(1974)) When there is no dependency between the noise and the input signal of the system, both estimators give us a consistent and asymptotically (i.e. $m \to \infty$) efficient estimate of M_k (cf. section 3.1).

The derivation of a maximum likelihood method for the model, used in chapter 4, is very similar to what we have done before in this section.

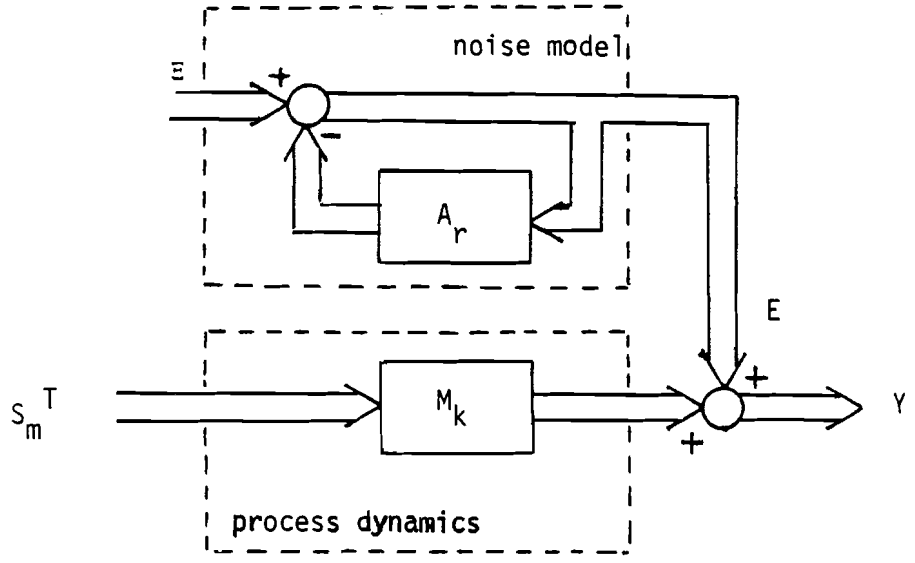


fig. 2.3.-2 an extended model for the estimation of Markov-parameters

This model is described by:

$$Y = S_m^T M_k + H_m^T A_r + \Xi \qquad \text{eq. 2.3.(7)}$$

where, again, Ξ is presumed to be Gaussian, white, channel independent noise. This extended model can be treated the same as the first model, which results in the same likelihood function L:

$$-\ln L = (m+1).\left(\frac{q}{2}\ln 2\pi + \tfrac{1}{2}\ln q + \ln\sigma\right) + \frac{1}{2\sigma^2}\,\mathrm{tr}\{(Y-\hat{Y})^T.(Y-\hat{Y})\} \qquad \text{eq. 5.1.(8)}$$

but here: $\hat{Y} = S_m^T.\hat{M}_k + H_m^T(\hat{M}_k).\hat{A}_r = \beta_m^T(\hat{N}).\hat{N}$

$$\frac{\partial(-\ln L)}{\partial \hat{N}} \quad \text{and} \quad \frac{\partial(-\ln L)}{\partial\sigma} = 0 \qquad \text{eq. 5.1.(13)}$$

Using the results of eq. 5.1.(10) and 5.1.(11), minimization of (-ln L) is obtained by that value of $\hat{N}$ that fulfils:

$$\frac{\partial \mathrm{tr}\{(Y-\hat{Y})^T.(Y-\hat{Y})\}}{\partial \hat{N}} = 0 \qquad \text{eq. 5.1.(14)}$$

This results in the minimization of the same non-quadratic function as in chapter 4. The estimate $\hat{N}_{ML}$, fulfilling eq. 5.1.(14), is identical to the solution of $\hat{N}_{LS}$ that is found when using one of the iterative procedures. The recursive method of 4.3, however, gives us an approximate estimate of $\hat{N}_{ML}$. Since the equation errors are not corrected during each update, it is not possible to guarantee that the maximum of the likelihood function is found. This means that the estimate given by the recursive method is in most cases not asymptotically efficient, but is a consistent and unbiased estimate.

The ML-estimate and the estimators provided by the iterative least squares methods are also asymptotically efficient, which means that among the unbiased estimators no other one has a smaller asymptotic (co)variance.

Estimation of Markov-parameters, wether or not with estimation of noise parameters, results in an unbiased estimate if the noise is independent

from the input signal of the system. This implies that we will get a biased estimate when the effects of truncation of the impulse response can not be neglected: This can be proved as follows:

Suppose $$Y = S_m^T.M_k + S_\infty^T.M_\infty + E \qquad \text{eq. 5.1.(15)}$$

where $S_\infty^T.M_\infty$ represents the 'tail' of the impulse response.

Rewriting eq. 5.1.(15) leads to:

$$M_k = \underbrace{(S_m S_m^T)^{-1}.S_m Y}_{\hat{M}_k} - (S_m S_m^T)^{-1}.S_m E - (S_m S_m^T) S_m S_\infty^T M_\infty \qquad \text{eq. 5.1.(16)}$$

Since $S_m S_m^T$ is positive definite, the estimate $\hat{M}_k$ will tend to M_k, and thus be unbiased, when:

$$\lim_{m\to\infty} S_m E = 0 \quad \text{and} \quad \lim_{m\to\infty} S_m S_\infty^T = 0 \qquad \text{eq. 5.1.(17)}$$

The first condition says that no dependence is allowed between the input signal of the system and the additive output noise and the second condition can only be fulfilled when $S_\infty^T=0$, i.e. that the effects of truncation of the impulse response may be neglected or when the input signal consists of white noise samples.

Concluding this section we may say that in the cases as described in chapter 3 and 4, the maximum likelihood method coincides with the least squares method.

5.2 An updating algorithm for ML-estimation of Markov-parameters

In this section, we will extend the model that was derived in 2.3.
In the model of 2.3 the following equation was valid:

$$\underline{e}(i) = A(1).\underline{e}(i-1) + \ldots + A(r)\underline{e}(i-r) + \underline{\xi}(i) \qquad \text{eq. 5.2.(1)}$$

From this equation we learn that the only dependence of the output of the colouring filter $\underline{e}(i)$ with the input noise $\underline{\xi}(i)$ at the same moment i, is

a dependence from input channel j to output channel j. Hence, there is no cross coupling from input channel j to output p at the same moment ($p \neq j$).
To be able to take into account such a direct coupling over the channels, eq. 5.2.(1) must be changed into:

$$\underline{e}(i) = A(1)\underline{e}(i-1) + \ldots + A(r)\underline{e}(i-r) + B(0).\underline{\xi}(i) \qquad \text{eq. 5.2.(2)}$$

where B(0) is a (q×q)-matrix that is a moving average (MA) parameter. Because of this, our model, now, contains 1 MA-parameter and a number of auto-regressive parameters.

The model described in 2.3 is now modified to that of fig. 5.2.-1

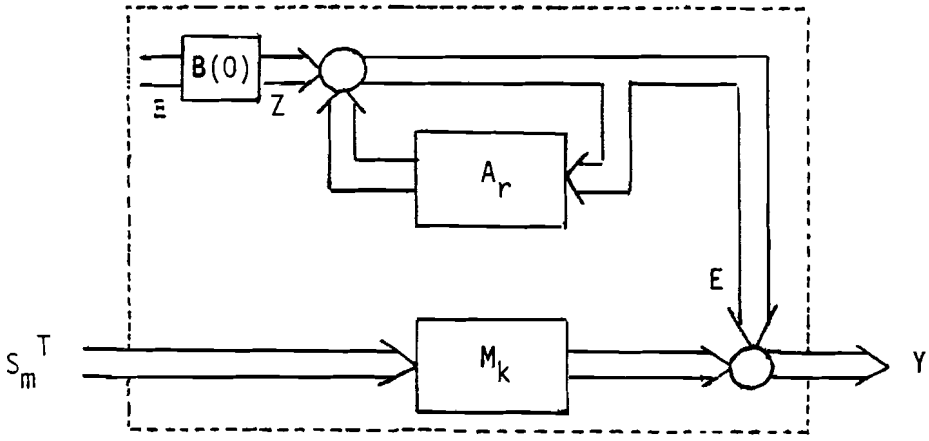


fig. 5.2.-1 model, extended for dependence over the noise channels

According to fig. 5.2.-1 and eq. 5.2.(2), we find:

$$Y = S_m^T.M_k + H_m^T.A_r + \underbrace{\Xi.B(0)^T}_{Z} \qquad \text{eq. 5.2.(3)}$$

In eq. 5.2.(3) Z is defined as:

$$Z = \Xi.B(0)^T \qquad \text{eq. 5.2.(4)}$$

In this equation Ξ is assumed to be white noise which is independent in time and place. Because of the multiplication of Ξ by $B(0)^T$, Z is white

noise that is only white in time, but not in place.

Using eq. 5.2.(3) we would like to estimate $\hat{M}_k$, $\hat{A}_r$ and $\hat{B}(0)$. When we would try to solve this problem like we did in the previous section, we would have to minimize a function, with respect to $\hat{M}_k$, $\hat{A}_r$ and $\hat{B}(0)$, that contains Ξ. This, however, would be a big problem, since we don't know Ξ.

In the previous section the input noise was assumed to have a covariance matrix $R=\sigma^2 I$. But the method used in the previous section will also work without this assumption. Hence, let us pay attention to the probability density function $p(Z)$ instead of $p(\Xi)$:

$$p(Z) = \frac{1}{2\pi^{\frac{q(m+1)}{2}}.(\det R)^{\frac{m+1}{2}}} \exp\left[-\tfrac{1}{2}\sum_{i=0}^{m} \underline{\zeta}^T(\ell+i).R^{-1}.\underline{\zeta}(\ell+i)\right] \qquad \text{eq. 5.2.(5)}$$

where $Z = (\underline{\zeta}(\ell),..,\underline{\zeta}(\ell+m))^T$ and R is the covariance matrix of $\underline{\xi}(i)$. Taking the logarithm of eq. 5.2.(5) results in:

$$-\ln L = \frac{(m+1)q}{2}\ln 2\pi + \frac{m+1}{2}.\ln \det R + \tfrac{1}{2}\sum_{j=\ell}^{\ell+m} \underline{\zeta}^T(j).R^{-1}.\underline{\zeta}(j) \qquad \text{eq. 5.2.(6)}$$

The likelihood function is now a function of R and of the parameters $\hat{M}_k$ and $\hat{A}_r$ since:

$$Z = Y - \hat{Y} = Y - S_m^T\hat{M}_k - H_m^T(\hat{M}_k).\hat{A}_r \qquad \text{eq. 5.2.(7)}$$

The necessary conditions which have to be fulfilled for the minimum of (-ln L) are:

$$\frac{\partial(-\ln L)}{\partial R} = 0 \quad \text{and} \quad \frac{\partial(-\ln L)}{\partial \hat{N}} = 0 \qquad \text{eq. 5.2.(8)}$$

where $\hat{N} = \begin{bmatrix}\hat{M}_k \\ \hat{A}_r\end{bmatrix}$.

Taking the derivative of (-ln L) with respect to R gives us:

$$\frac{\partial(-\ln L)}{\partial R} = \tfrac{1}{2}\sum_{j=\ell}^{\ell+m}(-(R^{-1})^T\underline{\zeta}(j).\underline{\zeta}^T(j)(R^{-1})^T) + \frac{m+1}{2}.(R^{-1})^T = 0$$

eq. 5.2.(9)

which can be rewritten to:

$$(R^{-1})^T.(\sum_{j=\ell}^{\ell+m}\underline{\zeta}(j).\underline{\zeta}^T(j)).(R^{-1})^T = (m+1).(R^{-1})^T \qquad \text{eq. 5.2.(10)}$$

This means that $R = \frac{1}{m+1}\sum_{j=\ell}^{\ell+m}\underline{\zeta}(j).\underline{\zeta}^T(j)$ eq. 5.2.(11)

is a necessary condition for minimization of -ln L.
Substitution of this necessary condition in eq. 5.2.(6) results in:

$$-\ln L = \overbrace{\tfrac{1}{2}\sum_{j=\ell}^{\ell+m}\underline{\zeta}^T(j).(\frac{1}{(m+1)}\sum_{k=\ell}^{\ell+m}\underline{\zeta}(k)\underline{\zeta}^T(k))^{-1}.\underline{\zeta}(j)}^{RHS_1} + \frac{(m+1)q}{2}\ln 2\pi +$$

$$+ \frac{m+1}{2}\ln\left[\det(\frac{1}{(m+1)}\sum_{k=\ell}^{\ell+m}\underline{\zeta}(k).\underline{\zeta}^T(k))\right] \qquad \text{eq. 5.2.(12)}$$

Since the first part of the right hand side of eq. 5.2.(12) is a scalar, it can be written as:

$$RHS_1 = \frac{(m+1)}{2}\ tr\{\sum_{j=\ell}^{\ell+m}\underline{\zeta}^T(j).(\frac{1}{(m+1)}\sum_{k=\ell}^{\ell+m}\underline{\zeta}(k).\underline{\zeta}^T(k))^{-1}.\underline{\zeta}(j)\}$$

eq. 5.2.(13)

using $tr(\underline{a}.\underline{a}^T) = tr(\underline{a}^T.\underline{a})$:

$$RHS_1 = \frac{m+1}{2}\ tr\left[\sum_{j=\ell}^{\ell+m}(\sum_{k=\ell}^{\ell+m}\underline{\zeta}(k).\underline{\zeta}^T(k))^{-1}.\underline{\zeta}(j).\underline{\zeta}^T(j)\right] \qquad \text{eq. 5.2.(14)}$$

$$= \frac{m+1}{2}\ tr\left[(\sum_{k=\ell}^{\ell+m}\underline{\zeta}(k).\underline{\zeta}^T(k))^{-1}.(\sum_{j=\ell}^{\ell+m}\underline{\zeta}(j).\underline{\zeta}^T(j))\right] \qquad \text{eq. 5.2.(15)}$$

$$RHS_1 = \frac{m+1}{2} \, tr[I] = \frac{(m+1)q}{2} \qquad \text{eq. 5.2.(16)}$$

Substituting this result in eq. 5.2.(12) gives:

$$-\ln L = \frac{m+1}{2} \left[1+\ln\left(\det\left(\frac{1}{m+1} \cdot \sum_{k=\ell}^{\ell+m} \underline{\zeta}(k)\underline{\zeta}^T(k)\right)\right)\right] + q.\ln 2\pi \qquad \text{eq. 5.2.(17)}$$

The minimum of $(-\ln L)$ can easily be seen to be the minimum of:

$$\det \sum_{k=\ell}^{\ell+m} \underline{\zeta}(k)\underline{\zeta}^T(k) = \det Z^T.Z \qquad \text{eq. 5.2.(18)}$$

So it is possible to find an estimate $\hat{N}$ by minimization of $\det(Z^T.Z)$ with respect to $\hat{N}$.
From eq. 5.2.(19):

$$\det Z^T.Z = \det(B(0).\Xi^T.\Xi.B(0)^T) = \sigma^2.\det\,(B(0).B(0)^T) \qquad \text{eq. 5.2.(19)}$$

it can be seen that this minimum does not results in an unique solution of $B(0)$, since it is always possible to include an orthogonal matrix $(W.W^T=I)$ that can be regarded as an amplification factor.

In chapter 4 we introduced an updating algorithm that minimized $tr(\Xi^T\Xi)$ with respect to $\hat{N}$; where Ξ was defined as:

$$\Xi = Y - \beta_m{}^T(\hat{N}).\hat{N}.$$

But in the case described above Z is defined as $Z = Y - \beta_m{}^T(\hat{N}).\hat{N}$.
Thus Z and Ξ coincide in these two different models.
This means that instead of minimizing the $tr(Y-\hat{Y})^T.(Y-\hat{Y})$ we must minimize $\det(Y-\hat{Y})^T.(Y-\hat{Y})$, which is only a small modification, to be able to cope with direct coupling over the noise channels.

The estimate of $\hat{N}$, that is obtained by this method, is again a maximum likelihood estimate, having the nice properties already mentioned in the previous section.

In the chapters 4 and 5, we have discussed some algorithms that are able to deal with coloured additive output noise. In chapter 7 we will present the results of simulations that have been obtained by the use of these algorithms.

CHAPTER 6. SOME ASPECTS OF THE REALIZATION PROBLEM

Until now we have only paid attention to the estimation of Markov-parameters from input and output data of a system.
Having determined such a sequence of Markov-parameters, it would be very attractive if we were able to find a set of matrices A,B and C, where A has a certain dimension n, in such a way that the reconstructed series of parameters, by using the definition eq. 2.1.(7):

$$M(i) = \begin{cases} 0 & , i<0 \\ D & , i=0 \\ C.A^{i-1}.B, & i>0 \end{cases} \qquad \text{eq. 2.1.(7)}$$

is, in some sense, as close as possible to the exact impulse response of the system. This problem is called the (naive) partial realization problem and the set of matrices {A,B,C} is called a realization. We shall not take into account the D-matrix, because this is immediately known, when given a series of Markov-parameters, since D=M(0).

In chapter 2 we introduced the Generalized Hankel matrix as a matrix consisting of Markov-parameters.
Here we shall give a definition of the Hankel matrix H[j] which is a part of the Generalized Hankel matrix:

$$H[j] = \begin{bmatrix} M(1) & M(2) & \cdots & M(j) \\ M(2) & M(3) & \cdots & M(j+1) \\ \vdots & \vdots & & \vdots \\ M(j) & M(j+1) & \cdots & M(L-1) \end{bmatrix} \qquad \text{eq. 6.(1)}$$

Using a noise-free Hankel matrix of the system of sufficient dimensions ($j>n$), Ho and Kalman introduced an algorithm that constructs a realization {A,B,C} of a linear, time-invariant, state-space model.
For the description of some other methods, which have a lot in common with the Ho-Kalman algorithm, we refer to an extensive report on rea-

lization methods of Van den Hof (1982).

In the deterministic case, i.e. when the Markov-parameters are not contaminated with noise, the Ho-Kalman algorithm leads to an exact solution: So we find an exact realization {A,B,C}, which has also a minimal dimension.
This dimension can be found from the rank of the Hankel matrix as follows from the theorem 1 and theorem 2: (Van den Hof (1982)).

Theorem 1: The sequence of Markov-parameters {M(k)}, k=1,2,... has a finite dimensional realization {A,B,C} if and only if there are an integer r and constants a_i such that

$$M(r+j) = \sum_{i=1}^{r} a_i \cdot M(r+j-i) \quad \text{for all } j>0 \qquad \text{eq. 6.(2)}$$

the minimum value of r is called the realizability index.

Theorem 2: If {M(k)} has a finite dimensional realization then

$$n = \text{rank } H[r] \qquad \text{eq. 6.(3)}$$

where n is the minimal dimension of the state space, and H[r] is the Hankel matrix built up from 2r-1 Markov-parameters.

These theorems imply that, in the noise free case, no extra information is added to the Hankel matrix when it is built up from a sequence of Markov-parameters with length $L \geq 2r-1$:

$$n = \text{rank } H[r] = \text{rank } H[r+k] \qquad \text{for all } k \geq 0 \qquad \text{eq. 6.(4)}$$

The basic idea of the Ho-Kalman algorithm is the following:

When given a series of Markov-parameters {M(k)}, k=1,...,L+1, having a finite dimensional realization, the Hankel matrix H[r] can be decomposed in the following way, using the definition of the Markov-parameters (eq. 2.1.(5)):

$$H[r] = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{r-1} \end{bmatrix} . \begin{bmatrix} B & AB & A^2B & . & . & A^{r-1}B \end{bmatrix} \qquad \text{eq. 6.(5)}$$

$$= \Gamma[r] \; . \; \Delta[r]$$

where $\Gamma[r]$ is the r-observability matrix and
$\Delta[r]$ the r-controllability matrix.

We assume that {A,B,C} forms a minimum realization of the given sequence of dimension n.
According to eq. 6.(4) the Hankel matrix has a rank equal to n, and therefor both $\Gamma[r]$ and $\Delta[r]$ have at least rank n. Because we only take into consideration observable and controllable systems, the rank of both matrices can not exceed n, which leads to:

$$\text{rank } \Gamma[r] = \text{rank } \Delta[r] = n \qquad \text{eq. 6.(6)}$$

Resuming we can say that we need to decompose the Hankel matrix H[r] in two matrices, both of full rank.
Having achieved this, it is possible to obtain the matrices C and B directly from respectively the first q rows of the first matrix and the first p columns of the second matrix found by decomposition.
To obtain matrix A, we need a shifted matrix, indicated by an arrow. A shift of one block row is indicated by a vertically pointing arrow, and a shift of one block column by a horizontally pointing one.
This leads to:

$$H[j]\uparrow = \overleftarrow{H}[j] = \begin{bmatrix} M_2 & M_3 & M_4 & . & . & M_{j+1} \\ M_3 & M_4 & M_5 & . & . & M_{j+2} \\ . & . & & . & & . \\ . & . & & & . & . \\ M_{j+1} & M_{j+2} & M_{j+3} & . & . & M_{L+1} \end{bmatrix} \qquad \text{eq. 6.(7)}$$

and $$H\uparrow = \overleftarrow{H} = \Gamma . A . \Delta \qquad \text{eq. 6.(8)}$$

From eq. 6.(8) A can be obtained by using the pseudo-inverse Γ^+ of Γ and Δ^+ of Δ:

$$A = \Gamma^+ . \overleftarrow{H} . \Delta^+ \qquad \text{eq. 6.(9)}$$

From the construction of the shifted Hankel matrix (eq. 6.(7)) it can be seen that we need an extra Markov parameter M(L+1).

The triplet (A,B,C) found by this procedure is only one possibility of an equivalence class of the system under study.
This can be seen easily by extending the decomposition of eq. 6.(5) by an invertable matrix T:

$$H[r] = \Gamma[r].T^{-1}.T.\Delta[r] = (\Gamma[r].T^{-1}).(T.\Delta[r]) \qquad \text{eq. 6.(10)}$$

which results in $C^*=C.T^{-1}$ and $B^*=T.B$ and (using 6.(9)) $A^*=T.A.T^{-1}$.
This triplet (A^*,B^*,C^*) is equivalent to (A,B,C) since it gives us the same sequence of Markov parameters.

According to Hajdasinski and Damen (1979), we can use the Singular Value Decomposition to decompose the Hankel matrix as follows:

$$H = W . \Sigma . V^T \qquad \text{eq. 6.(11)}$$

where H: is a $g \times \ell$ matrix

ρ: $\rho = \min(g,\ell)$

W: is a $g \times \rho$ matrix, consisting of ρ orthonormal eigenvectors of $H.H^T$

V: is a $\ell \times \rho$ matrix, consisting of ρ orthonormal eigenvectors of $H^T.H$

Σ: is a $\rho \times \rho$ diagonal matrix: $\text{diag}(\delta_1,\delta_2,\ldots,\delta_\rho)$

$\delta_1 \geq \delta_2 \geq \ldots \geq \delta_\rho \geq 0$

and δ_i is the square root of an eigen-value of $H^T.H$ (or $H.H^T$), and is called a singular value.

Because of the special properties of W, V and Σ, mentioned above, we can write:

$$\text{rank } H = \text{rank } \Sigma \leq \rho \qquad \text{eq. 6.(12)}$$

This means that the rank of H is equal to the number of singular values unequal zero.
Because all singular values unequal zero are positive, eq. 6.(11) can be written as:

$$H = W \cdot \Sigma^{\frac{1}{2}} \cdot \Sigma^{\frac{1}{2}} \cdot V^T \qquad \text{eq. 6.(13)}$$

where

$$\Sigma^{\frac{1}{2}} = \begin{bmatrix} \delta_1^{\frac{1}{2}} & 0 & \cdots & 0 \\ \cdot & \cdot & & \cdot \\ \cdot & & \delta_n^{\frac{1}{2}} & \cdot \\ \cdot & & 0 & \cdot \\ 0 & \cdot & \cdot & 0 \end{bmatrix} (\rho \times \rho) \qquad \text{eq. 6.(14)}$$

and n = rank H.

Choosing $\Gamma = W.\Sigma^{\frac{1}{2}}$ and $\Delta = \Sigma^{\frac{1}{2}}.V^T$, we have found a decomposition of H into two matrices of full rank n, which is the product of observability and controllability matrix.
Thus, by using SVD it is possible to find an observability and controllability matrix that can be used in the Ho-Kalman algorithm to obtain a realization {A,B,C} of the system under consideration.

Until so far, we have only treated the deterministic case, but we want to find a realization built of estimated Markov-parameters which are, of course, disturbed by noise.
In this noisy situation, there does not exist anymore a finite dimensional realization that reconstructs {M(k)}, k=1,2,... in an exact way.
The rank of the Hankel matrix will, therefore, always increase when more Markov-parameters are used to build up that Hankel matrix.
The realization problem can now be formulated as finding a finite dimensional (n) realization that generates a sequence of Markov-parameters, that is, in some way, as close as possible to the given sequence (of

length L>>n) of estimated Markov-parameters. This can be obtained by reduction of the rank of the Hankel matrix to rank n. This reduction can be regarded as a noise filter that operates on the Hankel matrix and thus on the Markov-parameters, in such a way that we can reconstruct {M(k)}, k=1,2,... with a finite dimensional realization.
A reduction of the rank of the Hankel matrix can be obtained by neglecting the smallest singular values and putting them to zero, because they are considered to be caused by the noise of the estimated parameters. This is illustrated for an example with n=3 (Van den Hof (1982)) in fig. 6.-1 .

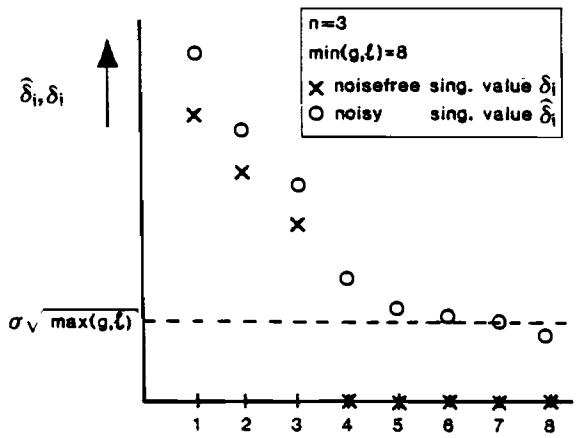


fig. 6.-1 singular values of a 3rd order system in the deterministic and in the noisy case.

where $\sigma\sqrt{\max(g,\ell)}$ is the expected noise level on the Markov-parameters.

One of the reasons for choosing a model based on Markov-parameters in section 2.2 was that we did not need any knowledge about the order of the system beforehand. Now, it appears that this characteristic of the system under study is found automatically by solving the realization problem.
For more extensive information about the realization problem we refer to publications of Van den Hof, Hajdasinski and Damen (cf. the list of references).

CHAPTER 7 RESULTS OF SIMULATIONS

In the chapters 3, 4 and 5 we have described some algorithms for the estimation of Markov-parameters. To be able to test the qualities of these algorithms, many simulations have been done. A great advantage of simulations above estimations on real data is that all external influences on the simulated process can be controlled. This means that we can choose the properties of the systems, used for simulation, or of the input data in such a way that the results of the simulations are well suited for drawing conclusions on the properties of the methods.

In section 7.1 we shall give a description of the systems that we used for the simulations, a short enumeration of the methods used for estimating the Markov-parameters and the criteria for the quality of a method. The sections 7.2-7.5 deal with the results of the experiments as obtained by simulations of 4 systems. Finally, we shall give a short summary of the results in section 7.6.

7.1 Description of simulations and processing

To be able to estimate the Markov-parameters of a system, we need, of course, input and output signals. Because of the assumptions we made about the character of the noise (cf. chapter 2), we suppose that only the output is corrupted by noise. This means that we use an undisturbed input signal for the system and an output signal, that is the sum of the response on this input signal and additive output noise that may be coloured noise.
After an input and output signal have been simulated, an estimation procedure is executed, estimating a number of Markov-parameters of the process and auto-regressive parameters of the noise colouring filter. These estimated Markov-parameters will be denoted as $\{\hat{M}(i)\}_L$, $i=1,2,\ldots,L$ in contra-distinction to the exact Markov-parameters which are denoted as $\{M(i)\}_L$, $i=1,2,\ldots,L$.
A measure for the quality of the estimated parameters is the relative error in the estimated Markov-parameters sequence: ERRO (see also Van den Hof (1982)):

$$ERRO = \frac{\sum_{i=1}^{L} \|\hat{M}(i) - M(i)\|_E^2}{\sum_{i=1}^{L} \|M(i)\|_E^2} \qquad \text{eq. 7.1.(1)}$$

From these estimated parameters a Hankel matrix is built and the realization algorithm, as described in chapter 6, is applied to this Hankel matrix. This algorithm results in a realization $\{\tilde{A},\tilde{B},\tilde{C}\}$ that generates a sequence of Markov-parameters $\{\tilde{M}(i)\}_L$, $i=1,2,\ldots,L$ (filtered by the realization algorithm).
The error of these filtered Markov-parameters in comparison to the exact values is defined as (cf. Van den Hof (1982)):

$$ERR1 = \frac{\sum_{i=1}^{L} \|\tilde{M}(i) - M(i)\|_E^2}{\sum_{i=1}^{L} \|M(i)\|_E^2} \qquad \text{eq. 7.1.(2)}$$

Schematically, the simulation and the identification can be presented as is shown in fig. 7.-1, with the following explanation:

1. The input signals, that were used for simulation, existed of 3 kinds of signals:
   - a white Gausian noise sequence with $\sigma=1.0$ and $\mu=0$
   - a Pseudo Random Binairy Noise Sequence with an amplitude switching between -1 and +1.
   - a Pseudo Random Binary Noise Sequence that is filtered by a first order low pass filter.

2. The input signals of the noise filter were sequences of white, channel independent Gaussian noise with $\sigma=0.1$ and $\mu=0$.

3. To simulate the process the following systems were chosen:

   SYSTEM 1 , has 2 inputs, 2 outputs and dimension 2:

$$A = \begin{bmatrix} .3 & 0. \\ 0. & .2 \end{bmatrix}, \quad B = C = \begin{bmatrix} 1. & 0. \\ 0. & 1. \end{bmatrix} \qquad \text{eq. 7.1.(3)}$$

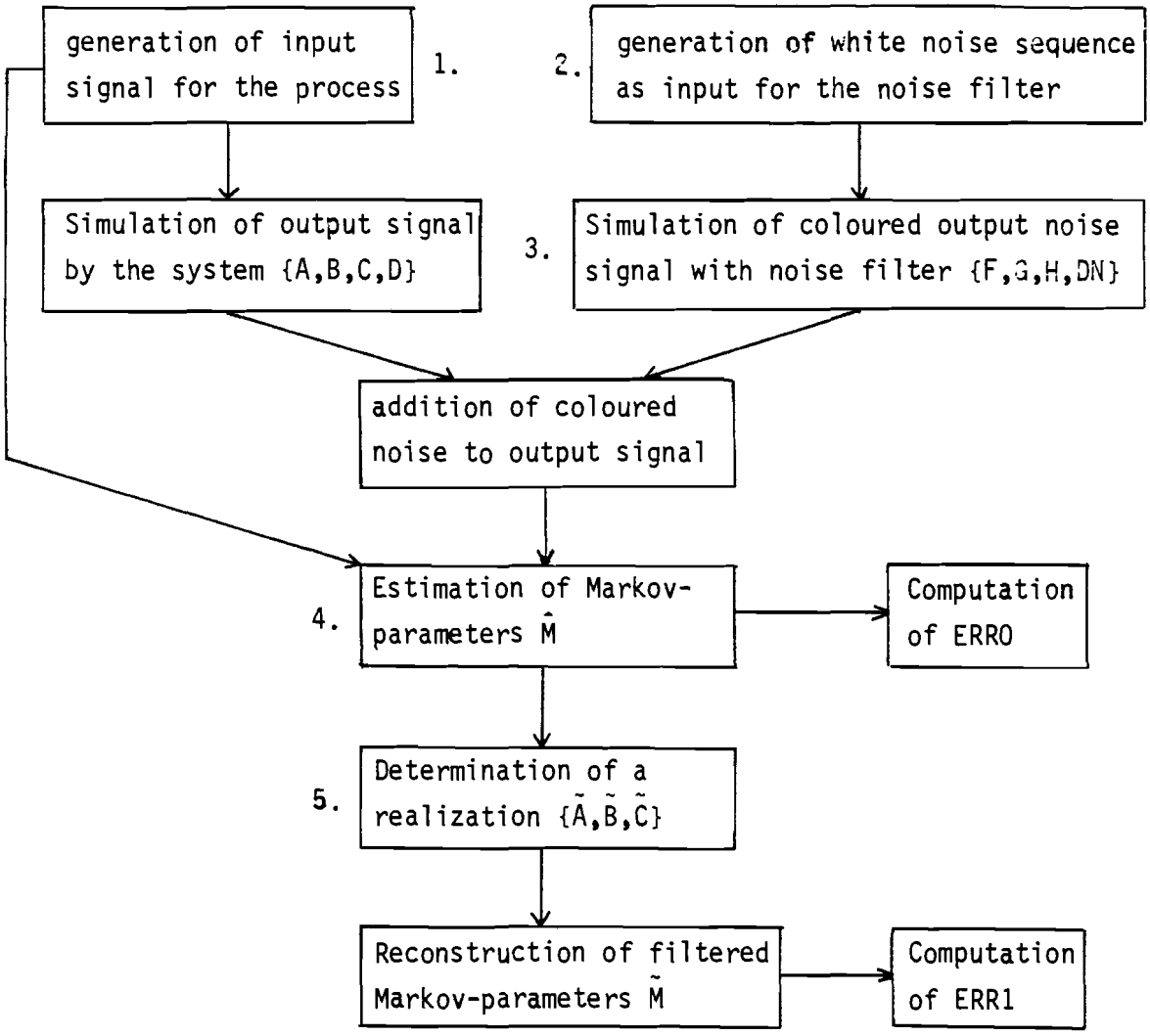


fig. 7.-1 flow chart of simulations and identification

SYSTEM 2 and SYSTEM 3 have both 3 inputs, 2 outputs, dimension 4 and the same B- and C-matrix (according to Damen and Hajdasinski (1982)):

$$B = \begin{bmatrix} 1. & 0. & 1. \\ -1 & .5 & .5 \\ 0. & 1. & -.5 \\ 0. & .5 & -1. \end{bmatrix} \qquad C = \begin{bmatrix} .5 & 0. & 1. & 1. \\ 1. & -.5 & .5 & -.5 \end{bmatrix} \qquad \text{eq. 7.1.(4)}$$

For a clear view on the eigenvalues of the systems, the A-matrices are

chosen in a diagonal way:

SYSTEM 2: A = diag(0.7, 0.6, 0.2, 0.1)

SYSTEM 3: A = diag(0.9, 0.8, 0.3, 0.2) eq. 7.1.(5)

For the simulations we have restricted ourselves to a strictly proper system which means that the D-matrix of the system is chosen to be zero.

As is seen very easily from eq. 7.1.(3) SYSTEM 1 is a MIMO-system consisting of two decoupled SISO-systems in parallel.
For SYSTEM 2 and SYSTEM 3 we have chosen 2 large and 2 small eigenvalues to be able to find out what happens to the smallest eigenvalues after estimation.
The biggest eigenvalues of SYSTEM 3 are so large that we deal with a very slowly decreasing impulse response. This implies that we should take into consideration many Markov-parameters to avoid any influence of the 'tail' of the truncated impulse response. When not enough Markov-parameters are estimated, we expect to notice this tail influence in the estimation results.

For the noise colouring filter we have also chosen 3 systems:

NOISE SYSTEM 1, having 2 inputs, 2 outputs and dimension 2:

$$F = \begin{bmatrix} 0.85 & 0.0 \\ 0.0 & 0.75 \end{bmatrix} \qquad G = H = \begin{bmatrix} 1.0 & 0.0 \\ 0.0 & 1.0 \end{bmatrix} \qquad \text{eq. 7.1.(6)}$$

and an input-output matrix DN equal to zero. This noise system is also a combination of 2 decoupled SISO-systems in parallel. The noise generated by this system is only used in combination with SYSTEM 1, where the noise is low frequent with respect to the system at hand.

NOISE SYSTEM 2, has also 2 inputs, 2 outputs and dimension 2:

$$F = \begin{bmatrix} 0.3 & 0.0 \\ 0.0 & 0.2 \end{bmatrix} \qquad G = H = \begin{bmatrix} 1.0 & 0.0 \\ 0.6 & 0.5 \end{bmatrix} \qquad \text{eq. 7.1.(7)}$$

and again DN equal to zero. The noise coloured by this system is added to the outputs of SYSTEM 2 and SYSTEM 3 and is no longer channel independent.

NOISE SYSTEM 3 is the same as NOISE SYSTEM 2 except for the input-output matrix, that is now unequal to zero:

$$DN = B(0) = \begin{bmatrix} 1.0 & 0.3 \\ 0.4 & 0.6 \end{bmatrix} \qquad \text{eq. 7.1.(8)}$$

The noise coloured by this system is only added to the output of SYSTEM 2.

4. The algorithms applied to the simulated input and output data are based on the methods described in chapters 4 and 5. This implies that in all cases we will deal with algorithms that are able to estimate auto-regressive parameters of a noise colouring filter. To recapitulate the chosen methods and the names given to the algorithms, we present the following list:

method nr.	name	explanation
1	RECMAK1	Estimation procedure based on the recursive algorithm as described in section 4.3.
2	MARKEST-1	Estimation procedure based on the updating algorithm that solves: $\beta_m.Y=\beta_m.\beta_m^T.\hat{N}$, as described in section 4.4; only 1 iteration is executed.
3	EXACTMARK	Procedure based on the explicit iterative algorithm that solves: $Y=\beta_m^T.\hat{N}$, as described in section 4.2; only 1 iteration is executed.
4	MARKEST-25	Estimation procedure based on the updating algorithm of 4.4. This method is exactly the same as method 2, except for the fact that MARKEST-25 carries out 25 iterations.

5 MARKMIN This estimation procedure minimizes the loss function $tr\{\Xi^T.\Xi\}$ by the use of 'hill-climbing'-techniques as described in 4.4. The parameters estimated by MARKEST-1 are used as a starting value.

6 MARKMIND Estimation procedure based on the maximum likelihood principle as depicted in section 5.2. In this algorithm the loss function $\det(Z^T.Z)$ is minimized (...MIND) by using a 'hill-climbing'-technique and the parameters estimated by MARKEST-1 are chosen as a starting value.

5. To the estimated Markov-parameters a realization algorithm is applied based on a decomposition of the Hankel matrix. We used the Ho-Kalman algorithm, modified by Damen/Hajdasinski (Van den Hof (1982)) as was briefly described in chapter 6.

Finally, it can be remarked that all experiments have been carried out for 20 different input-output signals sequences to be able to draw reasonable conclusions from the results obtained by the estimation procedures and the realization algorithm.
The experiments were done on a VAX 11/750 computer and double precision arithmetics were used throughout all computations.

7.2 Results of simulations using SYSTEM 1

SYSTEM 1 has small eigenvalues (cf. eq. 7.1.(3)) and thus a fast decreasing impulse response. This implies that we don't have to estimate many Markov-parameters to avoid 'tail'-influences.
Two kinds of input signals have been used to excitate SYSTEM 1:

- white noise signals (σ=1.0, μ=0.0)
- Pseudo Random Binairy noise sequences (PRBNS) with an amplitude switching between +1.0 and -1.0 (switch time $\geq$ sample time).

The responses on these input signals have been disturbed by noise that was coloured by NOISE SYSTEM 1. The input signal of this noise system was white, channel independent noise with σ=0.1 and μ=0.0.

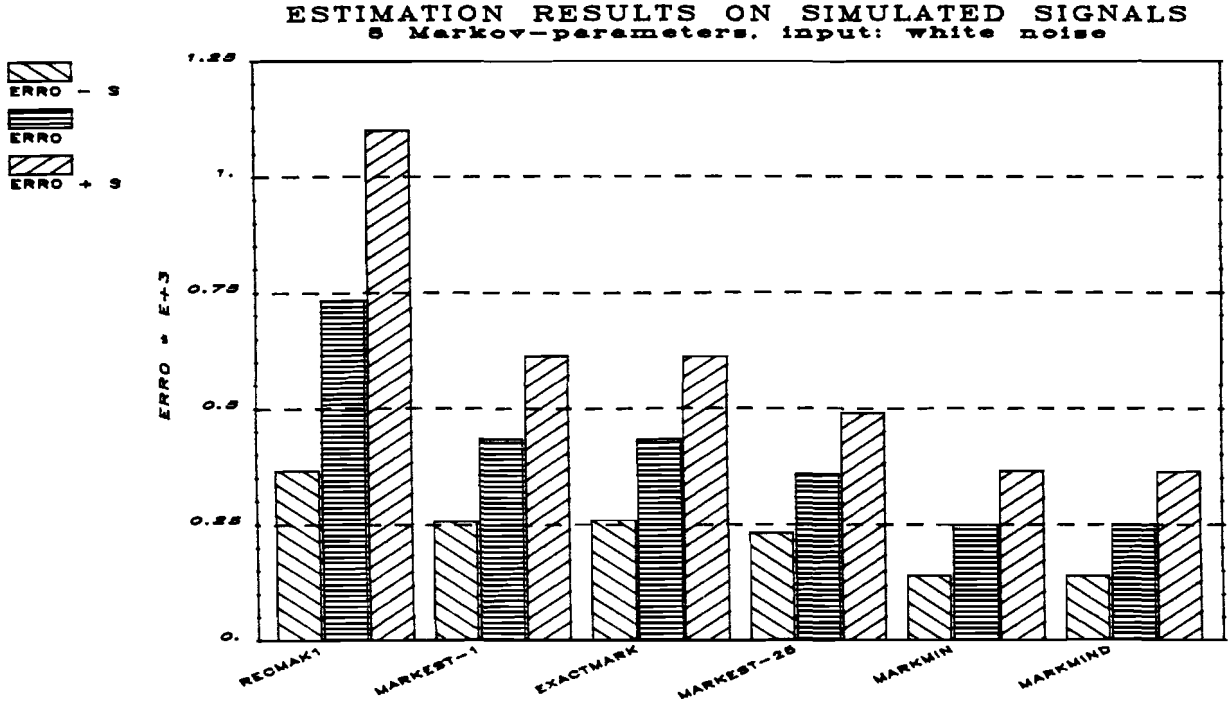
Comparing the properties of SYSTEM 1 and NOISE SYSTEM 1, we see that a high-frequent output signal is distorted by a low-frequent noise signal. The signal-to-noise ratio at the output is approximately 3. This means that we have a high noise level on the output signal and when estimating 8 Markov-parameters, the noise due to 'tail'-influences is small compared to the added output noise. Estimation of 2 auto-regressive noise parameters will be sufficient to give an exact description of NOISE SYSTEM 1.

First we shall give the estimation results when the input signal consists of 20 white noise sequences of 1000 samples each.

	RECMAK1	MARKEST-1	EXACTMARK	MARKEST-25	MARKMIN	MARKMIND
$\overline{ERRO}^{20}$	0.7325 E-3	0.4348 E-3	0.4352 E-3	0.3602 E-3	0.2512 E-3	0.2514 E-3
s.d.ERRO	0.3672 E-3	0.1777 E-3	0.1764 E-3	0.1294 E-3	0.1126 E-3	0.1128 E-3

tab. 7.2./1 Estimation results of SYSTEM 1; input: white noise

fig. 7.2.-1

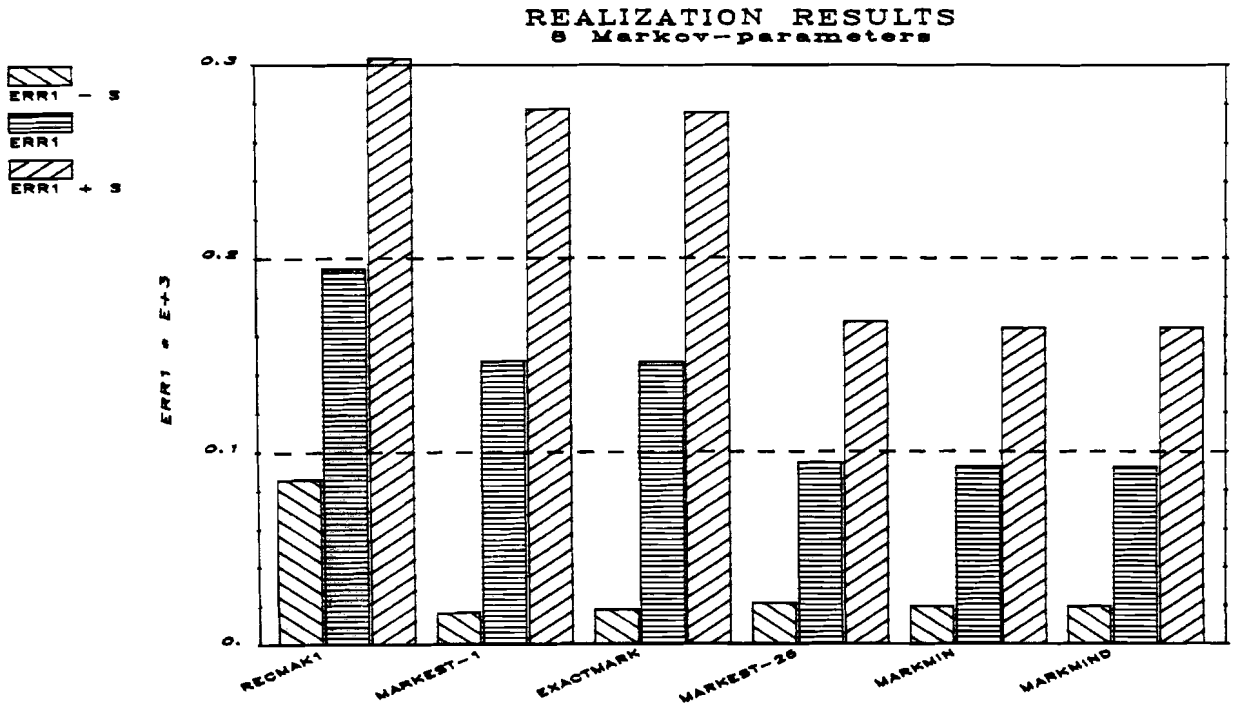


SYSTEM 1

In all bar charts in this chapter the mean values of errors in the estimated (ERR0) or in the reconstructed (ERR1) Markov-parameters will be presented together with this value $\pm$ its standard deviation (s) based on 20 simulations.
As can be seen from these results, the recursive method RECMAK1 gives us the least accurate result. An obvious improvement in the accuracy of the estimated parameters can be obtained by using one of the other two non-iterative methods: MARKEST-1 or EXACTMARK.
As was depicted in chapter 4, in case of a well conditioned input signal these two algorithms should give the same result and this can, indeed, be noticed from these results.
The method MARKEST-25, using 25 iterations by successive substitution, gives a better estimate then the non-iterative methods, while MARKMIN and MARKMIND, that minimize the loss function by 'hill-climbing'-techniques, give the best estimates. From this it can be concluded that 25 iterations were not sufficient to reach the minimum of the loss function. The improvements per iteration in the estimates, however, are so small that it can be doubted whether the exact minimum would be reached, in this case, using successive substitution. The energy represented by $tr[\Xi^T.\Xi]$, that is found by these iterative methods agrees well with the energy of the white noise that was supplied to the noise system.
In this case, where the input noise of the noise filter is channel independent, both MARKMIN and MARKMIND give the same results, as could be expected from the theory.
After analyzing the estimation results, we shall now take a look at the quality of the reconstructed Markov-parameters after noise filtering by the realization algorithm of chapter 6 (tab. 7.2./2 and fig. 7.2.-2).

	RECMAK1	MARKEST-1	EXACTMARK	MARKEST-25	MARKMIN	MARKMIND
$\overline{ERR1}^{20}$	0.1944 E-3	0.1467 E-3	0.1465 E-3	0.9401 E-4	0.9161 E-4	0.9173 E-4
s.d.ERR1	0.1086 E-3	0.1304 E-3	0.1287 E-3	0.7287 E-4	0.7204 E-4	0.7210 E-4

tab. 7.2./2 Realization results of SYSTEM 1; input: white noise



SYSTEM 1

fig. 7.2.-2 Realization results of SYSTEM 1; input:white noise

In fig. 7.2.-2 we notice that the differences in the quality of the Markov-parameters, as obtained by the use of the different estimation algorithms, have decreased. This is due to the filtering effect of the realization algorithm.

The same experiment has been repeated for the case that the input signals were PRBN-sequences. The results of the experiment are presented in tab. 7.2./3 and fig. 7.2.-3.

	RECMAK1	MARKEST-1	EXACTMARK	MARKEST-25	MARKMIN	MARKMIND
$\overline{ERRO}^{20}$	0.8016 E-3	0.3399 E-3	0.3400 E-3	0.3000 E-3	0.2599 E-3	0.2595 E-3
s.d.ERRO	0.4532 E-3	0.1055 E-3	0.1054 E-3	0.1131 E-3	0.1093 E-3	0.1092 E-3

tab. 7.2./3 Estimation results of SYSTEM 1; input: PRBNS

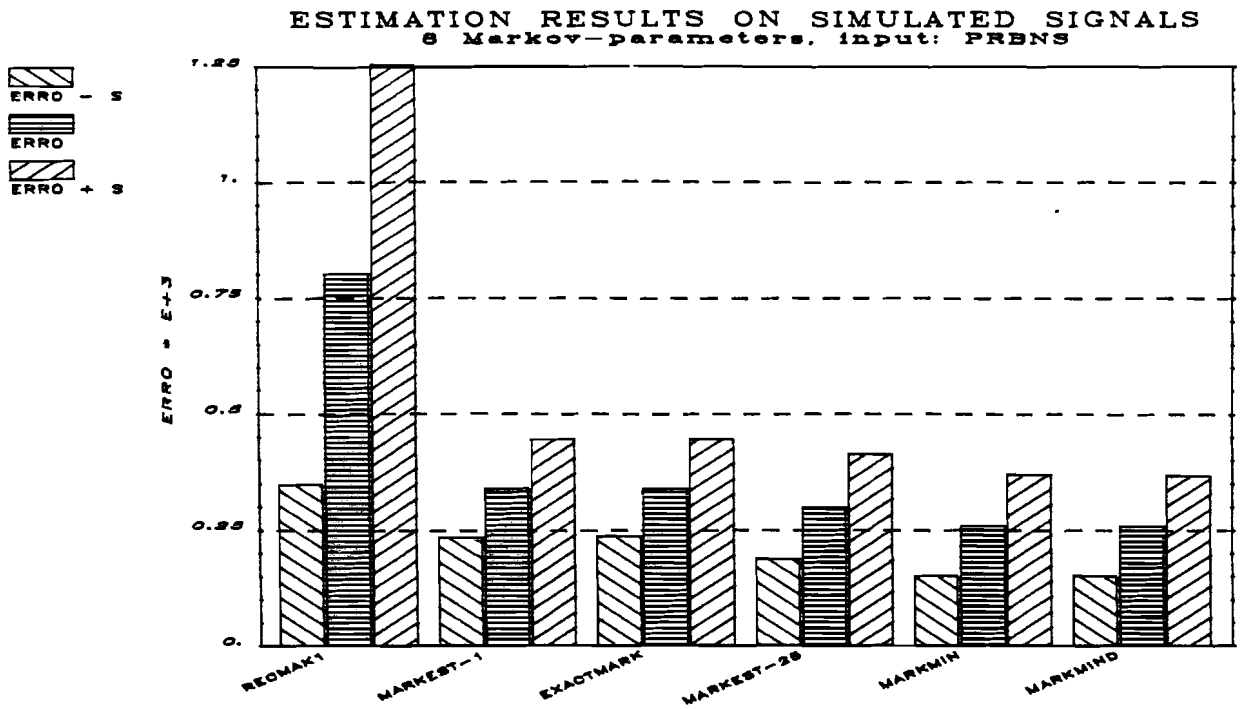


fig. 7.2.-3 Estimation results of SYSTEM 1; input: PRBNS

The results of the realization are shown in tab. 7.2./4 and fig. 7.2.-4.

	RECMAK1	MARKEST-1	EXACTMARK	MARKEST-25	MARKMIN	MARKMIND
$\overline{ERR1}^{20}$	0.2003 E-3	0.1358 E-3	0.1338 E-3	0.9828 E-4	0.9624 E-4	0.9606 E-4
s.d.ERR1	0.1352 E-3	0.8469 E-4	0.8376 E-4	0.7203 E-4	0.7087 E-4	0.7082 E-4

tab. 7.2./4 Realization results of SYSTEM 1; input: PRBNS

This experiment gives us similar results as the first one with white noise as input signal and this is caused by the fact that the PRBN-sequences were not filtered, which means that enough information was present in these input signals to fully excitate SYSTEM 1.

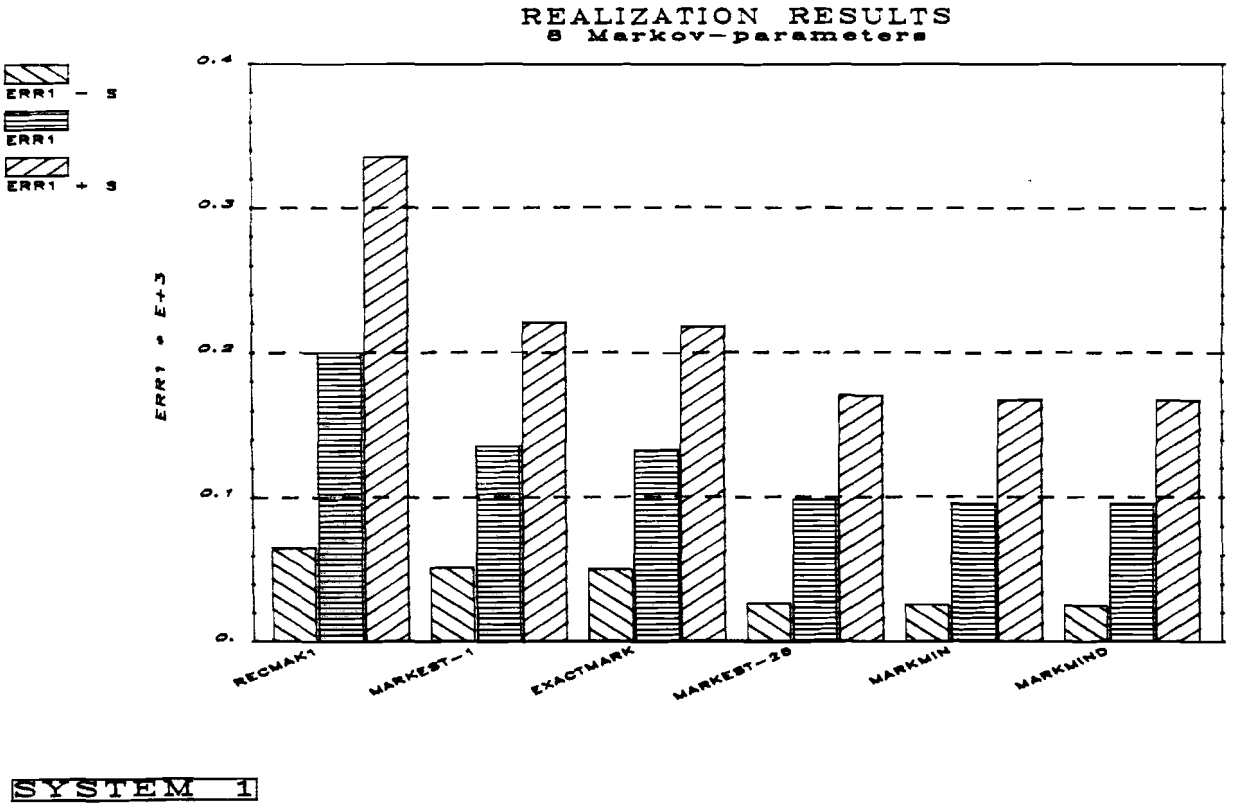


fig. 7.2.-4 Realization results of SYSTEM 1; input: PRBNS

From the results of both experiments we can conclude that the use of a non-recursive method results in better estimates of the Markov-parameters and in case of a bad S/N-ratio, use of iterative methods can further improve the estimated parameters.

7.3 Experiments with SYSTEM 2

In the previous section we have discussed the situation of a bad Signal-to-Noise ratio at the output. Now, we will take a look at the effects when we deal with a low noise level at the output. SYSTEM 2 is a 4^{th}-order system with two large eigenvalues (0.7 and 0.6) and two small ones (0.2 and 0.1). To avoid much influence of 'tail'-effects, 14 Markov-parameters have been estimated.
To excitate SYSTEM 2, 2 kinds of input signals have been chosen:

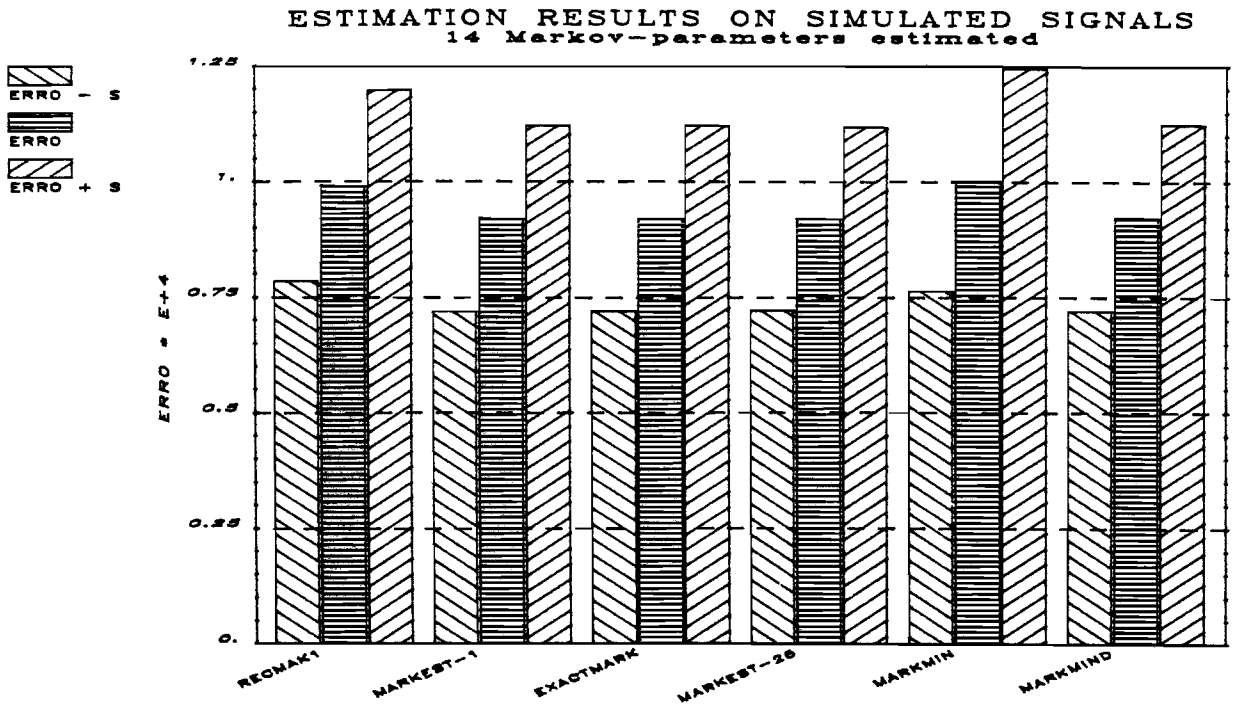
- white noise signals (σ=1.0, μ=0.0).
- filtered Pseudo-Random binairy noise sequences (PRBNS)

The PRBN-sequences are filtered using a low pass filter with such a characteristic that a part of the energy at frequencies needed to excitate the smallest eigenvalues has disappeared. The responses on these input signals have been disturbed by noise that was coloured by NOISE SYSTEM 2. The input noise of this noise filter was white Gaussian and channel independent, with σ=0.1 and μ=0.0.
This results in a S/N-ratio that is greater than 10. The noise level is so small that the 'tail'-effects are of the same magnitude as the noise influences. Here, also 2 auto-regressive noise parameters have been estimated. The estimation results found by estimating the Markov-parameters of SYSTEM 2 can be found in tab. 7.3./1 and fig. 7.3.-1.

	RECMAK1	MARKEST-1	EXACTMARK	MARKEST-25	MARKMIN	MARKMIND
$\overline{ERRO}^{20}$	0.9904 E-4	0.9201 E-4	0.9206 E-4	0.9198 E-4	0.1004 E-3	0.9223 E-4
s.d.ERRO	0.2051 E-4	0.1998 E-4	0.2004 E-4	0.1983 E-4	0.2403 E-4	0.2026 E-4

tab. 7.3./1 Estimation results of SYSTEM 2; input: white noise

From these results we can again conclude that the recursive method gives a result which is worse than the other 2 non-iterative methods.
It is also clear that, in this case, the use of an iterative method doesn't give any significant improvement. MARKMIN even gives a result that is a little worse than that of the recursive method. An explanation of this phenomenon will be given in section 7.4.
The deviation in the estimated AR-parameters of the noise model is smaller than 5% and the $tr[\Xi^T.\Xi]$ that is found, agrees with the noise level that was present at the input of the noise filter.



SYSTEM 2

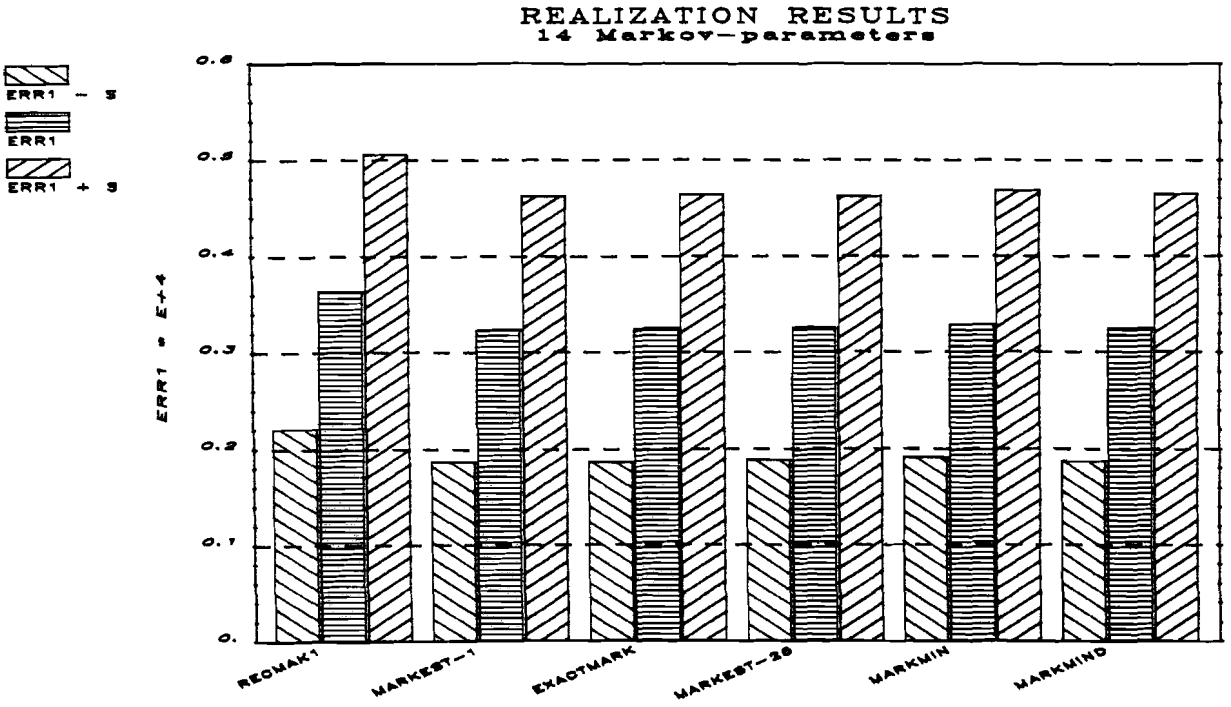
fig. 7.3.-1 Estimation results of SYSTEM 2; input: white noise

The results of the realization, using the estimated Markov-parameters, are presented in tab. 7.3./2 and fig. 7.3.-2.

	RECMAK1	MARKEST-1	EXACTMARK	MARKEST-25	MARKMIN	MARKMIND
$\overline{ERR1}^{20}$	0.3638 E-4	0.3247 E-4	0.3251 E-4	0.3252 E-4	0.3297 E-4	0.3254 E-4
s.d.ERR1	0.1419 E-4	0.1376 E-4	0.1389 E-4	0.1370 E-4	0.1387 E-4	0.1387 E-4

tab. 7.3./2 Realization results of SYSTEM 2; input: white noise

From these results we can draw the same conclusions as in section 7.3, which means that the differences between the estimation results are smoothened by the realization algorithm.



SYSTEM 2

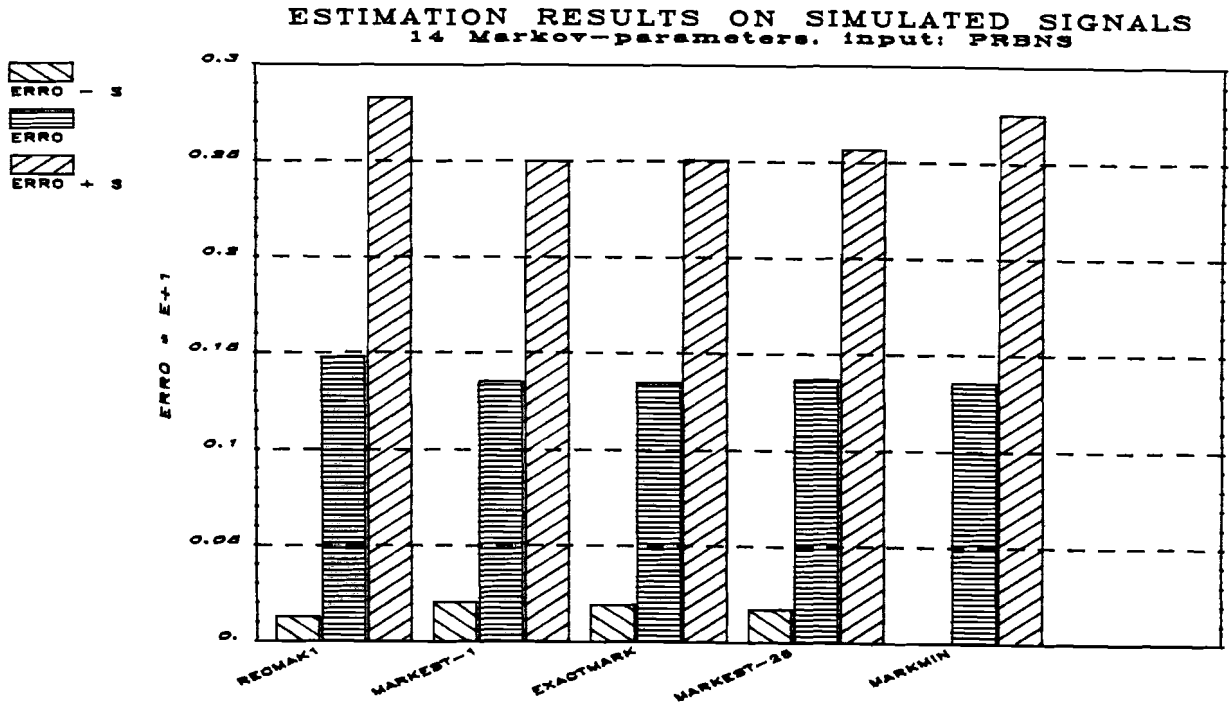
fig. 7.3.-2 Realization results of SYSTEM 2; input: white noise

Now, we shall pay attention to an experiment where the input signals are filtered PRBN-sequences. In this experiment we did not apply MARKMIND, because it would, in this case, give a result similar to that of MARKMIN. The estimation results are shown in tab. 7.3./3 and fig. 7.3.-3.

	RECMAK1	MARKEST-1	EXACTMARK	MARKEST-25	MARKMIN
$\overline{\text{ERRO}}^{20}$	0.1479 E-1	0.1351 E-1	0.1347 E-1	0.1368 E-1	0.1354 E-1
s.d.ERRO	0.1348 E-1	0.1147 E-1	0.1157 E-1	0.1196 E-1	0.1396 E-1

tab. 7.3./3 Estimation results of SYSTEM 2; input: filtered PRBNS

Looking at these results, the first thing we notice is that the error ERRO is much larger than in the first experiment, where was dealt with white noise as input. This is caused by the character of the filtered PRBN-signal,



SYSTEM 2

fig. 7.3.-3 Estimation results of SYSTEM 2; input: filtered PRBNS

because it does not contain enough energy of high frequenties to fully excitate the process.

From the results of the realization, tab. 7.3./4 and fig. 7.3.-4, we learn that application of the realization algorithm to the estimated parameters even worsens the Markov-parameters when the dimension is chosen equal to the dimension of SYSTEM 2 (n=4).

	RECMAK1	MARKEST-1	EXACTMARK	MARKEST-25	MARKMIN
$\overline{ERR1}^{20}$	0.1899 E-1	0.1805 E-1	0.1801 E-1	0.1816 E-1	0.1770 E-1
s.d.ERR1	0.1256 E-1	0.1095 E-1	0.1103 E-1	0.1135 E-1	0.1286 E-1

tab. 7.3./4 Realization results of SYSTEM 2; input: filtered PRBNS

Looking at one of the estimated impulse responses, fig. 7.3.-5, we see that oscillations around the exact impulse response occur with a frequency equal to the sampling frequency. This phenomenon is found in all results of this experiment.

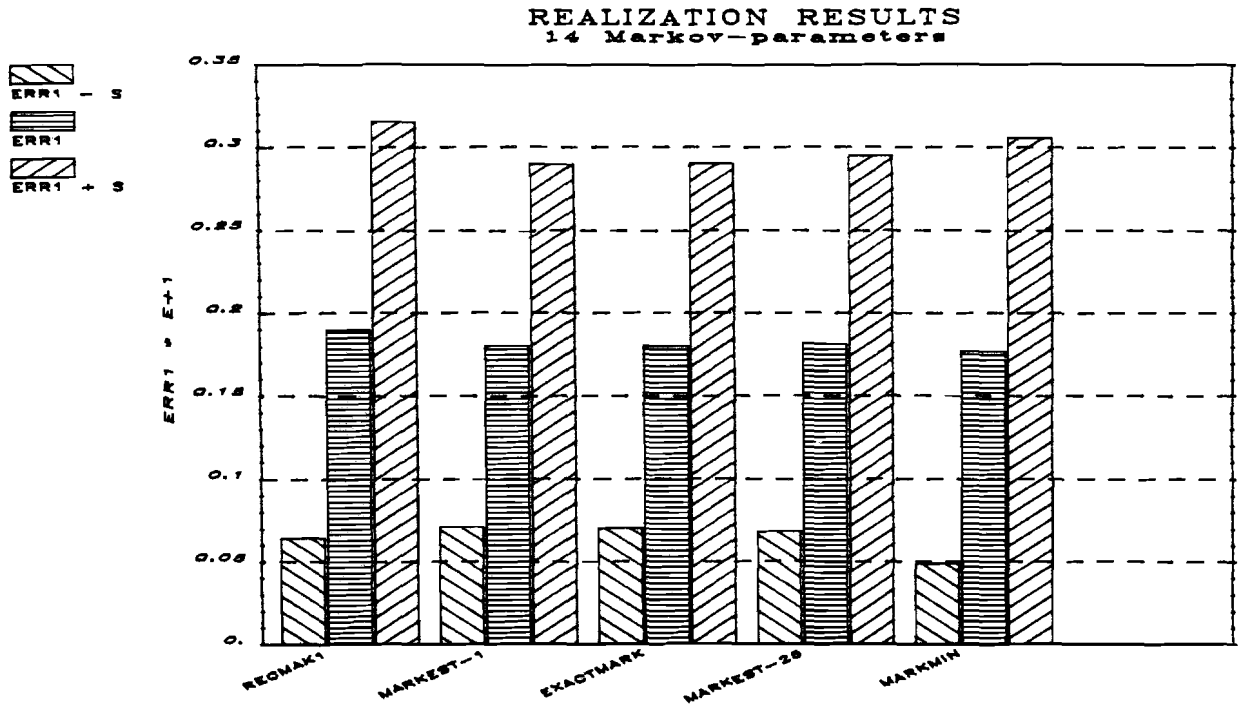


fig. 7.3.-4 Realization results of SYSTEM 2; input: filtered PRBNS

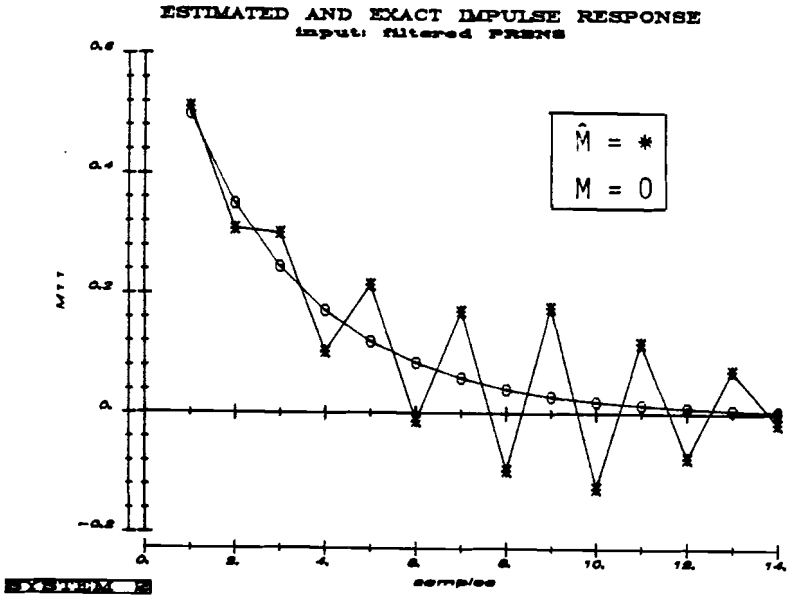


fig. 7.3.-5 'oscillating' estimated impulse response and exact impulse response of SYSTEM 2; input: filtered PRBNS

From the results of both experiments it is clear that in case of a large S/N-ratio the iterative methods don't give much improvement of the estimated Markov-parameters, while the non-recursive and non-iterative methods give us an accurate result, that is obtained very simply.

7.4 Simulations using SYSTEM 3

In the experiments, so far, we have chosen the noise level, that is added to the output in such a way that 'tail'-influences could be considered to be small in comparison to the noise.
In this section, we will notice what happens when the effects of the 'tail' of the impulse can not be neglected compared to the noise that is added

to the output.
When we consider the eigenvalues of SYSTEM 3 we see that this system has two very large eigenvalues and two small ones. Because of this property, this system will have a very slowly decreasing impulse response.
When we, now, only estimate 14 Markov-parameters and add a little bit of noise to the output, it is clear that the experiments must show effects of neglect of the 'tail'.
In this experiment only one kind of input signal is applied to the input of SYSTEM 3: white Gaussian noise (σ=1.0, μ=0.0).
The noise that is added to the output is obtained by filtering white, Gaussian channel independent noise (σ=0.1 and μ=0.0) with NOISE SYSTEM 2, resulting in a signal-to-noise ratio, caused by the noise, that is much smaller than the S/N-ratio caused by 'tail'-influences: S/N ≈ 5.
Tab. 7.4./1 and fig. 7.4.-1 give the results of this experiment (experiments using MARKMIND have not been done).

	RECMAK1	MARKEST-1	EXACTMARK	MARKEST-25	MARKMIN
$\overline{ERRO}^{20}$	0.3609 E-2	0.1390 E-2	0.1374 E-2	0.9443 E-3	0.4610 E-1
s.d.ERRO	0.1773 E-2	0.8245 E-3	0.8219 E-3	0.4585 E-3	0.2859 E-2

Tab. 7.4./1 Estimation results of SYSTEM 3; input: white noise

These results show a phenomenon that we already encountered in the previous section: The iterative procedure MARKMIN results in very bad Markov-parameters, although the minimum of the loss function, that is found, coincides with the noise level as supplied to the noise filter. This additive noise, however, is only a small part of the noise at the output because of the truncation of the impulse response. In this experiment we see again that the recursive method gives a much worse result than the other non-iterative methods, and minimization of the loss-function by successive substitution gives a small improvement, compared to the non-iterative methods. The minimum that is found is different from the

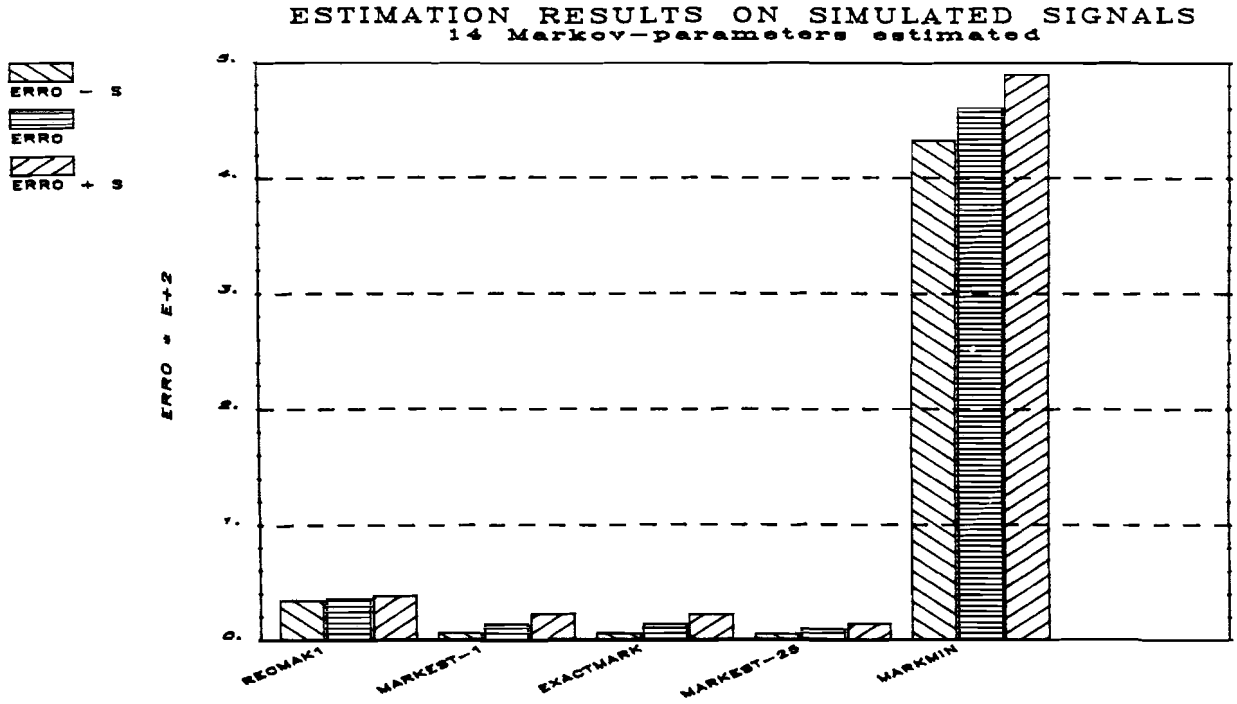


fig. 7.4.-1 Estimation results of SYSTEM 3; input: white noise

minimum that is found is different from the minimum found by MARKMIN and convergence to that minimum is very slow. The minimum is not always found. The same remarks are valid for the results of the realization, using the estimated Markov-parameters (tab. 7.4./2 and fig. 7.4.2).

	RECMAK1	MARKEST-1	EXACTMARK	MARKEST-25	MARKMIN
$ERR1^{20}$	0.2991 E-2	0.1056 E-2	0.1041 E-2	0.6333 E-3	0.4813 E-1
s.d.ERR1	0.1942 E-2	0.8119 E-3	0.8067 E-3	0.4532 E-3	0.2451 E-2

tab. 7.4./2 Realization results of SYSTEM 3; input: white noise

When we take a look at one of the Markov-parameter series $\hat{M}_{11}(i)$, as estimated by MARKMIN, we notice that this series tends to zero at the end, while the

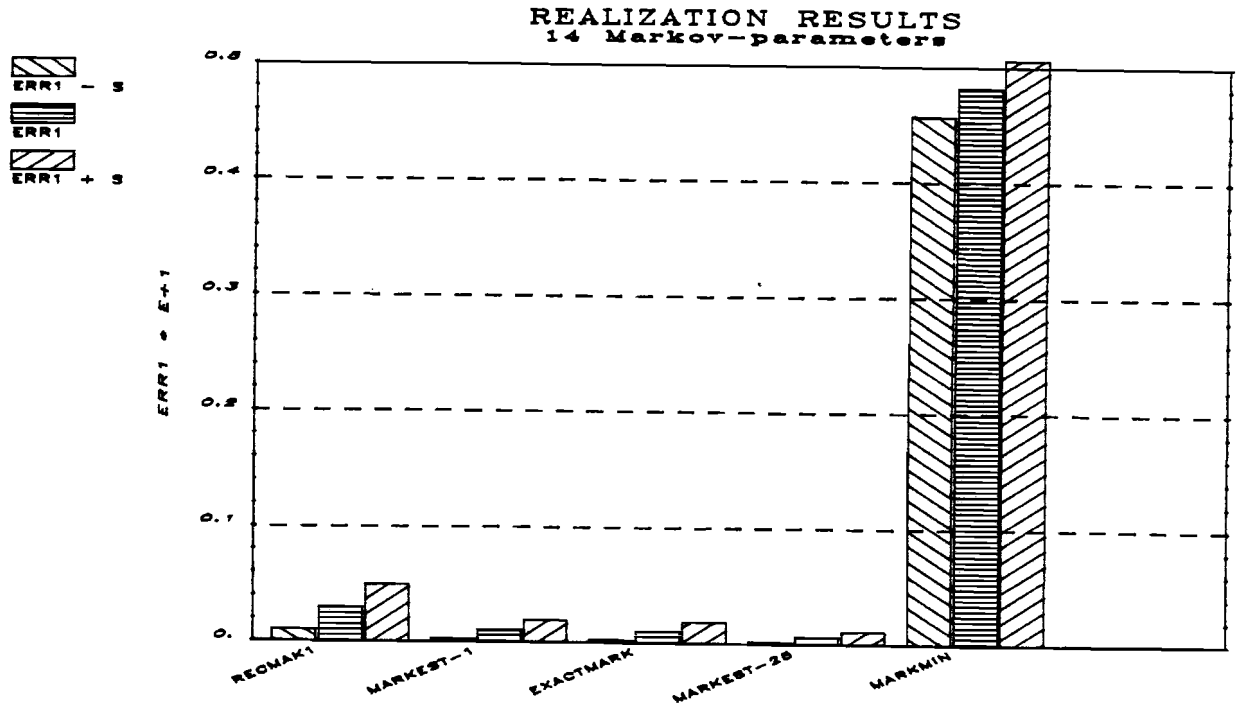


fig. 7.4.-2 Realization results of SYSTEM 3; inputs: white noise

series estimated by MARKEST-25 fluctuates around the exact impulse response $M_{11}(i)^*$. This is probably caused by the effect that MARKMIN tries to explain the 'tail' of the impulse response by the Markov-parameters of the system and the AR-parameters of the noise system.

The noise parameters that are estimated are not able to give a reasonable description of the noise model, as could be expected because of the truncation of the impulse response, which was not negligible.
The conclusion from these results is that the iterative procedure MARKMIN does not result in an unbiased estimate when 'tail'-influences occur, although the input is excited with white Gaussian noise.

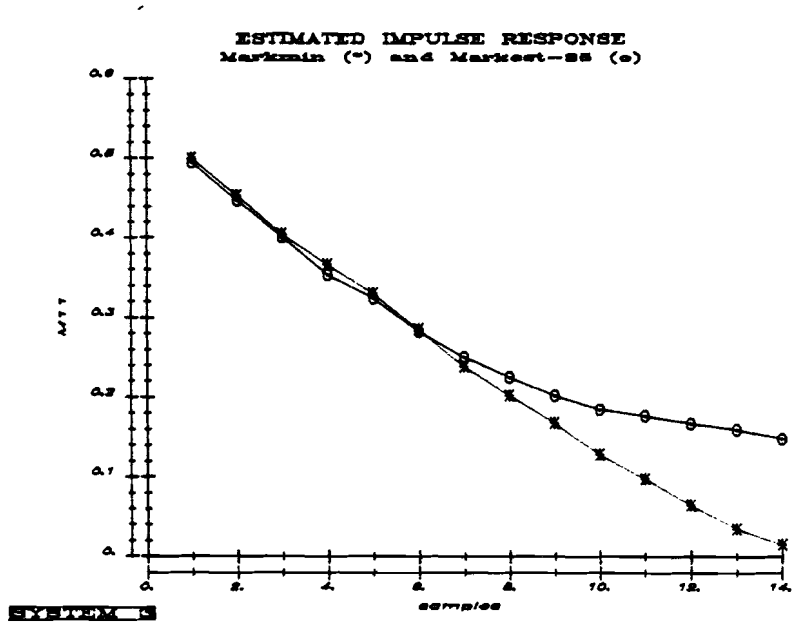


fig. 7.4.-3 Estimated impulse response by MARKMIN and MARKEST-25

7.5 Simulations when the additive noise contains a direct term

In this section we shall discuss an experiment which is the same as the first one of section 7.3, except for the character of the additive input noise. In this case, the output noise is not channel independent and thus the only method that gives an exact solution in the sense of a maximum likelihood estimate, is the one that minimizes $\det(Z^T.Z)$ as is described in 5.2 (MARKMIND).

The experiment of 7.3 with white noise as input of SYSTEM 2, has been repeated under the same condition except for the input-output matrix DN of the noise model that is now chosen to be:

$$DN = \begin{bmatrix} 1.0 & 0.3 \\ 0.4 & 0.6 \end{bmatrix} \qquad \text{eq. 7.5.(1)}$$

This implies that the output i of the noise filter is dependent of input

j at the same moment. Thus, in this case the moving-average parameter B(0) is unequal to the identity matrix, but

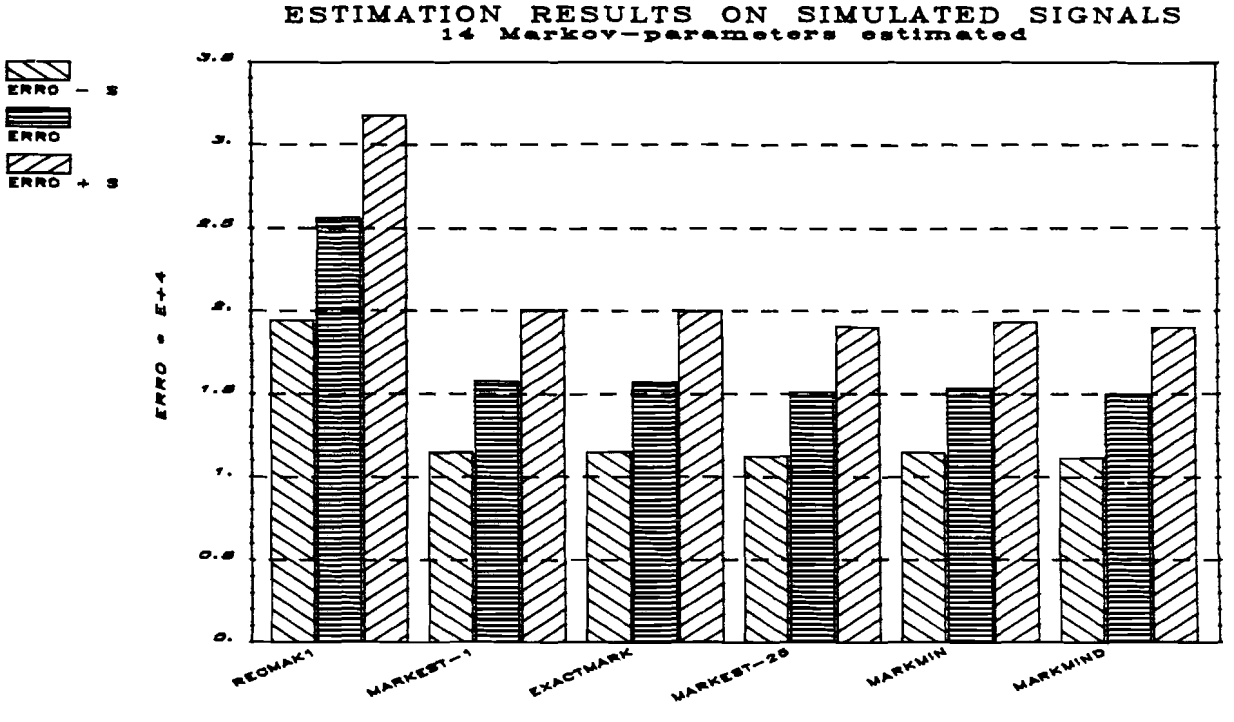
$$B(0) = DN \qquad \text{eq. 7.5.(2)}$$

The results of this experiment are presented in tab. 7.5./1 and fig. 7.5.-1.

	RECMAK1	MARKEST-1	EXACTMARK	MARKEST-25	MARKMIN	MARKMIND
$\overline{ERRO}^{20}$	0.2563 E-3	0.1575 E-3	0.1573 E-3	0.1515 E-3	0.1538 E-3	0.1505 E-3
s.d. ERRO	0.6170 E-4	0.4306 E-4	0.4231 E-4	0.3926 E-4	0.3939 E-4	0.3909 E-4

tab. 7.5./1 Estimation results of SYSTEM 2 + B(0); input: white noise

fig. 7.5.-1



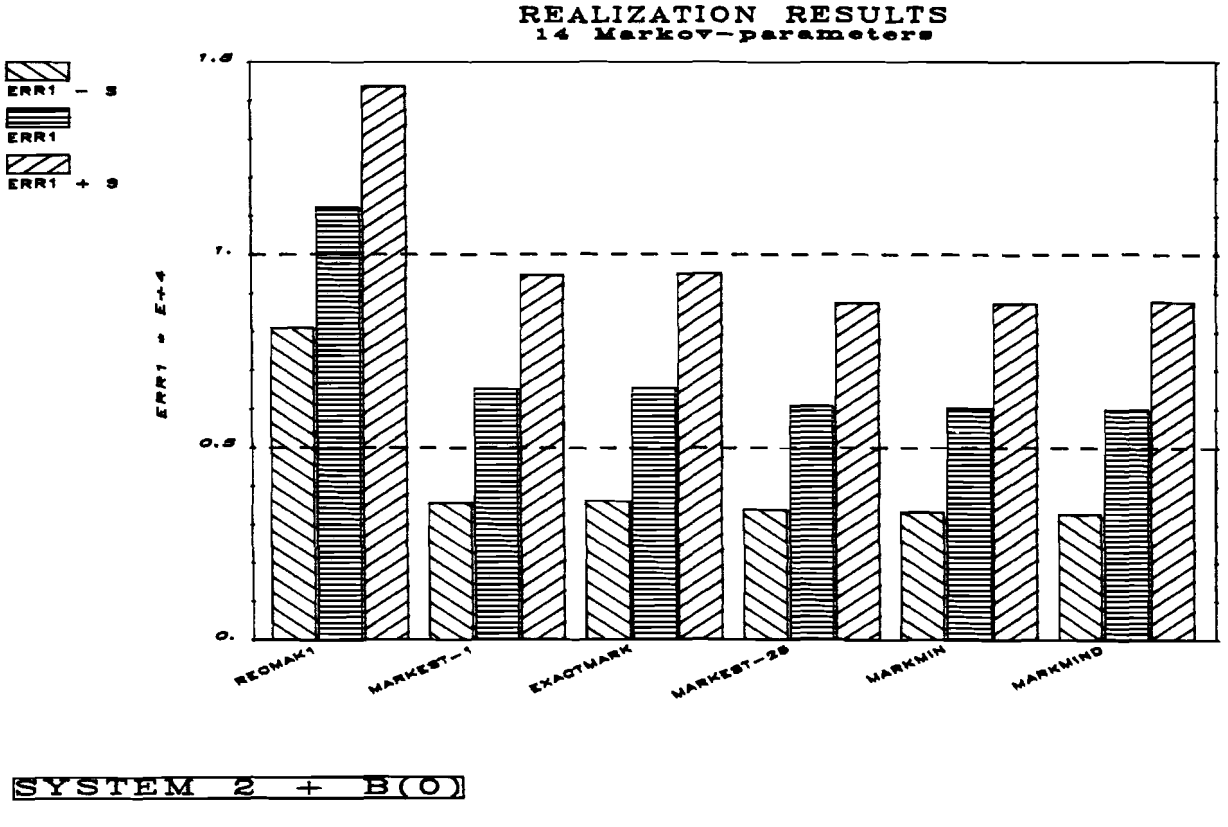
When we compare tab. 7.3./1 and tab. 7.5./1, the error in the estimated Markov-parameters has increased relatively more for the recursive method than for the other methods. The difference between MARKMIN and the other

iterative methods, as noticed in section 7.3 and 7.4, has vanished because the output noise is now larger than the noise due to truncation of the impulse response. Looking at fig. 7.5.-1, we see that, in case of a large S/N-ratio, an iterative method doesn't give much improvement, as was already found in section 7.3. From this experiment it is not possible to say, whether MARKMIND gives a better estimate or not, and thus more experiments with different noise levels and channel dependent noise are needed.
The same remarks can be made for the results of the realization that are presented in tab. 7.5./2 and fig. 7.5.-2.

	RECMAK1	MARKEST-1	EXACTMARK	MARKEST-25	MARKMIN	MARKMIND
$ERR1^{20}$	0.1123 E-3	0.6506 E-4	0.6545 E-4	0.6058 E-4	0.6012 E-4	0.5981 E-4
s.d.ERR1	0.3132 E-4	0.2948 E-4	0.2942 E-4	0.2676 E-4	0.2691 E-4	0.2722 E-4

tab. 7.5./2 Realization results of SYSTEM 2 + B(0); input: white noise

fig. 7.5.-2



7.6 Remarks

The main results of this chapter will be briefly summarized here.
In case of a small Signal-to-noise ratio ($1 < SNR < 10$) the use of an iterative method is very useful since it can give a substantial improvement of the quality of the estimated Markov-parameters under the condition that the noise is not caused by truncation of the impulse response.
When the signal-to-noise ratio at the output of the system is large ($SNR \geq 10$), the use of an iterative method does not result in a much better estimate and, therefore, in this case a non-iterative method may be good enough.
When we don't have any information about the signal-to-noise ratio, 1 iteration can be excuted resulting in an estimate of the parameters and of the residual, which gives an indication of the noise level. After determination of the eigenvalues of the system by applying the realization algorithm to the estimated Markov-parameters, we are able to find out whether more iterations are useful.

The auto-regressive noise parameters are well estimated by all methods if the noise level of the 'tail'-influences is small compared to the magnitude of the additive output noise.
In all experiments we find that the error in the estimated Markov-parameters, obtained by the recursive method is larger than the error in the parameters as found by any other method.
When the main part of the noise is caused by 'tail'-influences, the noise parameters are not found, but the auto-regressive parameters belonging to the process model (Except for the recursive method that was not able to find the auto-regressive parameters of the process from the noise caused by 'tail'-effects.). When the 'tail'-influences could not be neglected, iterations using 'hill-climbing' techniques resulted in a bad estimate of the Markov-parameters, due to the biasedness of the estimator (system not in model set).
When the input noise of the noise filter was not channel independent we didn't find very much difference from the case, where it was channel independent. This is probably caused by the large signal-to-noise ratio

in that particular experiment.
In the cases were we dealt with well conditioned input signals, the noise filtering by the realization algorithm resulted in a significant improvement of the estimates.

In case of output noise that is not caused by neglect of the 'tail' of the impulse response, the following table can be used that summarizes the most important properties of the estimation methods (tab. 7.6./1).

	accuracy			needed computation time and storage capacity
	large S/N-ratio no tail effects	small S/N-ratio no tail effects	tail effects	
RECMAK1	-	--	-	++
MARKEST-1	+	-	+	+
EXACTMARK	+	-	+	+
MARKEST-25	++	+	++	-
MARKMIN	++	++	--	--
MARKMIND	++	++	(*)	--

(*): not tested

tab. 7.6./1 Summary of the properties of the estimation methods

CONCLUSIONS

In this report several procedures have been developed for the estimation of Markov-parameters and noise parameters, based on the Least Squares and Maximum Likelihood principle. The modelling of the noise leads to a non-linear least squares problem. An approximate solution of this problem has been obtained by three non-iterative procedures and an exact solution was found by three iterative methods.
The properties of these methods have been investigated in theory and in simulation results, from which the following conclusions can be drawn:

- Dealing with a large Signal-to-noise ratio of the output signal, the Markov-parameters of the system and the parameters of the noise model are well estimated by both iterative and non-iterative methods, provided that the output noise is not caused by 'tail'-influences. This 'tail'-influence occurs when the system does not fit into the model set, since not enough Markov-parameters have been taken into account.

- A small Signal-to-noise ratio of the output signal leads to less accurate results for the non-iterative methods than for the iterative ones, under the condition that this Signal-to-noise ratio is not due to 'tail' effects.

- When not enough Markov-parameters are estimated, noise caused by 'tail'-influences will be present in the output signal. This leads to a biased estimate of the Markov-parameters for the iterative ML-estimation procedure using 'hill-climbing' techniques. The iterative LS-method using successive substitution, however, leads to an improvement of the estimated parameters compared to the results of the non-iterative methods.

- When 'tail'-influences have to be dealt with, the parameters of the noise model can not be determined by any of estimation procedures.

- In all cases, the recursive method results in an estimate that is less accurate than the ones, obtained by the alternative non-iterative procedures.

- In case of a process with a slowly decreasing impulse response that is observed with a high sampling rate, estimation of Markov-parameters is not very efficient because many parameters have to be estimated to eliminate 'tail'-influences. It is also not efficient because no use is made of the relation that exists between the Markov-parameters.

LIST OF SYMBOLS

A	System matrix (n×n)
$A(i)$	Auto-regressive MIMO-parameter
A_r	Matrix containing r auto-regressive MIMO-parameters
B_m	Matrix containing (shifted-) input data and (shifted-) equation errors
B	Distribution matrix (n×p)
$B(i)$	Moving-average MIMO-parameter
C	Output matrix (q×n)
Γ	(extended) observability matrix
D	Input-output matrix (q p)
DN	Input-output matrix of noise model
Δ	(extended) controllability matrix
δ	Singular value (δ_i)
E	Equation error matrix ((m+1)×q)
$\underline{e}(i)$	equation error vector at time instant i
F	System matrix of noise model
G	Distribution matrix of noise model
H	Hankel matrix
H	Output matrix of noise model
H_m	Matrix containing (shifted-) equation error vectors ((m+1)×r.q)
$K(z^{-1})$	Transfer function matrix
k	Number of Markov-parameters
L	Likelihood function
L	Number of Markov-parameters used for a realization
$M(i)$	Markov-parameter with index i
$\hat{M}(i)$	Estimated Markov-parameter with index i
$\tilde{M}(i)$	Reconstructed Markov-parameter with index i
M_k	Matrix containing (k+1) Markov-parameters
$\hat{M}_k$	Matrix containing (k+1) estimated Markov-parameters
$\tilde{M}_k$	Matrix containing (k+1) reconstructed Markov-parameters
M^*	Diagonal block matrix containing the matrices with Markov-parameters
m	Number of samples

N	Parameter matrix containing the Markov-parameters and the auto-regressive noise parameters
$\hat{N}$	Parameter matrix containing the estimated Markov- and AR-parameters
n	System dimension, order
p	Number of inputs
q	Number of outputs
R	Covariance matrix
r	Realizability index
r	Number of Auto-regressive noise parameters
S_m	Input matrix $((m+1)\times(k+1).p)$
Σ	Diagonal matrix of singular values
σ^2	Variance of the noise
$T(0,m)$	Generalized Toeplitz matrix
$\underline{u}(i)$	Input vector at time instant i $(p\times 1)$
V	Loss function
V	Matrix of right singular vectors
W	Matrix of left singular vectors
$\underline{x}(i)$	State vector at time instant i $(n\times 1)$
Y	Output matrix $((m+1)\times q)$
Y_m^*	Matrix containing output and shifted output data $((m+1)\times q.r)$
$\underline{y}(i)$	Output vector at time instant i $(q\times 1)$
Ξ	Noise matrix containing white, channel independent, Gaussian noise $((m+1)\times q)$
$\underline{\xi}(i)$	Noise vector at time instant i, containing white, channel independent noise
Z	Matrix containing white Gaussian noise that is not channel independent
$\underline{\zeta}(i)$	Noise vector containing white Gaussian noise that is not channel independent

Appendix A. Some aspects of transfer matrix, state space and ARMA representations

In this appendix we shall describe some aspects of a representation using the transfer function or state space representation and their relation to Auto Regressive (AR) and Moving Average (MA) models.

A.1 A representation using the transfer function and its relation to ARMA-models

The input-output relation of a MIMO-system can be described by a model, using the transfer matrix $K(z^{-1})$ as follows:

$$\underline{y}(k) = K(z^{-1}) \cdot \underline{u}(k) \qquad \text{eq. A.(1)}$$

where $K(z^{-1})$ is a rational matrix of variable z^{-1} of dimensions $q \times p$:

$$K(z^{-1}) = \begin{bmatrix} K_{11}(z^{-1}) & \cdots & K_{1p}(z^{-1}) \\ \vdots & & \vdots \\ K_{q1}(z^{-1}) & \cdots & K_{qp}(z^{-1}) \end{bmatrix} \qquad \text{eq. A.(2)}$$

and $K_{ij}(z^{-1})$ is the transfer function from input j to output i.

A disadvantage of this type of description with respect to identification procedures is the problem of non-uniqueness; by adding a pole to the denominator and a zero to the numerator of a transfer function we get the same input-output description, but, when estimating the coefficients of the transfer function, we get a different parameterization.

For this reason often models are used that represent only a part of the transfer function. When the transfer function is approximated by a polynomial, the model is called a Moving Average (MA) model. An approximation by the inverse of a polynomial is called an Auto Regressive

(AR) model.
To make this more clear, we decompose $K(z^{-1})$:

$$K(z^{-1}) = A^{-1}(z^{-1}) \cdot B(z^{-1}) \qquad \text{eq. A.(3)}$$

where: $A(z^{-1}) = (I+A(1)z^{-1}).(I+A(2)z^{-1})...(I+A(r_1)z^{-1})$

$B(z^{-1}) = B(0).(I+B(1)z^{-1}).(I+B(2)z^{-1})...(I+B(r_2)z^{-1})$

$r_1 = \text{rank } A(z^{-1}), \quad r_2 = \text{rank } B(z^{-1})$

$A(i)$: $q \times q$ -matrix, $B(i)$: $p \times p$ -matrix, $B(0)$: $q \times p$ -matrix.

Hence, $A(z^{-1})$ represents the denominator part of $K(z^{-1})$ and $B(z^{-1})$ the numerator part.

It is possible to approximate $K(z^{-1})$ by a Moving Average model by writing $A^{-1}(z^{-1})$ as:

$$A^{-1}(z^{-1}) = (I+A(r_1)z^{-1})^{-1}.(I+A(r_1-1)z^{-1})^{-1}...(I+A(1)z^{-1})^{-1}$$

eq. A.(4)

Since we only take into account stable systems, we will deal with systems of which the eigenvalues have an absolute value smaller than one. Hence we can use the series expansion as described by Gantmacher (1959):

$$(I - A^{-1}) = \sum_{i=0}^{\infty} A^i \qquad (|\lambda_k|<1 \ , k=1,2,...) \qquad \text{eq. A.(5)}$$

to write $A^{-1}(z^{-1})$, for $|z|=1$, as follows:

$$A^{-1}(z^{-1}) = (I-A(r_1)z^{-1}+A^2(r_1)z^{-2}-...).(I-A(r_1-1)z^{-1}+$$
$$A^2(r_1-1)z^{-2}-...)...(I-A(1)z^{-1}+A^2(1)z^{-2}...)$$

eq. A.(6)

$$K(z^{-1}) \simeq B(0) + \Big[B(0).\sum_{i=1}^{r_2} B(i) - \sum_{i=1}^{r_1} A(i).B(0)\Big].z^{-1} +$$

$$\Big[\sum_{i=1}^{r_1}\sum_{j=1}^{i-1} A(i).A(j).B(0) + B(0).\sum_{i=1}^{r_2}\sum_{j=1}^{i-1} B(i).B(j) +$$

$$- \sum_{i=1}^{r_1}\sum_{j=1}^{r_2} A(i).B(0).B(j) + \sum_{i=1}^{r_1} A^2(i).B(0)\Big].z^{-2} + \ldots$$

eq. A.(7)

The coefficients $B^*(i)$ of dimension $q \times p$, of z^{-i}, $i = 0,1,\ldots$ are called the moving average parameters.

Hence, it is possible to approximate eq. A.(1) by an infinite Moving Average model, where the tail, $i > nr$, can possibly be neglected.

$$\underline{y}(k) = \sum_{i=0}^{nr} B^*(i).z^{-i}.\underline{u}(k) \qquad \text{eq. A.(8)}$$

which is the same as:

$$\underline{y}(k) = B^*(0)\underline{u}(k) + B^*(1)\underline{u}(k-1) + \ldots + B^*(nr)\underline{u}(k-nr) \qquad \text{eq. A.(9)}$$

(nr is the number of parameters of the MA-model that we take into account).

It is also possible to approximate $K(z^{-1})$ by an Auto-Regressive model. Instead of

$$\underline{y}(k) = A^{-1}(z^{-1}).B(z^{-1}).\underline{u}(k) \qquad \text{eq. A.(10)}$$

we write :

$$\underbrace{B^{-1}(z^{-1}).A(z^{-1})}_{K^*(z^{-1})}.\underline{y}(k) = \underline{u}(k) \qquad \text{eq. A.(11)}$$

where $B(z^{-1})$ and $A(z^{-1})$ are redefined as:

$$A(z^{-1}) = A(0).(I+A(1)z^{-1}).(I+A(2)z^{-1})...(I+A(r_1)z^{-1})$$

$$B(z^{-1}) = (I+B(1)z^{-1}).(I+B(2)z^{-1})...(I+B(r_2)z^{-1})$$

and $A(i)$: $q \times q$ -matrix, $A(0)$: $p \times q$ -matrix, $B(i)$: $p \times p$ -matrix.

Under the restriction $|z| = 1$ and using A.(5), $B^{-1}(z^{-1})$ can be expanded into:

$$B^{-1}(z^{-1}) = (I-B(r_2)z^{-1}+B^2(r_2)z^{-2}-...).(I-B(r_2-1)z^{-1}+$$
$$B^2(r_2-1)z^{-2}-...)...(I-B(1)z^{-1}+B^2(1)z^{-2}...)$$

eq. A.(12)

Substitution of A.(12) in A.(11) leads to the following expression for $K^*(z^{-1})$ of dimension $p \times q$:

$$K^*(z^{-1}) \approx A(0) + \Big[A(0).\sum_{i=1}^{r_1} A(i) - \sum_{i=1}^{r_2} B(i).A(0)\Big].z^{-1} +$$
$$\Big[\sum_{i=1}^{r_2} B^2(i).A(0) + \sum_{i=1}^{r_2}\sum_{j=1}^{i-1} B(i).B(j).A(0) +$$
$$+ A(0).\sum_{i=1}^{r_1}\sum_{j=1}^{i-1} A(i).A(j) - \sum_{i=1}^{r_2}\sum_{j=1}^{r_1} B(i).A(0).A(j)\Big].z^{-2}+...$$

eq. A.(13)

The coefficients $A^*(i)$ of dimension $p \times q$, of z^{-i} , $i=0,1,...$, are called the auto-regressive parameters.

Thus it is possible to approximate eq. A.(11) by an infinite Auto Regressive model, where we may possibly neglect the tail $i>mr$:

$$(A^*(0) + A^*(1).z^{-1} + ... + A^*(mr).z^{-mr}).\underline{y}(k) = \underline{u}(k) \qquad \text{eq. A.(14)}$$

(mr is the number of auto-regressive parameters that is taken into account)

Eq. A.(14) can also be written as :

$$A^*(0).\underline{y}(k) = -A^*(1).\underline{y}(k-1) + \ldots - A^*(mr).\underline{y}(k-mr) + \underline{u}(k)$$

eq. A.(15)

Premultiplication of this equation by the pseudo-inverse $(A^*(0))^+$ of $A^*(0)$ leads to the auto-regressive model that has been used in section 2.3 for the description of the noise model.

A.2 The relation between a state space representation and ARMA-models

In this section we shall describe how we can obtain the auto-regressive parameters of an AR-model, when we have a state space description of a system.

Suppose, we have the following state space representation:

$$\underline{x}(k+1) = A \,.\, \underline{x}(k) + B \,.\, \underline{u}(k) \qquad \text{eq. 2.1.(2a)}$$

$$\underline{y}(k) = C \,.\, \underline{x}(k) + D \,.\, \underline{u}(k) \qquad \text{eq. 2.1.(2b)}$$

(For the meaning of the symbols is referred to section 2.1).
Combining eq. 2.1.(2a) and 2.1.(2b) we get a representation using a transfer function in terms of {A,B,C,D} :

$$\underline{y}(k) = \left(C \,.\, (zI - A)^{-1} . B + D \right) . \underline{u}(k) \qquad \text{eq. A.(16)}$$

Since all eigenvalues of A are assumed to have an absolute value smaller than one, we can use eq. A.(5) to expand eq. A.(16) for $|z|=1$:

$$\underline{y}(k) = z^{-1}.C.(I+z^{-1}.A+z^{-2}.A^2+\ldots).B.\underline{u}(k) + D.\underline{u}(k) \qquad \text{eq. A.(17)}$$

Which is the same as :

$$\underline{y}(k) = D.\underline{u}(k) + C.B.\underline{u}(k-1) + C.A.B.\underline{u}(k-2) + C.A^2.B.\underline{u}(k-3) + \ldots$$

eq. A.(18)

In eq. A.(18) we immediately recognize the convolution sum of the input signal with the Markov-parameters.
The model of eq. A.(18), using the Markov-parameters, is in fact a Moving Average model as can be seen by comparing eq. A.(18) with A.(9). Thus, it is very easy to obtain a Moving Average model, when having a state space description since the moving average parameters coincide with the Markov-parameters of the system.

We could transform eq. A.(18) to an auto-regressive model by the procedure explained in section A.1.
An alternative method we get by substituting eq. A.(18) in the right hand side of eq. A.(15):

$$A^*(0).\underline{y}(k) = -A^*(1).D.\underline{u}(k-1) - (A^*(2).D+A^*(1).C.B).\underline{u}(k-2) -\ldots + \underline{u}(k)$$

eq. A.(19)

Comparing eq. A.(19) with eq. A.(18), premultiplied by $A^*(0)$, gives us the following relation between the auto-regressive parameters $A^*(i)$ and $\{A,B,C,D\}$:

$$A^*(0).D = I \quad \text{and} \quad A^*(i).D = -\sum_{j=0}^{i-1} (A^*(j).C.A^{i-j-1}.B)$$

eq. A.(20)

The solution of the recurrent relation eq. A.(20) gives us the auto-regressive parameters $A^*(i)$ in terms of $\{A,B,C,D\}$. Because of the complexity of eq. A.(20) it is clear that this kind of changes from one model to another one is very unattractive.

Appendix B. Description and properties of the Singular Value Decomposition

In this appendix we shall give a review of the Singular Value Decomposition (SVD) and some of its properties. The SVD is an eigenvalue-like concept with remarkable numerical qualities, that is defined as follows:

Theorem 1. For any m×n matrix A, there exists a factorization:

$$A = W \,.\, \Sigma \,.\, V^T \qquad \text{eq. B.(1)}$$

where W is a (m×m)-orthonormal matrix (i.e. $W.W^T=I$),
V is a (n×n)-orthonormal matrix ($V.V^T=I$),
Σ is a (m×n)-pseudodiagonal matrix: $\Sigma = \mathrm{diag}(\sigma_1,\sigma_2,\ldots,\sigma_\rho)$;
$\rho = \min(m,n)$, $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_\rho \geq 0$. (cf. eq. B.(4))

The diagonal elements of Σ are called the singular values of A and the columns of W and the columns of V, the left respectively the right singular vectors of A.

The singular structure of A is very closely related to the eigen-structures of $A.A^T$ and $A^T.A$. This can easily be seen by substitution of eq. B.(1) and using the properties of V and W:

$$A \,.\, A^T = W \,.\, \Sigma \,.\, \Sigma^T \,.\, W^T \qquad \text{eq. B.(2)}$$

$$A^T \,.\, A = V \,.\, \Sigma^T \,.\, \Sigma \,.\, V^T \qquad \text{eq. B.(3)}$$

Hence the left (resp.right) singular vectors are the eigen-vectors of $A.A^T$ (resp.$A^T.A$), and the singular values of A are the positive square roots of the eigen-values of $A.A^T$ and $A^T.A$.

Because W and V are non-singular matrices, the rank of A is equal to the rank of Σ. This implies that the rank of A is equal to the number of non-zero singular values (counting multiplicities).

Let this rank be $r \leq \min(m,n)$ then it is possible to partition A as follows:

$$A = \underset{r \quad m-r}{[W_1 \; W_2]} \cdot \begin{matrix}\left[\begin{matrix}\Sigma_1 & 0\\ 0 & 0\end{matrix}\right] \begin{matrix})r\\ )m-r\end{matrix}\\ \underset{r \quad n-r}{}\end{matrix} \cdot \begin{matrix}\left[\begin{matrix}V_1^T\\ V_2^T\end{matrix}\right] \begin{matrix})r\\ )n-r\end{matrix}\end{matrix} = W_1 \cdot \Sigma_1 \cdot V_1^T \qquad \text{eq. B.(4)}$$

Using this partitioning it is also possible to approximate A by a matrix B with rank $r(B) < r$, so that $\|A-B\|_E$ is minimal, by setting the smallest $r-r(B)$ singular values in Σ, equal to zero.

When the matrix A is regarded as an operator that maps a $(n\times1)$-vector $\underline{x}$ into the range of A, which is a subspace of the m-dimensional space, by $\underline{y} = A.\underline{x}$, this operator is one-to-one and invertible:

$$\underline{y} = A \cdot \underline{x} = W_1.\Sigma_1.V_1^T \cdot \underline{x} \qquad \text{eq. B.(5)}$$

$$\underline{x} = A^+ \cdot \underline{y} = V_1.\Sigma_1^{-1}.W_1^T \cdot \underline{y} \qquad \text{eq. B.(6)}$$

where A^+ is the pseudo-inverse of A.
So A^+ is an operator that maps an $(m\times1)$-vector $\underline{y}$ into an n-dimensional space and eq. B.(6) gives us the least-squares solution $\underline{x}$ of the set of equations $\underline{y} = A \cdot \underline{x}$.

Finally, we shall give some results concerning the relation between the spectral-norm (s-norm) of a matrix and its singular values.

$$\|A\|_S \triangleq \max_{\underline{x}\neq\underline{0}} \frac{\|A\underline{x}\|_E}{\|\underline{x}\|_E} = \sigma_1 \qquad \text{eq. B.(7)}$$

$$\|A^+\|_s \triangleq \max_{\underline{y}\neq\underline{0}} \frac{\|A^+\underline{y}\|_E}{\|\underline{y}\|_E} = \sigma_r^{-1} \qquad \text{eq. B.(8)}$$

This means that the s-norm of a matrix A is equal to it's biggest singular value and the s-norm of it's pseudo-inverse A^+ is equal to the inverse of the smallest singular value of A unequal to zero.

Suppose $A\underline{x} = \underline{y}$ and $\Delta\underline{x}$ is the error in the solution $\underline{x}$ when $\underline{y}$ changes to $\underline{y} + \Delta\underline{y}$, so $A(\underline{x} + \Delta\underline{x}) = \underline{y} + \Delta\underline{y}$. Then the following expression is valid for the ratio between the relative errors in $\underline{x}$ and $\underline{y}$:

$$\frac{\|\Delta\underline{x}\|_E / \|\underline{x}\|_E}{\|\Delta\underline{y}\|_E / \|\underline{y}\|_E} = \frac{\|A^+.\Delta\underline{y}\|_E / \|\underline{x}\|_E}{\|\Delta\underline{y}\|_E / \|A\underline{x}\|_E} \leq \|A^+\|_s . \|A\|_s = \frac{\sigma_1}{\sigma_r} \qquad \text{eq. B.(9)}$$

and $\|A^+\|_s . \|A\|_s \triangleq C(A)$ is called the condition number of A. It is the most severe increase in relative inaccuracy when solving $\underline{x}$ from $A\underline{x} = \underline{y}$.

For more information about SVD and it's properties we refer to the extensive literature on this topic (e.g. Staar and Vandewalle (1982)).

Appendix C. Derivation of an updating algorithm for NLLS-estimation

In section 4.2 we found that an estimate $\hat{N} = \begin{bmatrix} \hat{M}_k \\ \hat{A}_r \end{bmatrix}$ could be found by solving:

$$\beta_m(\hat{M}_k).Y = \beta_m(\hat{M}_k).\beta_m{}^T(\hat{M}_k).\hat{N} \qquad \text{eq. C.(1)}$$

This non-linear equation is based on m samples and the exact solution is found by successive substitution of $\hat{N}$ in $\beta_m(\hat{M}_k)$ resulting in an iterative process that may converge to the exact solution of eq. C.(1) (if $i \to \infty$):

$$\beta_m(\hat{M}_{k,i}).Y = \beta_m(\hat{M}_{k,i}).\beta_m{}^T(\hat{M}_{k,i}).\hat{N}_{i+1} \qquad \text{eq. C.(2)}$$

where the indices i and i+1 point to the number of iterations that has been executed. Having found the solution of eq. C.(1), it is possible to update the matrices of this equation using a new sample m+1 and the final estimate $\hat{N}$ as it was found after m samples. This estimate is from now on called $\hat{N}_m$, where the index m points to the number of samples that is used to obtain this estimate.
In this appendix we shall only describe how the matrices $\beta_{m+1}(\hat{M}_k).Y$ and $\beta_{m+1}(\hat{M}_k).\beta_{m+1}{}^T(\hat{M}_k)$ can be obtained based on old data: $\beta_m(\hat{M}_k).Y$, $\beta_m(\hat{M}_k).\beta_m{}^T(\hat{M}_k)$, the old estimate $\hat{N}_m$ and a new sample of the input and output data. The set of equations that can be built by these matrices, may lead to an estimate $\hat{N}_{m+1}$ based on m+1 samples.

Using the notation of eq. 4.2.(11):

$$\beta_{m+1}{}^T(\hat{M}_{k,m}) = (S_{m+1}{}^T | H_{m+1}{}^T(\hat{M}_{k,m})) \text{ and } \hat{N}_{m+1,0} = \begin{bmatrix} \hat{M}_{k,m+1,0} \\ \hat{A}_{r,m+1,0} \end{bmatrix} \qquad \text{eq. C.(3)}$$

Eq. C.(1) can be rewritten (based on m+1 samples) as:

$$\begin{bmatrix} S_{m+1}.Y_{m+1} \\ H_{m+1}(\hat{M}_{k,m}).Y_{m+1} \end{bmatrix} = \begin{bmatrix} S_{m+1}.S_{m+1}{}^T & S_{m+1}.H_{m+1}{}^T(\hat{M}_{k,m}) \\ H_{m+1}(\hat{M}_{k,m}).S_{m+1}{}^T & H_{m+1}.H_{m+1}{}^T(\hat{M}_{k,m}) \end{bmatrix} \cdot \begin{bmatrix} \hat{M}_{k,m+1,0} \\ \hat{A}_{r,m+1,0} \end{bmatrix}$$

eq. C.(4)

Solution of eq. C.(4) results in a first estimate $\hat{N}_{m+1,0}$ that may converge to $\hat{N}_{m+1}$ by successive substitution in eq. C.(4).

NOTE: From now on $\hat{A}_r$ is assumed to contain r+1 auto-regressive parameters.

To 'build up' this equation we need to find resp. $S_{m+1}.Y_{m+1}$, $H_{m+1}.V_{m+1}$,

$S_{m+1}.S_{m+1}^T$, $S_{m+1}.H_{m+1}^T$ and $H_{m+1}.H_{m+1}^T$.
Let us start with the updating of $H_m.H_m^T$ to $H_{m+1}.H_{m+1}^T$.
As we have seen in section 4.2 H_m^T can be written as:

$$H_m^T = Y_m^* - \left[S_{m,\ell-1}^T \middle| S_{m,\ell-2}^T \middle| \ldots \middle| S_{m,\ell-r-1}^T\right] . M^* \qquad \text{eq. 4.2.(3)}$$

where the 'shorthand' notation is used that was introduced in 4.2.
Using this equation $H_{m+1}.H_{m+1}^T$ leads to :

$$\begin{aligned} H_{m+1}.H_{m+1}^T = {} & Y_{m+1}^{*T}.Y_{m+1}^* - Y_{m+1}^{*T}.\left[S_{m+1,\ell-1}^T \middle| \ldots \middle| S_{m+1,\ell-r-1}^T\right] .M^* + \\ & - M^{*T}.\left[S_{m+1,\ell-1}^T \middle| \ldots \middle| S_{m+1,\ell-r-1}^T\right]^T.Y_{m+1}^* + \\ & + M^{*T}.\left[S_{m+1,\ell-1}^T \middle| \ldots \middle| S_{m+1,\ell-r-1}^T\right]^T. \\ & .\left[S_{m+1,\ell-1}^T \middle| \ldots \middle| S_{m+1,\ell-r-1}^T\right] .M^* \end{aligned} \qquad \text{eq. C.(5)}$$

First, we take a look at $Y_{m+1}^{*T}.Y_{m+1}^*$:

$$Y_{m+1}^{*T}.Y_{m+1}^* = \left[Y_m^{*T} \middle| \begin{matrix} \underline{y}(\ell+m) \\ . \\ . \\ \underline{y}(\ell+m-r) \end{matrix}\right] . \left[\begin{matrix} Y_m^* \\ \hline \underline{y}^T(\ell+m)\ldots\underline{y}^T(\ell+m-r) \end{matrix}\right] \qquad \text{eq. C.(6)}$$

$$= Y_m^{*T}.Y_m^* + \begin{bmatrix} \underline{y}(\ell+m) \\ \vdots \\ \underline{y}(\ell+m-r) \end{bmatrix} . \left(\underline{y}^T(\ell+m)\ldots\underline{y}^T(\ell+m-r) \right) \qquad \text{eq. C.(7)}$$

From eq. C.(7) we see that $Y_m^{*T}.Y_m^*$ is updated by addition of a dyadic product of a vector containing the last r+1 output signals.
The second term of the right hand side of eq. C.(5) will not be computed since it is equal to the transpose of the third term.

To be able to compute this part we need :

$$(S_{m+1,\ell-1}{}^T|\ldots|S_{m+1,\ell-r-1}{}^T)^T.Y_{m+1}{}^* =$$

$$= \left[\begin{array}{c|c} & \underline{u}(\ell+m) \\ S_{m,\ell-1} & \vdots \\ & \underline{u}(\ell+m-k) \\ \hline & \vdots \\ \hline & \underline{u}(\ell+m-r) \\ S_{m,\ell-r-1} & \vdots \\ & \underline{u}(\ell+m-r-k) \end{array}\right] \cdot \left[\frac{Y_m{}^*}{\underline{y}^T(\ell+m)\ldots\underline{y}^T(\ell+m-r)}\right] =$$

$$(S_{m,\ell-1}{}^T|\ldots|S_{m,\ell-r-1}{}^T)^T.Y_m{}^* + \left[\begin{array}{c} \underline{u}(\ell+m) \\ \vdots \\ \underline{u}(\ell+m-k) \\ \hline \vdots \\ \hline \underline{u}(\ell+m-r) \\ \vdots \\ \underline{u}(\ell+m-r-k) \end{array}\right] .(\underline{y}^T(\ell+m)\ldots\underline{y}^T(\ell+m-r)) \qquad \text{eq. C.(8)}$$

This can be regarded as the addition of a dyadic product of two vectors to a matrix containing the data, known after m samples. The entire third term of the right hand side of eq. C.(5) is found by premultiplication of C.(8) by M^* containing the Markov-parameters as known after m samples. The last part we have to compute is:

$$(S_{m+1,\ell-1}{}^T|\ldots|S_{m+1,\ell-r-1}{}^T)^T.(S_{m+1,\ell-1}{}^T|\ldots|S_{m+1,\ell-r-1}{}^T) =$$

$$= (S_{m,\ell-1}{}^T|\ldots|S_{m,\ell-r-1}{}^T)^T.(S_{m,\ell-1}{}^T|\ldots|S_{m,\ell-r-1}{}^T) +$$

$$+ \left[\begin{array}{c} \underline{u}(\ell+m) \\ \cdot \\ \underline{u}(\ell+m-k) \\ \hline \cdot \\ \hline \underline{u}(\ell+m-r) \\ \cdot \\ \underline{u}(\ell+m-k-r) \end{array}\right] .(\underline{u}^T(\ell+m)\ldots\underline{u}^T(\ell+m-k)|\ldots|\underline{u}^T(\ell+m-r)\ldots\underline{u}^T(\ell+m-k-r))$$

eq. C.(9)

Pre- and postmultiplying C.(9) by M^* gives us the last term of the right hand side of eq. C.(5).

Resuming, we need 3 matrices to be able to compute $H_{m+1}.H_{m+1}{}^T$:

1. a matrix containing the sum of the dyadic products of vectors consisting of output signals
2. a matrix containing the sum of the dyadic products of a vector of output signals and a vector consisting of input signals
3. a matrix containing the sum of the dyadic products of vectors consisting of input signals.

The next block matrix which has to be updated is :

$$S_{m+1}.H_{m+1}{}^T = S_{m+1,\ell}.H_{m+1}{}^T \qquad \text{eq. C.(10)}$$

Using eq. 4.2.(3) we find:

$$S_{m+1,\ell}.H_{m+1}{}^T = S_{m+1,\ell}.Y_{m+1}{}^* - \left[S_{m+1,\ell}.S_{m+1,\ell}{}^T|\cdots|S_{m+1,\ell}.S_{m+1,\ell-r-1}{}^T\right].M^*$$

eq. C.(11)

Working out this equation, we get :

$$S_{m+1,\ell}.H_{m+1}^T = S_{m,\ell}.Y_m^* + \begin{bmatrix} \underline{u}(\ell+m+1) \\ . \\ \underline{u}(\ell+m-k+1) \end{bmatrix} . (\underline{y}^T(\ell+m)\ldots\underline{y}^T(\ell+m-r)) +$$

$$- \left[S_{m,\ell}.S_{m,\ell-1}^T \middle| \cdots \middle| S_{m,\ell}.S_{m,\ell-r-1}^T\right] . M^* +$$

$$- \begin{bmatrix} \underline{u}(\ell+m+1) \\ . \\ \underline{u}(\ell+m-k+1) \end{bmatrix} .(\underline{u}^T(\ell+m)\ldots\underline{u}^T(\ell+m-k) \middle| \ldots \middle| \underline{u}^T(\ell+m-r)\ldots\underline{u}^T(\ell+m-k-r)) . M^*$$

eq. C.(12)

$S_{m+1}.S_{m+1}^T$ can be found as follows:

$$S_{m+1}.S_{m+1}^T = S_{m,\ell}.S_{m,\ell}^T + \begin{bmatrix} \underline{u}(\ell+m+1) \\ . \\ \underline{u}(\ell+m-k+1) \end{bmatrix} .(\underline{u}^T(\ell+m+1)\ldots\underline{u}^T(\ell+m+1-k))$$

eq. C.(13)

When we compare the matrices needed in eq. C.(12) and C.(13) with the matrices that we needed for the updating of $H_m.H_m^T$ we see that they can be obtained by extension of the vectors that were used to update $H_m.H_m^T$. The same remark can be made for the matrices that we need to update $S_{m+1}.Y_{m+1}$ and $H_{m+1}.Y_{m+1}$, as:

$$S_{m+1}.Y = S_{m+1}.Y_{m+1} = S_{m,\ell}.Y_m + \begin{bmatrix} \underline{u}(\ell+m+1) \\ . \\ \underline{u}(\ell+m-k+1) \end{bmatrix} .(\underline{y}^T(\ell+m+1)) \qquad \text{eq. C.(14)}$$

and

$$H_{m+1}.Y = H_{m+1}.Y_{m+1} = Y_{m+1}^{*T}.Y_{m+1} - M^{*T}.\left[S_{m+1,\ell-1}^T \middle| \cdots \middle| S_{m+1,\ell-r-1}^T\right]^T.Y_{m+1} =$$

$$Y_m^{*T}.Y_m + \begin{bmatrix} \underline{y}(\ell+m) \\ . \\ y(\ell+m-r) \end{bmatrix} .\underline{y}^T(\ell+m+1) - M^{*T}.\left[S_{m,\ell-1}^T \middle| \cdots \middle| S_{m,\ell-r-1}^T\right]^T.Y_m +$$

$$- M^{*T} \cdot \left[\begin{array}{c} \underline{u}(\ell+m) \\ \cdot \\ \underline{u}(\ell+m-k) \\ \hline \cdot \\ \hline \underline{u}(\ell+m-r) \\ \cdot \\ \underline{u}(\ell+m-k-r) \end{array}\right] \cdot \underline{y}^T(\ell+m+1) \qquad \text{eq. C.(15)}$$

Taking into account all matrices, we need to update the following three matrices :

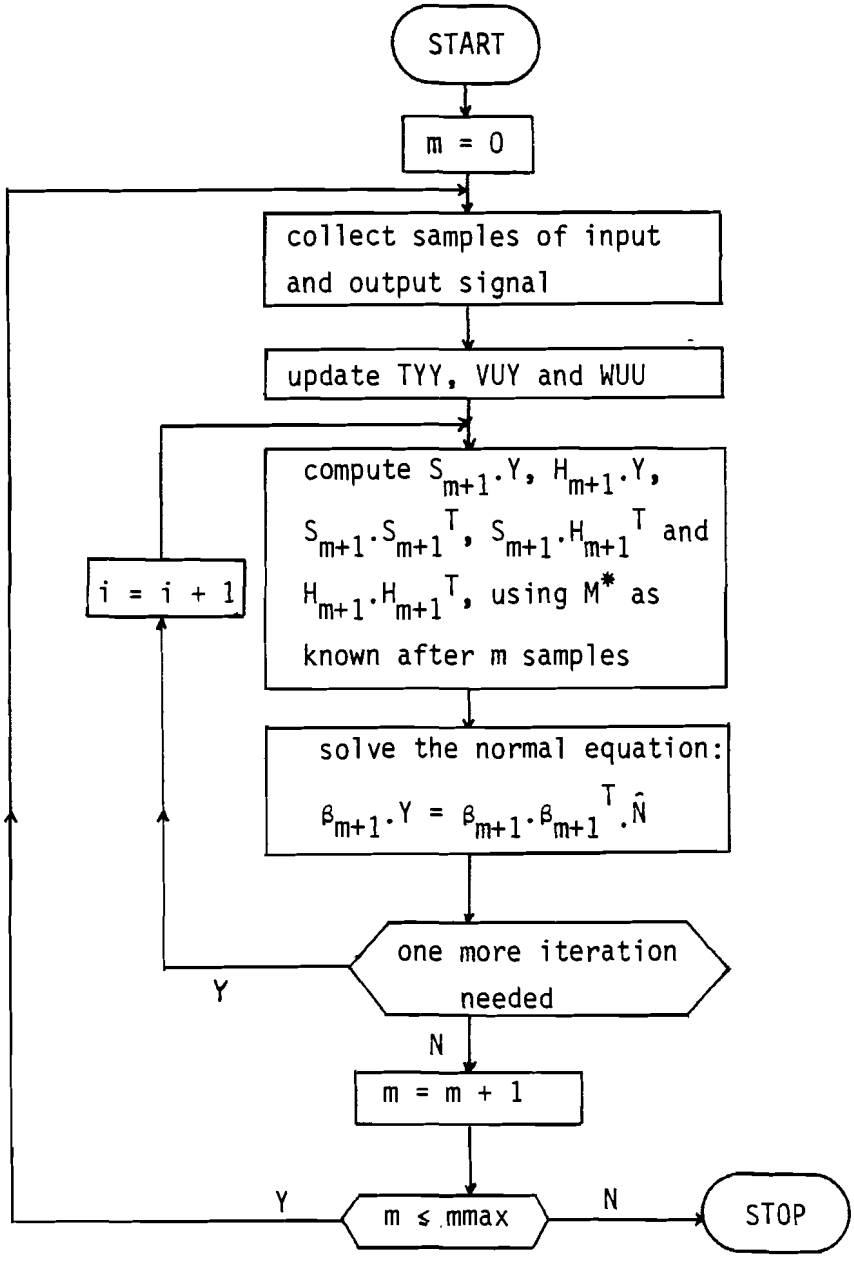
$$TYY_{m+1} = TYY_m + \left[\begin{array}{c} \underline{y}(\ell+m) \\ \cdot \\ \underline{y}(\ell+m-r) \end{array}\right] \cdot \left[\underline{y}^T(\ell+m+1) \,\middle|\, \underline{y}^T(\ell+m) \ldots \underline{y}^T(\ell+m-r)\right] \qquad \text{eq. C.(16)}$$

$$VUY_{m+1} = VUY_m + \left[\begin{array}{c} \underline{u}(\ell+m+1) \\ \cdot \\ \underline{u}(\ell+m+1-k) \\ \hline \cdot \\ \hline \underline{u}(\ell+m-r) \\ \cdot \\ \underline{u}(\ell+m-k-r) \end{array}\right] \cdot \left[\underline{y}^T(\ell+m+1) \,\middle|\, \underline{y}^T(\ell+m) \ldots \underline{y}^T(\ell+m-r)\right] \qquad \text{eq. C.(17)}$$

$$WUU_{m+1} = WUU_m + \left[\begin{array}{c} \underline{u}(\ell+m+1) \\ \cdot \\ \underline{u}(\ell+m+1-k) \\ \hline \cdot \\ \hline \underline{u}(\ell+m-r) \\ \cdot \\ \underline{u}(\ell+m-k-r) \end{array}\right] \cdot \left[\underline{u}^T(\ell+m+1) \ldots \underline{u}^T(\ell+m+1-k) \,\middle|\, \ldots \,\middle|\, \underline{u}^T(\ell+m-r) \ldots \underline{u}^T(\ell+m-k-r)\right] \qquad \text{eq. C.(18)}$$

where TYY is an $(r+1).q \times (r+2).q$-matrix
VUY an $(r+2).(k+1).p \times (r+2).q$-matrix and
WUU an $(r+2).(k+1).p \times (r+2).(k+1).p$-matrix .

These three fairly large matrices contain all information that is needed to update resp. $S_m.Y$, $H_m.Y$, $S_m.S_m^T$, $S_m.H_m^T$ and $H_m.H_m^T$. So eq. C.(1) can now be solved based on m+1 samples, which gives us the possibility to compute a new estimate $\hat{N}$ every time a new sample becomes available. This leads to the following algorithm:



Appendix D Formulae concerning matrix calculus

In this appendix we shall give an enumeration of a number of formulae concerning matrix calculus, that have been used in this report.

First we will deal with some properties of the derivatives of a vector with respect to a vector.
The derivative of the vector $\underline{y}$ with respect to the vector $\underline{x}$ is the matrix that is defined as:

$$\frac{\partial \underline{y}^T}{\partial \underline{x}} \triangleq \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \frac{\partial y_2}{\partial x_1} & \cdots & \frac{\partial y_m}{\partial x_1} \\ \frac{\partial y_1}{\partial x_2} & \frac{\partial y_2}{\partial x_2} & \cdots & \frac{\partial y_m}{\partial x_2} \\ \vdots & \vdots & & \vdots \\ \frac{\partial y_1}{\partial x_n} & \frac{\partial y_2}{\partial x_n} & \cdots & \frac{\partial y_m}{\partial x_n} \end{bmatrix} \qquad \text{eq. D.(1)}$$

of order $(n \times m)$ where $y_1, y_2, \ldots, y_m$ and $x_1, x_2, \ldots, x_n$ are the components of $\underline{y}$ and $\underline{x}$ respectively.
Using this definition the following properties can be derived:

$$\frac{\partial (A.\underline{x})^T}{\partial \underline{x}} = A^T \qquad \text{eq. D.(2)}$$

$$\frac{\partial \underline{x}^T.A}{\partial \underline{x}} = A \qquad \text{eq. D.(3)}$$

$$\frac{\partial \underline{x}^T.\underline{x}}{\partial \underline{x}} = 2.\underline{x} \qquad \text{eq. D.(4)}$$

$$\frac{\partial \underline{x}^T.A.\underline{x}}{\partial \underline{x}} = A.\underline{x} + A^T.\underline{x} \qquad \text{eq. D.(5)}$$

where A is an arbitrary matrix with the correct dimensions for multiplication.

Secondly, we shall consider the derivative of a scalar function of a matrix with respect to a matrix.

Let X be a matrix of dimension (m×n) and let y = f(X) be a scalar function of X. The derivative of y with respect to X, denoted by $\frac{\partial y}{\partial X}$ is the following matrix:

$$\frac{\partial y}{\partial X} \triangleq \begin{bmatrix} \frac{\partial y}{\partial x_{11}} & \frac{\partial y}{\partial x_{12}} & \cdots & \frac{\partial y}{\partial x_{1n}} \\ \frac{\partial y}{\partial x_{21}} & \frac{\partial y}{\partial x_{22}} & \cdots & \frac{\partial y}{\partial x_{2n}} \\ \vdots & \vdots & & \vdots \\ \frac{\partial y}{\partial x_{m1}} & \frac{\partial y}{\partial x_{m2}} & \cdots & \frac{\partial y}{\partial x_{mn}} \end{bmatrix} \qquad \text{eq. D.(6)}$$

Given this definition, it is possible to derive the following expressions:

$$\frac{\partial(\underline{a}^T.X.\underline{a})}{\partial X} = \underline{a}.\underline{a}^T \qquad \text{eq. D.(7)}$$

$$\frac{\partial(\underline{a}^T.X^T.X.\underline{a})}{\partial X} = 2.X.\underline{a}.\underline{a}^T \qquad \text{eq. D.(8)}$$

$$\frac{\partial(\underline{a}^T.X^{-1}.\underline{a})}{\partial X} = -(X^{-1}.\underline{a}.\underline{a}^T.X^{-1})^T \qquad \text{eq. D.(9)}$$

$$\frac{\partial\, tr(A.X)}{\partial X} = A^T \qquad \text{eq. D.(10)}$$

$$\frac{\partial\, tr(A^T.X^T.X.A)}{\partial X} = 2.X.A.A^T \qquad \text{eq. D.(11)}$$

$$\frac{\partial\, tr(X^T.A^T.A.X)}{\partial X} = 2.A^T.A.X \qquad \text{eq. D.(12)}$$

$$\frac{\partial\, \ln(\det X)}{\partial X} = (X^{-1})^T \qquad \text{eq. D.(13)}$$

Proofs of the formulae stated in this appendix and more information about matrix calculus can be found in Graham (1981).

REFERENCES

- Åström, K.J. (1980)
MAXIMUM LIKELIHOOD AND PREDICTION ERROR METHODS.
Automatica, Vol. 6, pp. 551-574, 1980.

- Backx, A.C.P.M., T.G.J. Beelen, A.J.T.M. Koenraads, and P.J. van Santen (1982)
IDENTIFICATION OF A DANNER PROCESS OF A GLASS FACTORY AT LOMMEL USING A MULTIVARIABLE EXTENSION OF THE EXTENDED MATRIX METHOD, (in Dutch).
Internal report, Philips Picos Glass, Eindhoven, 1982.

- Damen, A.A.H. and A.K. Hajdasinski (1982)
PRACTICAL TESTS WITH DIFFERENT APPROXIMATE REALIZATIONS BASED ON THE SINGULAR VALUE DECOMPOSITION OF THE HANKEL MATRIX.
Proc. of the 6th IFAC Symp. on Ident. and Syst. Param. Estim. , Washington D.C. , 1982.

- Damen, A.A.H. and P. Eykhoff (1982)
MAXIMUM LIKELIHOOD ESTIMATION
Journal A, Vol. 23, No. 4, pp. 183-188, 1982.

- Eykhoff, P. (1974)
SYSTEM IDENTIFICATION: PARAMETER AND STATE ESTIMATION.
London: Wiley and Sons, 1974.

- Franklin, G.F. and J.D. Powell (1980)
DIGITAL CONTROL OF DYNAMIC SYSTEMS.
Addison - Weley, Reading Mass., 1980.

- Gantmacher, F.R. (1959)
THE THEORY OF MATRICES, VOL. 1 and 2.
New York: Chelsea, 1959.

- Graham, A. (1981)
KRONECKER PRODUCTS AND MATRIX CALCULUS: WITH APPLICATIONS.
Ellis Horward limited, Chichester, 1981.

- Hajdasinski, A.K. (1976)
A MARKOV PARAMETER APPROACH TO IDENTIFICATION OF MULTIVARIABLE DYNAMICAL SYSTEMS.
Dep. of Electr. Eng., Eindhoven University of Technology, 1976; Internal Technical Report.

- Hajdasinski, A.K. (1978)
THE GAUSS-MARKOV APPROXIMATED SCHEME FOR IDENTIFICATION OF MULTIVARIABLE DYNAMICAL SYSTEMS VIA THE REALIZATION THEORY: AN EXPLICIT APPROACH.
Dep. of Electr. Eng., Eindhoven University of Technology, 1978; TH-Report 78-E-88.

- Hajdasinski, A.K. and A.A.H. Damen (1979)
REALIZATION OF THE MARKOV PARAMETER SEQUENCES USING THE SINGULAR VALUE DECOMPOSITION OF THE HANKEL MATRIX.
Dep. of Electr. Eng., Eindhoven University of Technology, 1979; TH-Report 79-E-95.

- Hajdasinski, A.K. (1980)
LINEAR MULTIVARIABLE SYSTEMS. Preliminary Problems in Mathematical Description, Modelling and Identification.
Dep. of Electr. Eng., Eindhoven University of Technology, 1980; TH-Report 80-E-106.

- Hajdasinski, A.K. (1980)
ESTIMATION OF THE ORDER OF MULTIVARIABLE DYNAMICAL SYSTEMS.
Journal A, Vol. 21, No. 1, pp. 21-32, 1980.

- Hajdasinski, A.K., P. Eykhoff, A.J.W. van den Boom and A.A.H. Damen (1982).
THE CHOICE AND USE OF DIFFERENT MODEL SETS FOR SYSTEM IDENTIFICATION.
Tutorial paper, 6th IFAC Symp. on Ident. and Syst. Param. Estim. , Washington D.C. , 1982.

- Sorenson, H.W. (1980)
PARAMETER ESTIMATION: PRINCIPLES AND PROBLEMS.
Marcel Dekker Inc., New York, 1980.

- Staar, J. and J. Vandewalle (1982)
SINGULAR VALUE DECOMPOSITION: A RELIABLE TOOL IN THE ALGORITHMIC ANALYSIS OF LINEAR SYSTEMS.
Journal A, Vol. 23, No. 2, pp. 69-74, 1982.

- Van den Hof, P.M.J. (1982)
APPROXIMATE REALIZATION OF NOISY LINEAR SYSTEMS: THE HANKEL AND PAGE MATRIX APPROACH.
Dep. of Electr. Eng., Eindhoven University of Technology, 1982; (M.Sc. Thesis).